Overview

WSO2 Enterprise Integrator (WSO2 EI) is a comprehensive integration solution that enables communication among various, disparate applications. Instead of having your applications communicate directly with each other in all their various formats, each application simply communicates with WSO2 EI, which acts mainly as an ESB to handle message transformations, and routes the messages to their appropriate destinations. The WSO2 EI product can be used to manage short-running, stateless integration flows (using the ESB profile) as well as long-running, stateful business processes (using the Business Process profile). The product also includes a separate Analytics profile for comprehensive monitoring, a Message Broker profile (WSO2 MB) that can be used for reliable messaging, a Micro Integrator profile that is container-friendly, as well as the WSO2 MSF4j profile, which you can use to run microservices for your integration flows.

The ESB profile in WSO2 EI provides its fundamental services through an event-driven and standards-based messaging engine (the bus), which allows integration architects to exploit the value of messaging without writing code. This ESB profile is a step ahead of the previous releases of WSO2 Enterprise Service Bus, as it provides data integration capabilities within the same runtime. This eliminates the need to use a separate data services server for your integration processes.

The Business Process profile in WSO2 EI enables developers to easily deploy long-running integration processes (business processes). These processes are written using the following: A subset of the BPMN 2.0 standard, WS-BPEL 2.0 and BPEL4People standards, and WS-Human Tasks. Powered by the Activiti BPMN Engine 5.21.0 and Apache Orchestration Director Engine (ODE) BPEL engine, the Business Process profile in WSO2 EI comes with a complete web-based graphical management console, enabling users to easily deploy, manage, view and execute processes as well as human tasks.

The Micro Integrator profile in WSO2 EI is designed to be container-friendly. Because its start-up time is faster than the ESB profile, the Micro Integrator profile allows you to perform advanced scenarios without sacrificing the speed required for a container-based architecture. Also, you can use the Micro Integrator profile to orchestrate atomic microservices into composite microservices that address more complex, real-world scenarios.

Therefore, WSO2 EI is essentially a collection of enterprise architecture design patterns (WSO2 ESB++) that can be implemented directly using a single product. This product is light-weight and versatile. It is 100% open source and is released under Apache Software License Version 2.0, one of the most business-friendly licenses available today.

Short-running integration flows

The following diagram illustrates the message-flow architecture in the ESB profile of WSO2 EI, which is used for implementing integration flows.

This shows how a request propagates to its actual endpoint through the ESB profile. Response handling is the reverse of this operation. Note that the components of the pipes are not in a specific order.
1. An application(client) sends a message to the **ESB** profile of WSO2 EI.
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 the Axis2 engine. The ESB profile can operate in two modes:
   - **Working with Mediators** - 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 the ESB profile of WSO2 EI, 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 that are required before sending the message to the receiver application.
7. The message is sent to the receiver application.

There are other areas like **Working with Scheduled Tasks** and **Events** that are not shown in the diagram. All these components can be **analyzed and monitored** using the EI-Analytics profile. Also, you can perform all message tracing activities via the EI-Analytics profile.

**Long-running business processes**

The following diagram illustrates the message-flow architecture in the Business Process profile of WSO2 EI, which is used for long-running, stateful business processes.

This is how a request message propagates through the Business Process profile:

1. An application(client) sends a message to the **Business Process** profile of WSO2 EI. Note that this request could be for a BPEL process, Human Task or a BPMN process.
2. The message is picked up by the transport layer, depending on whether it is intended for a BPEL process, Human Task or a BPMN process as explained below.
   - Business processes defined using BPEL, as well as Human Tasks are exposed as SOAP services. Therefore, the transport layer is essentially an Axis2 web service (corresponding to the BPEL process or the Human Task), which can receive messages from Axis2-supported transports such as HTTP, HTTPS, JMS, etc.
   - Business processes defined using BPMN are exposed as a secured REST API, which is built as a web application and deployed in embedded tomcat. Therefore, the transport layer for BPMN is essentially a web application that can receive RESTful messages.
   - Human Tasks allows you to integrate human interactions into the business process workflows.
3. The message will be forwarded to the relevant runtime: **BPEL runtime**, **BPMN runtime** or the **Human Task runtime**.
   - **BPEL Runtime** - The message will be processed against the compiled BPEL process definition. The **Integration Layer** sits between the ODE BPEL runtime and Axis2 for processing and forwarding.
received messages to ODE Runtime. If the message is received for a **BPEL process**, it goes through
the Axis2 engine, where QoS (quality of service) requirements such as WS-Addressing and WS-
Security will be processed using Axis2 modules such as rampart and addressing.

- **BPMN Runtime** - When REST requests are received by the BPMN REST API, it will perform relevant
  operations in the BPMN runtime against deployed BPMN processes. In addition to exposed REST
  service, admin services are provided for BPMN processes and instance management.
- **Human Task Runtime** - The received message will be executed against the human tasks defined in
  the system. If the execution is successful, an instance of the task will be created and relevant task
  instance data will be persisted to the database.

4. The **Data Access Objects Layer** is used to interact with the database for persisting BPEL process definition
   /instance data and Human Tasks instance data.