WSO2 API Manager 4.1.0 Fundamentals - Integration Profile

Mediating Messages

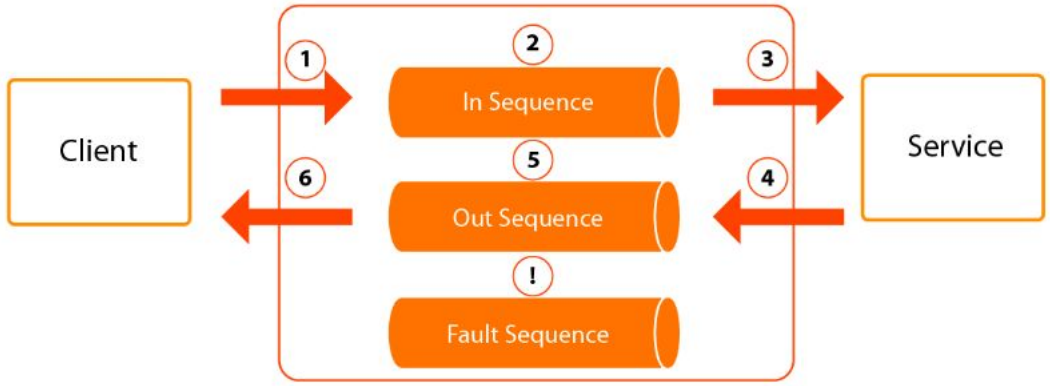
WSO2 Training

Module Objectives

At the end of this module, attendees will be able to:

- Understand message flows in WSO2 Micro Integrator.
- Understand the main integration use cases you can build using WSO2 Integration Studio and WSO2 Micro Integrator.
  - Message Routing
  - Message Transformation
  - Service Orchestration
  - Asynchronous Messaging
  - Protocol Switching
- Understand how to reuse integration solutions using templates.

High-Level Message Flow



At a high-level, WSO2 Micro Integrator is the middleware between a service and a client requesting the service.

- Messages navigate between client and service through sequences built in WSO2 Micro Integrator.
- Three main sequences exist: In Sequence, Out Sequence, and Fault Sequence.

Message Flow:

1. A client sends a request to a service.
2. Message flows thru the InSequence.
3. Message is delivered to the service.
4. Service generates a response.
5. Response is sent through the OutSequence.
6. Message is delivered to the client.

The FaultSequence can be used (if necessary) for error handling.

By default, the fault sequence logs the message, the payload, and any errors/exceptions encountered, and the drop mediator stops further processing. You should configure the fault sequence with the correct error handling logic instead of simply dropping messages.

Basic Mediators

Name	Description
Log Mediator	Logs full or part of the message at various severity levels (Trace, Debug, etc.).
Sequence Mediator	Invokes existing sequence. Sequence name can be static or dynamic.
Call Mediator	Sends a message out using static information or an endpoint definition.
Respond Mediator	Responds back to the client.
Switch Mediator	Evaluates message content against regular expressions and invokes the corresponding mediator (switch-case-default).
Property Mediator	Assigns a value to a property to be retrieved later
Drop Mediator	Stops processing the current message.
Fault Mediator	Transforms current message into a custom Fault message.

Message Routing

- Use the **Switch Mediator**.
- Allows content-based routing.
- Evaluates the XPath or JSONPath and returns a string (this string is evaluated against a regular expression in each case statement).
- If a matching case is found, others are not evaluated. If no match is found, a default will be executed.

Let's try it out!

Message Routing

Message Transformation

The follow mediators can be used for message transformation:

- Data Mapper Mediator
- PayLoad Factory Mediator
- XSLT Mediator
- FastXSLT Mediator
- Smooks Mediator
- XQuery Mediator

Message Transformation

The Data Mapper mediator can be used to convert and transform data visually. It can be used to:

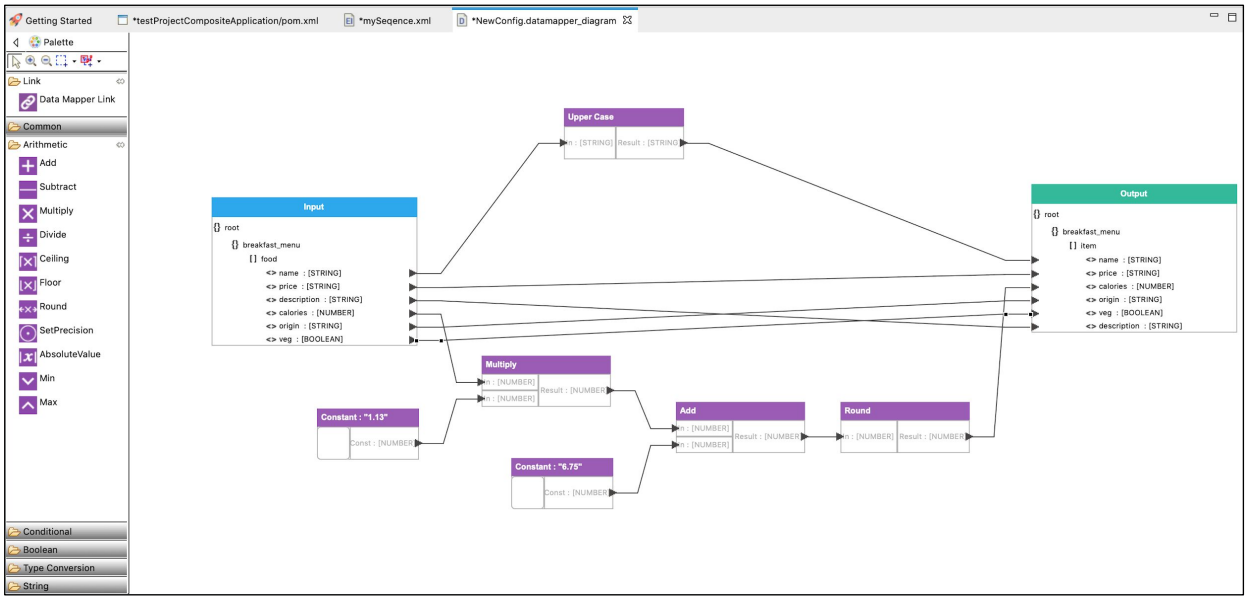
- Transform from/to JSON, XML, and CSV.
- Easily integrate with mediation sequences.
- Apply operations.

Find out more about the Data Mapper.

The Data Mapper mediator consists of three main configuration files:
- Input schema
- Output schema
- Mapping configuration

The mapping configuration is generated using WSO2 Integration Studio..

https://apim.docs.wso2.com/en/4.0.0/reference/mediators/data-mapper-mediator/



The Operations palette of the data mapper mediator includes common, arithmetic, conditional, and other operations.

Message Transformation

With the **payloadFactory** mediator, you can:

- Transform or replace the content of a message.
- Configure the format of a request or response and map it to the arguments.

Find out more about the payloadFactory mediator.

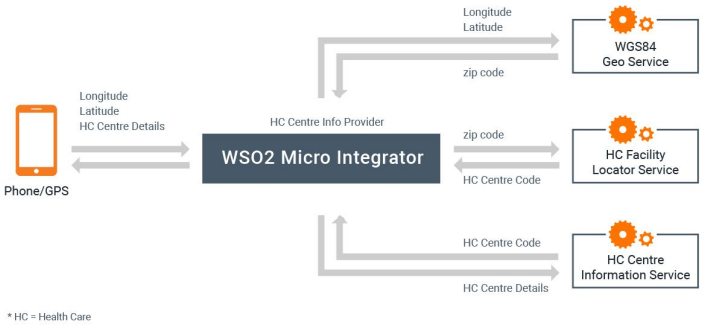
https://apim.docs.wso2.com/en/4.1.0/reference/mediators/payloadfactory-mediator/

Let's try it out!

Message Transformation

Service Orchestration (Service Chaining)

- A collection of mediators allows multiple services to be exposed as a single service.
- Orchestrates responses from one service and feeds them as inputs to other services.



Let’s try it out!

Service Orchestration

Templates

Prototypes of Endpoints or Sequences that can be used as a base for new objects:

- Endpoint templates
- Sequence templates

Let’s try it out!

Reusing Mediation Sequences

Asynchronous Messaging

The **store and forward** mechanism is used for guaranteed delivery using the message store and message processor in the Micro Integrator.

Asynchronous Messaging

Message Stores

- Temporarily stores messages before they are finally delivered to their destination by a message processor.
- Uses the **Store** mediator to create In Memory (default), RabbitMQ, JMS, or a custom message store (i.e. for guaranteed delivery).

Message Processors

- Deliver messages that have been temporarily stored in a message store.

The **store and forward** mechanism is used for guaranteed delivery using the message store and message processor in the Micro Integrator.

Let’s try it out!

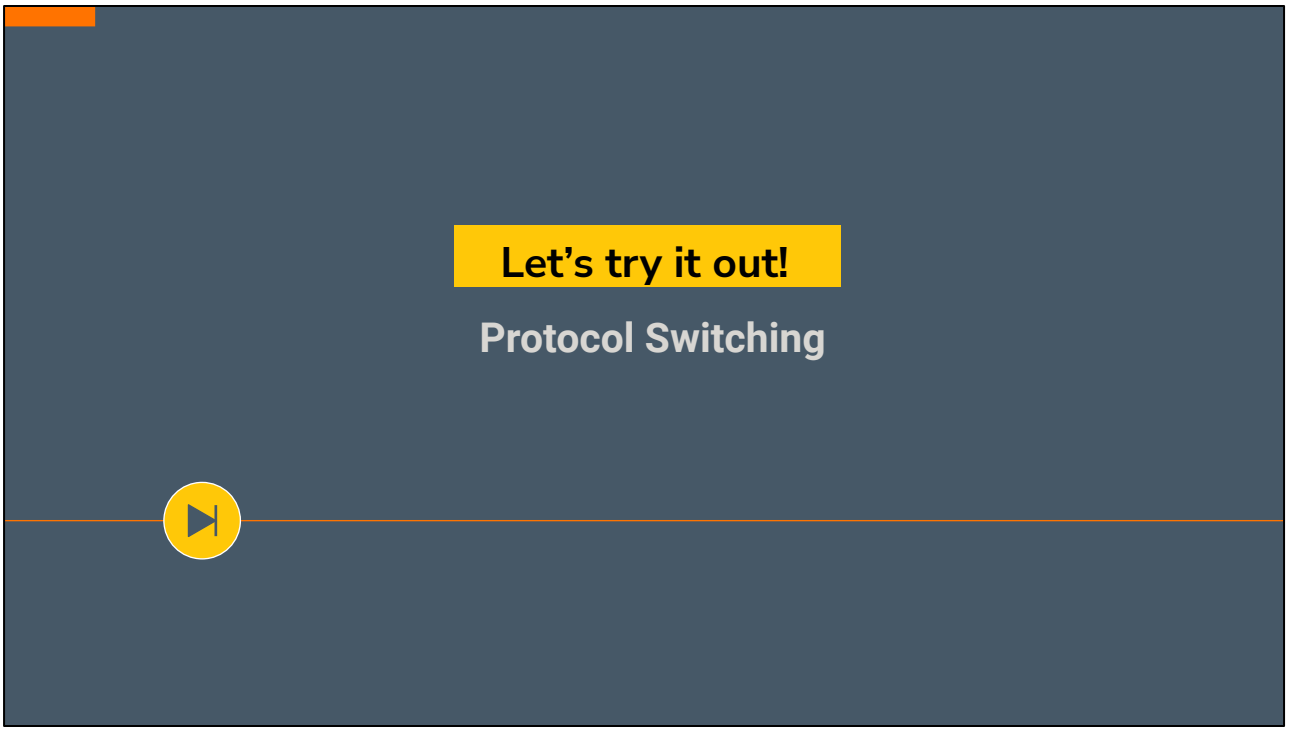
Asynchronous Messaging

Protocol Switching

There are two types of use cases:

- Translate messages between protocols.
  - **Example:** An HTTP message can be translated to a JMS message.

- Translate messages between common formats.
  - **Example:** If a client sends a JSON request when the existing service uses XML SOAP, a message translator is required in the request as well as in the response.


Let’s try it out!
Protocol Switching