SMART-FI

Exploiting Aggregated Open Data from Smart Cities for the Future Internet Society

D5.2: Orchestration and adaption methodologies report v2

Authors	Caner Tosunoglu, Eser Karakaya, Sinan Yucekaya, Ehsan Valizadeh, Serdar Yumlu
Institution lead	SAMPAS
Version	1.0
Reviewers	Serdar Yumlu
Work package	WP5
Task	T5.1
Due date	30/03/2018
Submission date	14/09/2018
Distribution level (CO, PU):	Public
Abstract	This document introduce the WSO2 Enterprise Integrator, including its features, and its architecture.
Keywords	Message routing, Message filtering, Message transformation , Content enriching, Protocol switching, Service chaining, Message storing and forwarding, Load balancing, Message entry points, Message processing units, Message exit points , Message stores and

	processors, Connectors, Transports, Message builders and formatters, Applying security to artifacts , Logging messages, Message tracing, Debugging mediation, Enterprise Integration Patterns , ESB tooling, Deploying microservices framework, Working with microservices
License information	This work is licensed under Content licensed under CC By 4.0. Samples licensed under Apache 2.0.

Document Description

Document Revision History

Version	*Date*	*Modifications Introduced*	
		Description of change	*Modified by*
v0.1	20/09/17	First draft version and TOC	Serdar Yumlu (SAMPAS)
v0.2	25/09/17	contributions	Ehsan Valizadeh (SAMPAS)
v0.3	15/10/17	contributions	Eser Karakaya (SAMPAS)
v0.4	25/10/17	contributions	Sinan Yucekaya (SAMPAS)
v0.5	10/11/17	contributions	Ehsan Valizadeh (SAMPAS)
v0.6	02/04/18	contributions	Ehsan Valizadeh (SAMPAS)
v0.7	10/04/18	contributions	Sinan Yucekaya (SAMPAS)
V0.8	05/07/18	Internal review	Serdar Yumlu (SAMPAS)
V0.9	25/08/18	Overall revision of document and prepare final version to release	Ehsan Valizadeh (SAMPAS)
V1.0	14/09/18	Final version to release	Serdar Yumlu (SAMPAS)

Table of Contents

Table of Figures

Table of Tables

Terms and abbreviations

SMART-FI	Provides services for smart city applications, creates business opportunities and improves public's quality of life using Open Data
CC	Creative Commons is an American non-profit organization devoted to expanding the range of creative works available for others to build upon legally and to share
SOA	A service-oriented architecture is a style of software design where services are provided to the other components by application components, through a communication protocol over a network.
ESB	An enterprise service bus implements a communication system between mutually interacting software applications in a service-oriented architecture. It implements a software architecture as depicted on the right.
Data Service	Data services refers to service-oriented architecture (SOA) applied to data sourced from the World Wide Web and the Internet as a whole. Web data services enable maximal mashup, reuse, and sharing of structured data (such as relational tables), semi-structured information (such as Extensible Markup Language (XML) documents), and unstructured information (such as RSS feeds, content from Web applications, commercial data from online business sources).

WSO2	WSO2 is an open source technology provider that increases the agility of digital businesses and enterprises engaging in digital transformation.
Microservices	Microservices is a variant of the service-oriented architecture (SOA) architectural style that structures an application as a collection of loosely coupled services. In a microservices architecture, services should be fine-grained and the protocols should be lightweight.
WP	Work Package

Executive Summary

This document aims to introduce the WSO2 Enterprise Integrator as an open source Integration product used within SMART-FI platform WP5 tasks objectives, including its features, and its architecture.

WSO2 Enterprise Integrator is an all-in-one package for integration. It contains the enterprise service bus (ESB) profile for mediation and data services, the message broker profile to support reliable messaging, the business process profile for long running processes and the analytics profile for statistical requirements of the system.

1 Introduction

This document supposes the second version of Orchestration and adaptation methodologies Report related to the objectives of task 5.1. WSO2 Enterprise Integrator (EI) will be investigated and presented generally as an agile integration platform for quick, iterative integration of any application, data, or system.

WSO2 Enterprise Integrator version (EI) is a powerful integration solution, which mediates and transforms data between various systems and applications, to provide the fundamental capabilities of a connected SOA architecture. It comes complete with multiple runtimes (i.e., for analytics, reliable messaging, and business processes) and tooling, all embedded in a single release.

This WSO2 Enterprise Integrator brings together certain functionalities encapsulated in the following WSO2 products:

- WSO2 Enterprise Service Bus (ESB)
- WSO2 Data Services Server (DSS)
- WSO2 Application Server (AS)
- WSO2 Business Process Server (BPS)
- WSO2 Message Broker (MB)

The Integration runtime in EI consists of WSO2 ESB, WSO2 DSS and WSO2 AS products. WSO2 EI ships a separate Business Process runtime for handling long-running business processes. Also, it ships Message broker and Analytics as separate runtimes to add the capabilities of reliable messaging and analytics.

1.1 Document structure

The document is structured as follow. Section 1 is introduction. In section 2, we will introduce the Enterprise Integrator. Section 3 presents Common usage scenarios of WSO2 Enterprise Integrator and in section 4 we describe how to get started with WSO2 Enterprise Integrator. Section 5 presents the conclusion. Section 6 contains references.

2 Introducing the Enterprise Integrator

WSO2 Enterprise Integrator (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 the EI, which handles transforming and routing the messages to their appropriate destinations. The WSO2 EI product can be used to manage short-running, stateless integration flows (using the **Integrator** runtime) as well as long-running, stateful business processes (using the **Business Process Server** runtime). The product also includes a separate **Analytics** runtime for comprehensive monitoring as well as a **Broker** runtime (WSO2 MB) that can be used for reliable messaging.

The **Integration** runtime in 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 **Integrator** product is a step ahead of the ESB product introduced by WSO2 (WSO2 Enterprise Service Bus), as it allows back-end applications and services required for an integration process to be hosted within the same runtime. This eliminates the need to use a separate application server or data services server for your integration processes.

The **Business Process** runtime in 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** runtime in 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.

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.

2.1 Short-running integration flows

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

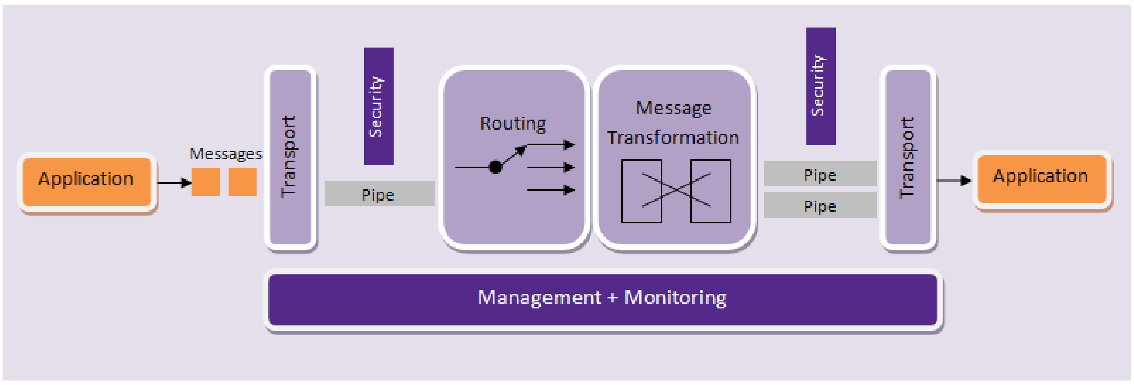


Figure 1 message-flow architecture in the Integration runtime of EI

This shows how a request propagates to its actual endpoint through the Integrator runtime. 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 **Integration** runtime 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 Integration runtime 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 the Integration runtime of 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 runtime. Also, you can perform all message tracing activities via the EI-Analytics profile.

2.2 Long-running business processes

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

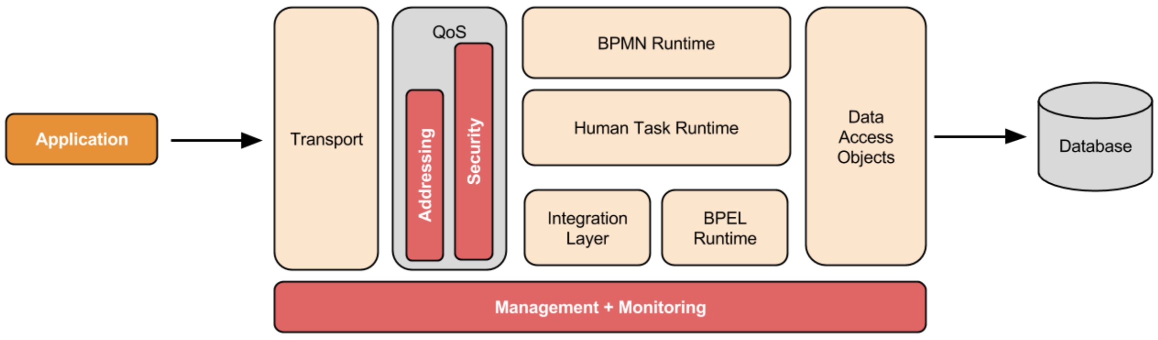


Figure 2 message-flow architecture in the Business Process runtime of EI

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

1. An application(client) sends a message to the **Business Process** runtime 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.
3. 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.
4. 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.
    - **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.
5. The Data Access Objects Layer is used to interact with the database for persisiting BPEL process definition/instance data and Human Tasks instance data.

3 Common usage scenarios of WSO2 Enterprise Integrator

This set of guides gives you a complete introduction to the fundamentals and most common usage scenarios of the Integration runtime in WSO2 Enterprise Integrator (WSO2 EI).

3.1 Integration Guides

In this guide we will use a connected healthcare service scenario where we will build on this use case as we progress through the guide.

The guide comprises of the following sections:

3.1.1 Sending a Simple Message

In this guide, you create a REST API in WSO2 Enterprise Integrator to connect to a REST back-end service that is defined as an HTTP Endpoint in WSO2 EI.

3.1.2 Routing Requests Based on Message Content

In this guide, you use a Switch mediator to route messages based on the message content to the relevant HTTP Endpoint defined in Enterprise Integrator.

3.1.3 Transforming Message Content

In this guide, you send a request message to a back-end service where the payload is in a different format when compared to the request payload expected by the back-end service. Data Mapper mediator in WSO2 EI is used to transform the request message payload to the format expected by the back-end service.

You send a message through the Enterprise Integrator to the back-end service using the Call mediator, instead of the Send mediator. Using the Call mediator, you can build a service chaining scenario as it allows you to specify all service invocations one after the other within a single sequence.

You then use the PayloadFactory mediator to take the response from one back-end service and change it to the format that is accepted by the other back-end service.

The diagram below depicts the sample scenario:

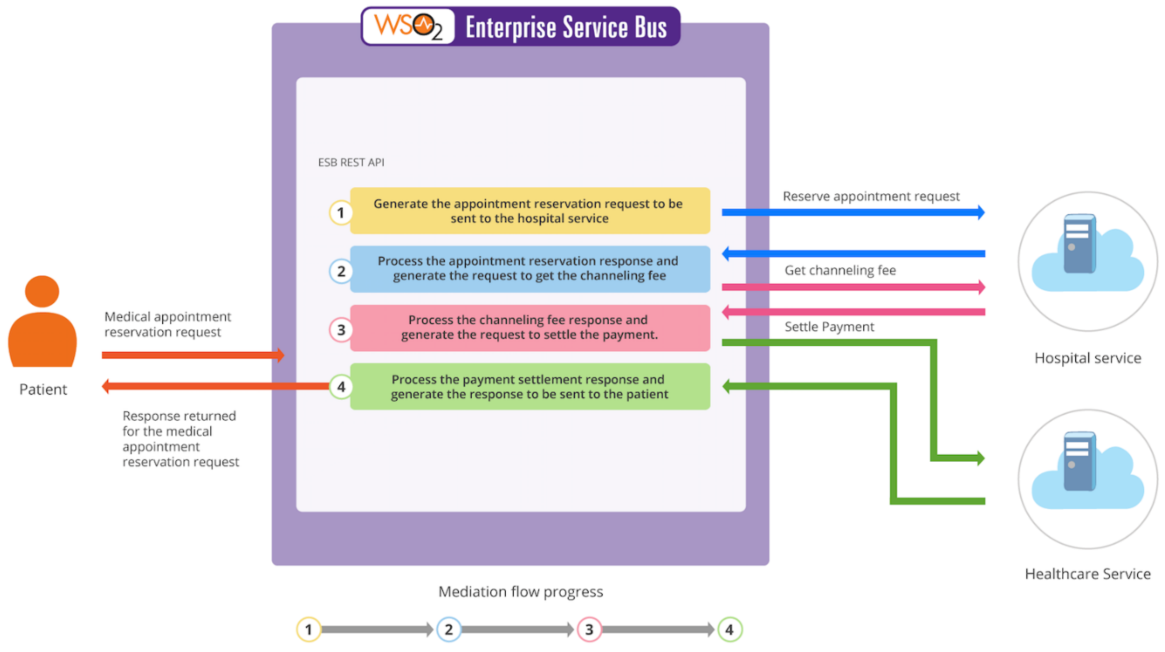

WSO2 Enterprise Service Bus
ESB REST API
1 Generate the appointment reservation request to be sent to the hospital service
2 Process the appointment reservation response and generate the request to get the channeling fee
3 Process the channeling fee response and generate the request to settle the payment.
4 Process the payment settlement response and generate the response to be sent to the patient
Reserve appointment request
Get channeling fee
Settle Payment
Hospital service
Healthcare Service
Patient
Medical appointment reservation request
Response returned for the medical appointment reservation request
Mediation flow progress
1
2
3
4

Figure 3 EI Mediation Flow Progress

3.1.4 Exposing Several Services as a Single Service

In WSO2 Enterprise Integrator (WSO2 EI) this is commonly referred to as Service Chaining where several services are integrated based on some business logic and exposed as a single, aggregated service.

3.1.5 Storing and Forwarding Messages

In this guide, instead of sending the request directly to the back-end service, you store the request message into an In Memory Message Store. You then use a Message Processor to retrieve the message from the store and then deliver the message to the back-end service.

3.1.6 Using the Gmail Connector

3.1.7 Using the Analytics Dashboard

3.2 Data Integration

The following guides will provide information on the main use cases of data integration, which includes creating data services to expose various datasources as services:

3.2.1 Creating a Data Service from Scratch

- Exposing an RDBMS as a Data Service
- Exposing a Google Spreadsheet as a Data Service
- Exposing CSV Data as a Data Service
- Exposing Excel Data as a Data Service
- Exposing RDF Data as a Data Service
- Exposing MongoDB as a Data Service
- Exposing a Web Resource as a Data Service
- Exposing a Carbon Datasource as a Data Service
- Exposing a Custom Datasource as a Data Service
- Exposing Cassandra as a Data Service
- Exposing a JNDI Datasource as a Data Service

3.2.2 Uploading a Created Data Service

After creating a data service (.dbs file) file manually from scratch, deploy it to a running WSO2 Enterprise Integrator (WSO2 EI) instance through the management console as explained below:

1. Log in to the management console and select **Upload** under **Data Service** .
2. Select the database backup (.dbs) file and click **Upload**. For example, you can upload one of the sample data services that are stored in the <EI_HOME>/samples/data-services/dbs folder.
3. If the file is deployed successfully, the **Deployed Services** window appears with the new data service listed. From here, you can manage your service. See Managing Data Services.

   Alternatively, copy the file to <EI_HOME>/repository/deployment/server/dataservices folder. It will be deployed instantly as hot deployment, which is enabled in Data Services Server by default.
   When adding a data service file to WSO2 EI for the first time, you will need to create a folder named dataservices in the following directory location: <EI_HOME>/repository/deployment/server.

3.2.3 Generating a Data Service

This option helps create data services automatically using a given database structure. When generating a data service, the server takes its table structure according to the structure specified in the datasource and automatically creates the SELECT, INSERT, UPDATE and DELETE operations.

- Step 1: Setting up a datasource
- Step 2: Creating a Carbon datasource
- Step 3: Generating a data service
- Step 4: Verifying the generated CRUD operations (Optional)
- Step 5: Invoking your data service

3.2.4 Exposing Data as a REST Resource

WSO2 Enterprise Integrator (WSO2 EI) allows you to expose data stored in various datasources as REST-style resources in addition to SOAP services. The guides on creating data services from scratch will guide you on how to expose data as SOAP services.

The following instructions explain how you can expose data as a REST resource:

- **Defining a REST resource**

  Let's take a data service that is already created using an RDBMS and expose the data as REST-style resources.

1. Follow the guide on exposing an RDBMS as a data service to create the **RDBMS** data service.
2. Log in to the product's Management Console.

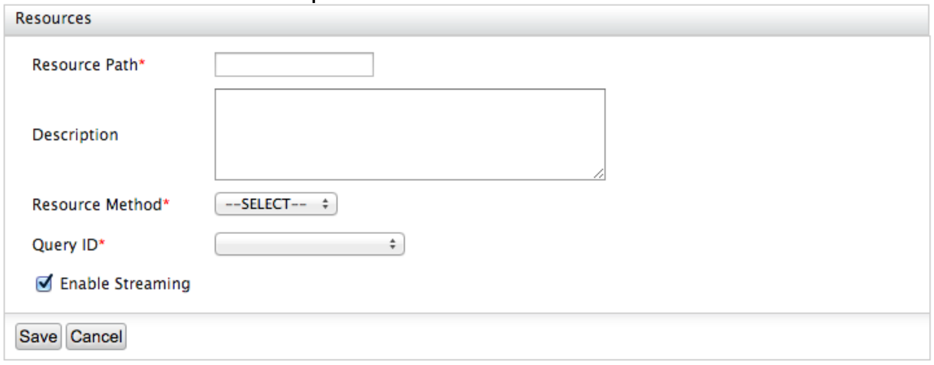
3. Click **List** under **Main** → **Services**. The **RDBMS** data service you created in the previous step will be listed.

4. Click the data service to open the **Service Dashboard** of that data service.

5. Click **Edit Data Service (Wizard)** to open the data service using the **Create Data Service** wizard.

6. Click **Next**, until you get to the **Resources** screen.

7. Click **Add New Resource** to open the **Resources** screen shown below. You can now expose the data as REST resource.

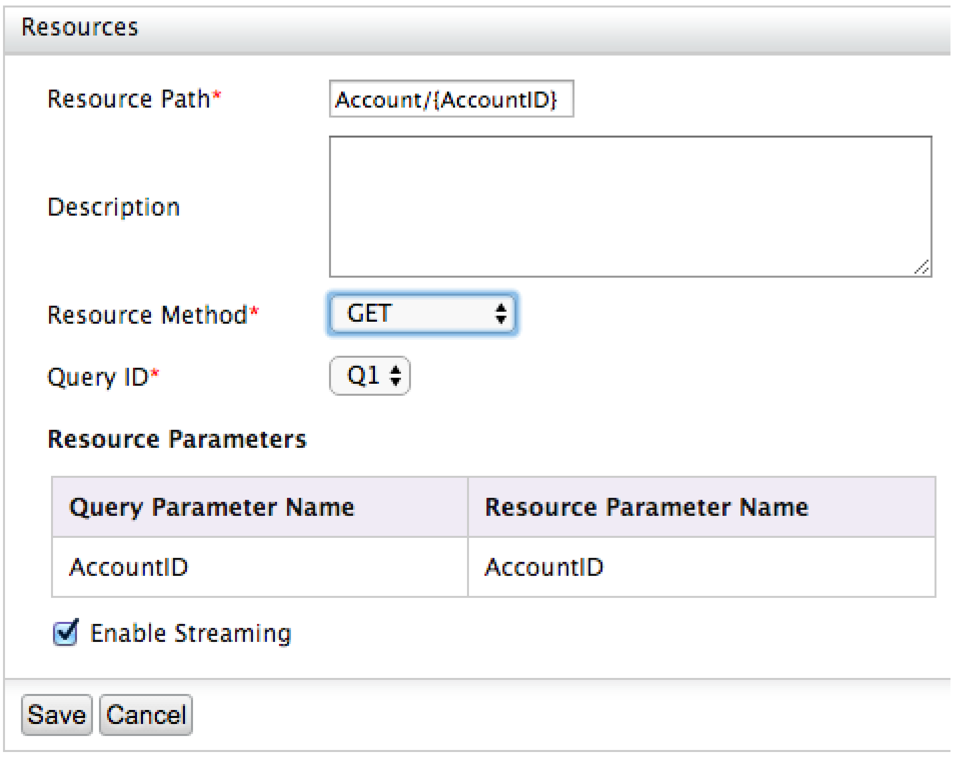


The fields in the above screen are explained below.

- **Resource Path:** The resource name that is appended to the end of the resource URI. There are two ways of giving the resource path:
    1. Use a query path with the following format: <resource_path_name>/{Input_Parameter}
    2. Use a query parameter by giving a name for the resource path. For example, enter Product, if you are querying for products.
- **Resource Method:** The HTTP operation (GET, POST, PUT or DELETE).
- **Query ID:** The corresponding query for the resource invocation.

8. We will proceed to create a resource using the GET method. Enter values as follows:

- Enter Account/{AccountID} in the **Resource Path** field.
- Select **GET** as the **Resource Method**.
- Select the query in the **Query ID** field.


Resources

Resource Path* Account/{AccountID}

Description

Resource Method* GET

Query ID* Q1

Resource Parameters

Query Parameter Name	Resource Parameter Name
AccountID	AccountID

☑ Enable Streaming

9. Save the resource.
10. Save the data service.

- **Invoking the REST resource**

The service can be invoked in REST-style via curl (http://curl.haxx.se). Shown below is the curl command to invoke the GET resource:

```
curl -X GET http://localhost:8280/services/RDBMS.HTTPEndpoint/Account/1
```

Sending message payloads
If you are sending request payloads to a REST resource defined in the data service, the object name specified in the payload must be in the following format: "_<HTTP_METHOD><RESOURCE_PATH>", where RESOURCE _PATH represents the path value specified in the data service resource. However, if the RESOURCE_PATH specified in the data service contains the "/" symbol, be sure to replace the "/" symbol with the underscore symbol ("_") in the payload object name. See the following examples.

If the RESOURCE_PATH is "wso2employee", the payload object name should be as follows:

- For HTTP POST requests: _*postwso2*employee
- For HTTP PUT requests: _*putwso2*employee

If the RESOURCE_PATH is "wso2/employee", the payload object name should be as follows:

- For HTTP POST requests: _*postwso2*_employee
- For HTTP PUT requests: _*putwso2*_employee

If you are sending a batch request, and if the resource path is "wso2/employee", the payload object should be as follows:

- For HTTP POST requests:

```
{
   "_postwso2_employee_batch_req": {
      "_postwso2_employee": [
         .........
      ]
   }
}
```

- For HTTP PUT requests:

```
{
   "_putwso2_employee_batch_req": {
      "_putwso2_employee": [
         .........
      ]
   }
}
```

3.2.5 Invoking Multiple Operations via Request Box

The **request box** feature allows you to invoke multiple operations (consecutively) to a datasource using a single operation. Follow the steps given below to define a data service that can invoke request box operations:

- Setting up a datasource
- Define a data service to invoke request box operations
- Invoking the data service

3.2.6 Invoking an Operation with Multiple Records

The batch requests feature allows you to send multiple (IN-Only) requests to a datasource using a single operation (batch operation). Follow the steps given below to define a data service that can invoke batch requests:

- Setting up a datasource
- Define a data service to insert records in batches
- Invoking the data service

3.2.7 Defining Nested Queries

Nested queries help you to use the result of one query as an input parameter of another, and the queries executed in a nested query works in a transactional manner. Follow the steps given below to add a nested query to a data service.

- Setting up a datasource
- Writing a nested query for a datasource
- Invoking the data service

3.2.8 Receiving Notifications from Data Services

Eventing support is provided by the WS-Eventing Web services standard. When a data service request or response triggers an event, the subscribers listening to those events receive notifications. The criteria for triggering an event as well as the destination to which the event notifications should be sent are defined per data service query. When a certain event-trigger is activated, emails will be sent to all the respective subscribers.

You can create an event trigger from a query as explained below.

- Before you begin
- Enabling notifications for a query in a data service
- Invoking the data service

3.2.9 Handling Distributed Transactions

The data integration feature in the ESB profile of WSO2 EI supports data federation, which means that a single data service can expose data from multiple datasources. However, if you have multiple RDBMSs connected to your data service, and if you need to perform IN-ONLY operations (operations that can insert data and modify data in the datasource) in a coordinated manner, the RDBMSs need to be defined as XA datasources. Consider a scenario where you have two MySQL databases. You can define a single data service for these databases and insert data into both as explained below.

- Setting up distributed MySQL databases
- Adding the datasources to a data service
- Defining queries for the datasources
- Inserting data into the distributed RDBMSs

3.2.10 Exposing Data as an OData Service

In this guide, we will run through the process of exposing and RDBMS as an OData service. When OData is enabled for an datasource, you do not need to manually define CRUD operations. These will be automatically created.

- Setting up an RDBMS
- Expose the RDBMS as an OData service
- Access the data service using CRUD operations

3.2.11 Scheduling Tasks

Task scheduling is used to invoke a data service operation periodically or for a specified number of times. The scheduling functionality is useful when a specific data service operation scheduled for execution is associated with an event-trigger. When such a scheduled task is run, the event can be automatically fired by evaluating the event trigger criteria. For example, we can schedule a task on getProductQuantity operation and set an event to send an email if the quantity goes down to some level.

The following topics are covered:

- Configuring server for task handling
- Adding Scheduled Tasks

4 Deep Dive

An ESB user configures how messages flow through the system, creates proxy services to integrate the back-end services that process messages, and essentially wires together the ESB. This section describes these tasks in the following sections:

4.1 Installation Guide

This section describes how to get started with WSO2 Enterprise Integrator. It contains the following information:

- Installation Prerequisites

Prior to installing WSO2 Enterprise Integrator, it is necessary to have the appropriate prerequisite software installed on your system. Verify that the computer has the supported operating system and development platforms before starting the installation.

System requirements

Table 1 System requirements

Memory	~ 2 GB minimum ~ 1 GB heap size. This is generally sufficient to process typical SOAP messages, but therequirements vary with larger message sizes and the number of messages processed concurrently.
Disk	~ 1 GB, excluding space allocated for log files and databases.

Environment compatibility

- All WSO2 Carbon-based products are Java applications that can be run on any platform that is Oracle JDK 1.8.* compliant.
- All WSO2 Carbon-based products are generally compatible with most common DBMSs. The embedded H2 database is suitable for development, testing, and some production environments. For most

enterprise production environments, however, we recommend you use an industry-standard RDBMS such as Oracle, PostgreSQL, MySQL, MS SQL, etc. For more information, see Working with Databases in the Administration Guide. Additionally, we do not recommend the H2 database as a user store.
- It is **not recommended to use Apache DS** in a production environment due to scalability issues. Instead, use an LDAP like OpenLDAP for user management.
- For environments that WSO2 products are tested with, see Compatibility of WSO2 Products.
- If you have difficulty in setting up any WSO2 product in a specific platform or database,

  - Installing the Product

    Installing WSO2 Enterprise Integrator is fast and easy. Before you begin, be sure you have met the installation prerequisites, and then follow the installation instructions for your platform.

    - Installing on Linux
    - Installing on Windows
    - Installing as a Windows Service

  - Running the Product

    The following sections describe how to run WSO2 Enterprise Integrator (WSO2 EI).

    - Starting the Integration profile
    - Starting the Business Process profile
    - Starting the Message Broker profile
    - Starting the Analytics profile
    - Stopping a profile
    - Working with the Management Console

4.2 Product Administration

WSO2 Enterprise Integrator (WSO2 EI) consists of three profiles: Integration profile, Business Process Management profile and the Message Broker profile. See Introducing the Enterprise Integrator for more information. These profiles are separate runtimes that need to be configured and managed separately. If you are a product administrator, the following topics will provide an overview of the various administration tasks that are common to all profiles of WSO2 EI. Follow the links provided under each section for detailed instructions on each topic.

[Upgrading from a previous release] [Clustering] [Changing the default databases] [Configuring users, roles and permissions] [Configuring security] [Configuring multitenancy] [Configuring the registry] [Performance tuning] [Changing the default ports] [Configuring custom proxy paths] [Customizing error pages] [Customizing the management console] [Monitoring] [Troubleshooting] [Applying patches] [Working with profile-specific administration tasks]

4.3 Integration

The topics in this section provide information on how you can integrate artifacts using WSO2 EI.

- Triggering Messages
- Mediating Messages

- Working with Transports
- Working with Endpoints
- Working with Templates
- Working with Modules
- Working with Message Builders and Formatters
- Working with Registry Artifacts
- Data Integration
- Managing Services
- Calling Admin Services from Apps
- JMS Support
- JSON Support
- REST Support
- WebSocket Support
- Integrating with Other Technologies
- WSO2 EI Tools
- Extending WSO2 EI

4.4 Business Process Management

The topics in this section provide information on how you can deploy, manage, view and execute business processes.

- Writing a Human Task Artifact
- Monitoring a BPMN Process
- Working with BPMN User Substitution
- Managing Business Processes
- Managing Human Tasks
- Managing BPMN
- Endpoint References
- Cleaning-up Business Process Management Artifacts

4.5 Message Brokering

WSO2 Enterprise Integrator (WSO2 EI) is shipped with a separate runtime for message brokering (EI Message Broker profile). This EI Message Broker profile is an instance of the WSO2 Message Broker product; thereby, it contains all the functionality that was available with WSO2 MB. When you use WSO2 EI for integration, you can easily configure the Message Broker runtime along with the Integrator runtime to achieve guaranteed delivery and reliable messaging. Further, you can also run the Message Broker profile in WSO2 EI to facilitate message brokering using Topics and Queues.

- Key concepts of message brokering
- Queues
- Topics
- Subscribers and Publishers
- Role-based permissions
- Metrics monitoring
- Management Console
- The slot-based architecture of the EI Message Broker
- Communicating with publishers
- Assigning slots
- Deleting a slot
- Reassigning slot when the last subscriber leaves
- Delivering messages to subscribers

4.6 WSO2 Enterprise Integrator Tooling

This section describes how you can use the tooling support provided via WSO2 EI tooling to create and manage artifacts.

See the following topics for detailed information on how to install WSO2 EI tooling, how to create artifacts via tooling, and how to package created artifacts into archives that you can deploy them.

- Installing Enterprise Integrator Tooling
- Running WSO2 Enterprise Integrator via Tooling
- Working with Enterprise Integrator Artifacts
- Packaging your Artifacts into Composite Applications
- Importing Existing Projects into Workspace

4.7 WSO2 Enterprise Integrator Analytics

This section describes how you can use Enterprise Integrator analytics to publish information related to the processing carried out by WSO2 Enterprise Integrator to the Analytics Dashboard.

See the following topics for detailed information on how to work with Enterprise Integrator analytics.

- Prerequisites to Publish Statistics
- Analyzing WSO2 EI via the Analytics Dashboard
- Extending EI Analytics
- Purging Analytics Data

5 Conclusion

WSO2 Enterprise Integrator has all the components required for an integration layer, packaged in a single distribution. You can run each profile and configure it easily to suite your requirements using the graphical user interface. To summarize, the ESB profile provides mediation and data services capabilities, the message broker profile can be used for handling queues, the business profile is used for workflow support, the analytics profile is used to monitor the integration layer and the MSF4J profile is used to host microservices.

6 References

- SMART-FI Proposal 2016
- SMART-FI Project Consortium Agreement
- Project website: http://www.smart-fi.eu/
- WSO2 website http://wso2.com