Unit 5 – Modeling Processes

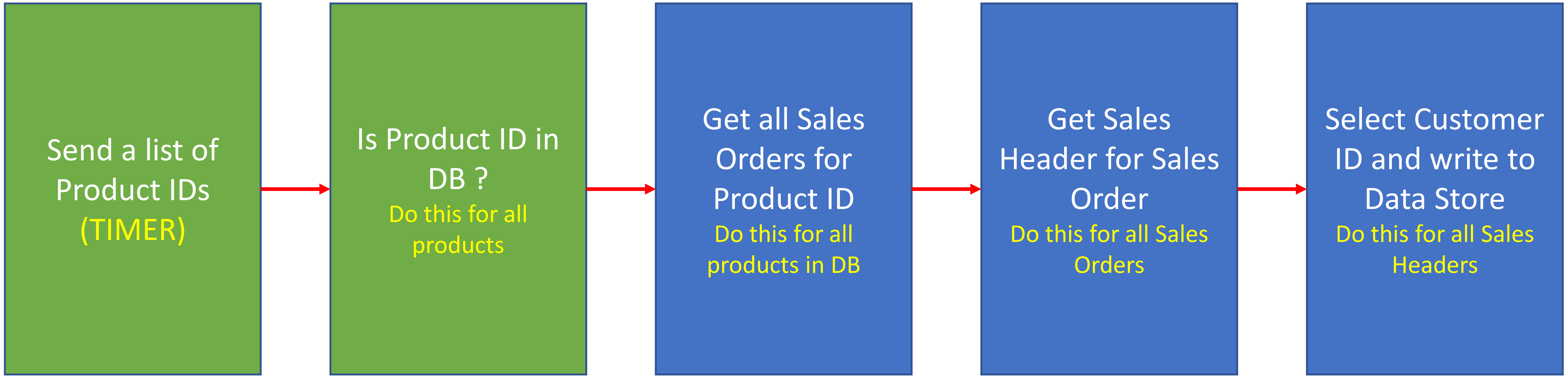
Developing with SAP Integration Suite

C_CPI_15

Agenda

- Modeling Integration Flows
- Learning the Basics
- Using Adapters
- Using Mappings
- Using Adapter Outbound Security
- Performing Exception Handling
- Using Scripting
- Using Adapter Inbound Security
- Using Integration Patterns

Business Scenario - Task flow


Send a list of Product IDs
(TIMER)
Is Product ID in DB ?
Do this for all products
Get all Sales Orders for Product ID
Do this for all products in DB
Get Sales Header for Sales Order
Do this for all Sales Orders
Select Customer ID and write to Data Store
Do this for all Sales Headers

Business Scenario – Steps involved

1. Create Integration Package and Integration Flow with Timer
2. Add Content Modifier (mock data – Product list)
3. Add General Splitter (split Products for iteration)
4. Add Content Modifier (add Product ID to Exchange Property)
5. Add Request Reply (is Product ID in database)
6. Add Router (If Product ID exists - CONTINUE, else END)

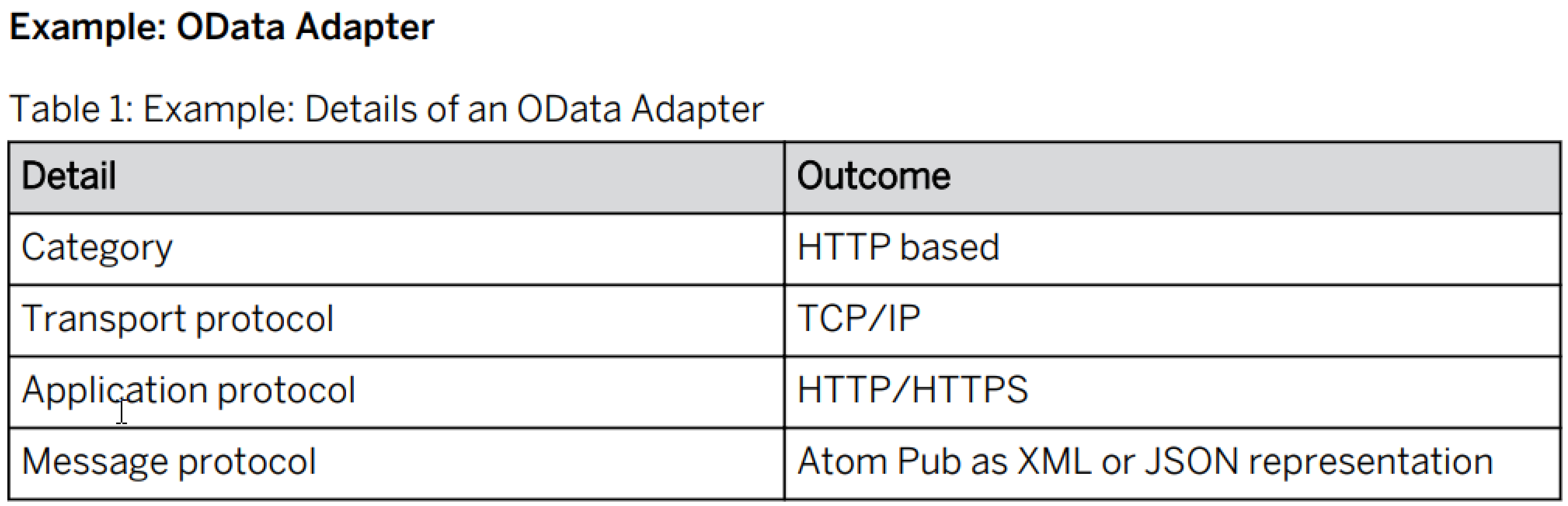
Using Adapters

- Wide variety of pre-built adapters available
- Support various application, transport protocols, message protocols
- Differentiation made between input and output adapters
- Broadly categorized into 2 groups
  - TCP based
  - Non TCP based

Features of OData Adapter

- Query wizard
  - Navigate the interface to be accessed with metadata document
- Page Processing mode
  - Read entries in multiple pages which are processed sequentially
- Automatically removing namespaces
  - Remove namespaces and prefixes automatically

Details of OData Adapter

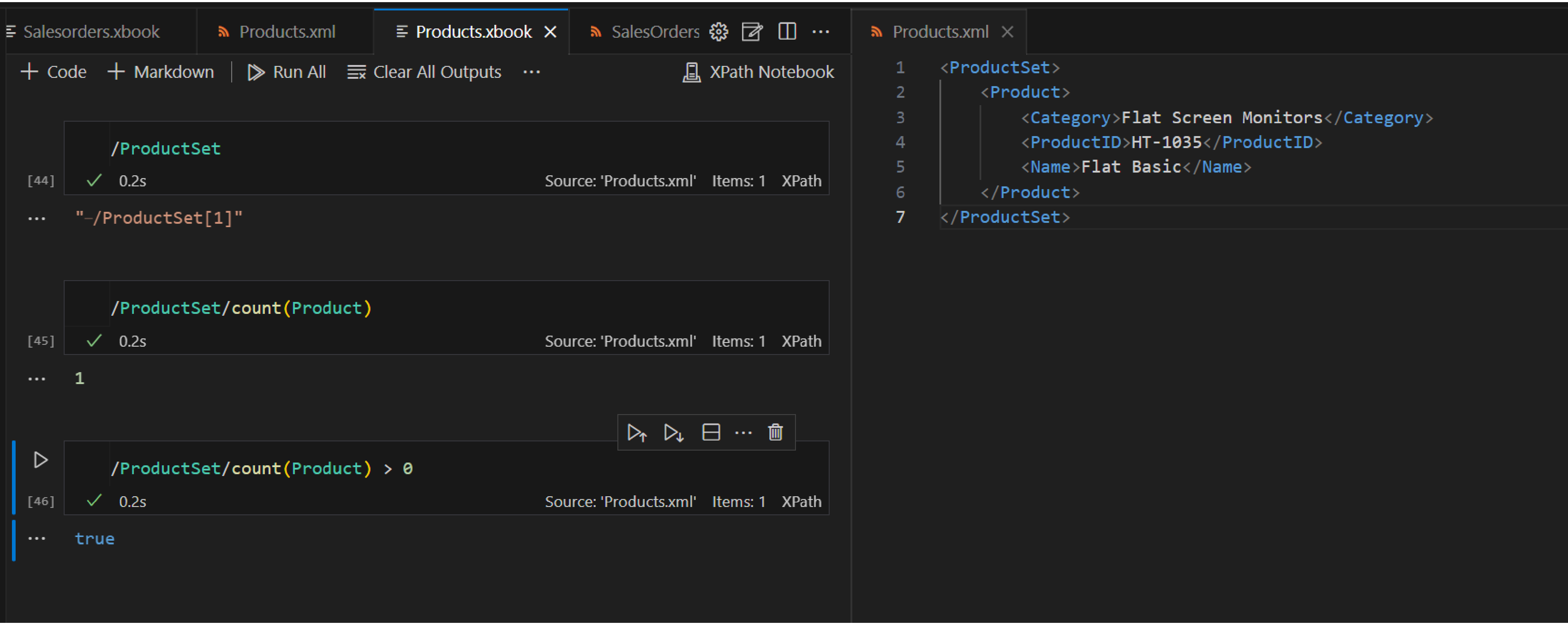


Example: OData Adapter

Table 1: Example: Details of an OData Adapter

Detail	Outcome
Category	HTTP based
Transport protocol	TCP/IP
Application protocol	HTTP/HTTPS
Message protocol	Atom Pub as XML or JSON representation

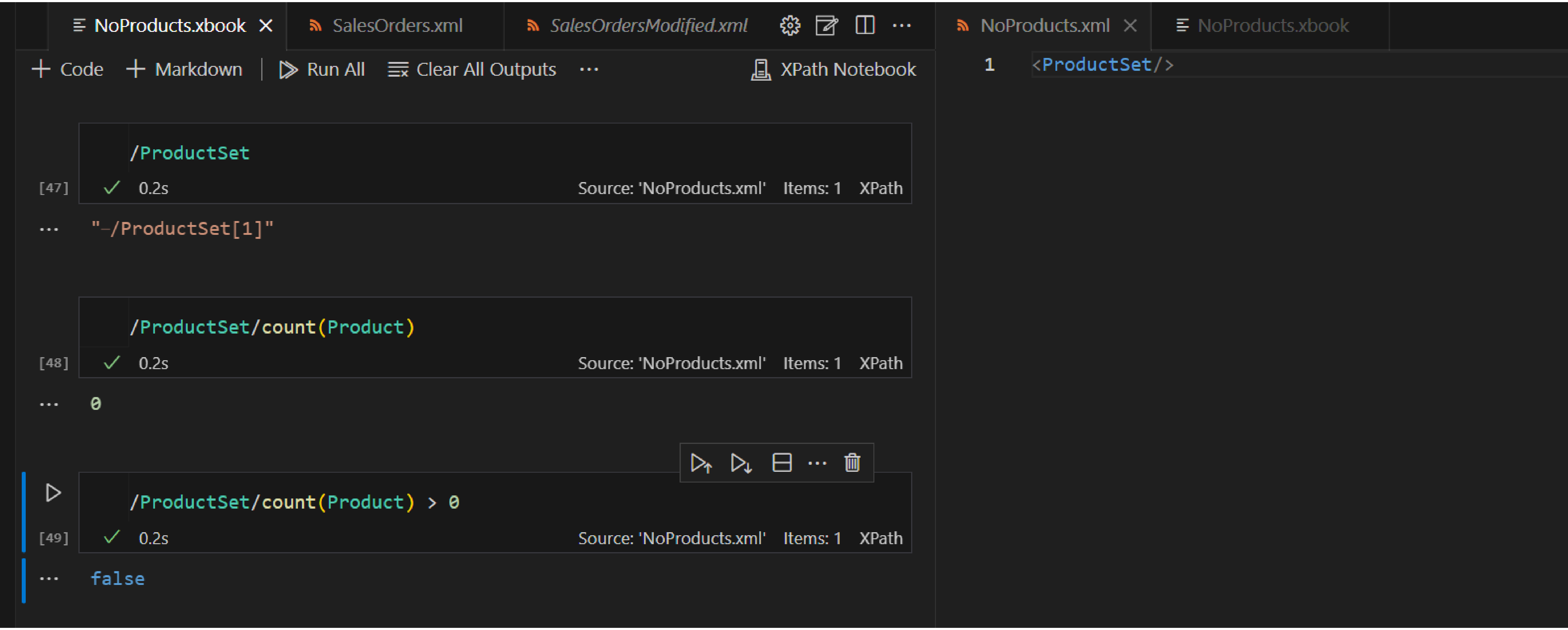
XPath Expressions



```
/ProductSet
[44] ✓ 0.2s                Source: 'Products.xml'  Items: 1  XPath
"~/ProductSet[1]"

/ProductSet/count(Product)
[45] ✓ 0.2s                Source: 'Products.xml'  Items: 1  XPath
1

/ProductSet/count(Product) > 0
[46] ✓ 0.2s                Source: 'Products.xml'  Items: 1  XPath
true
```

```xml
<ProductSet>
    <Product>
        <Category>Flat Screen Monitors</Category>
        <ProductID>HT-1035</ProductID>
        <Name>Flat Basic</Name>
    </Product>
</ProductSet>
```

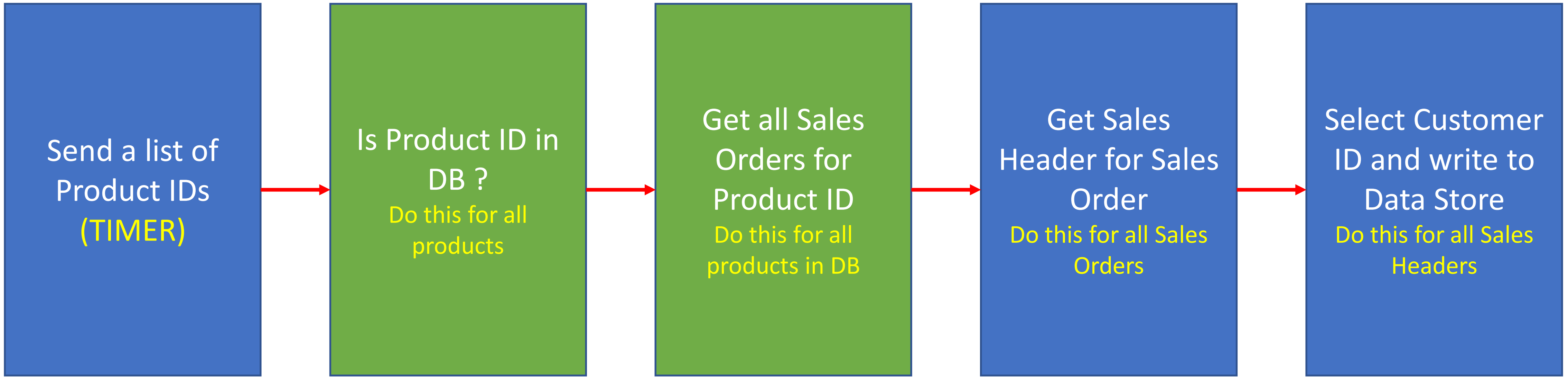
XPath Expressions



```
/ProductSet
[47] ✓ 0.2s    Source: 'NoProducts.xml'   Items: 1   XPath
"~/ProductSet[1]"

/ProductSet/count(Product)
[48] ✓ 0.2s    Source: 'NoProducts.xml'   Items: 1   XPath
0

/ProductSet/count(Product) > 0
[49] ✓ 0.2s    Source: 'NoProducts.xml'   Items: 1   XPath
false
```

```
<ProductSet/>
```

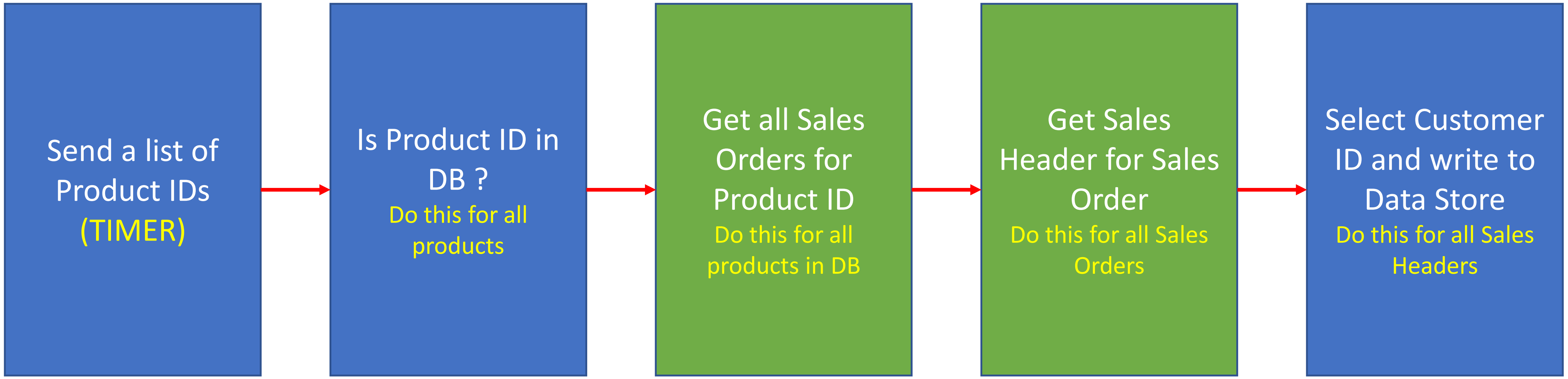
Business Scenario - Task flow


Send a list of Product IDs (TIMER)
Is Product ID in DB ? Do this for all products
Get all Sales Orders for Product ID Do this for all products in DB
Get Sales Header for Sales Order Do this for all Sales Orders
Select Customer ID and write to Data Store Do this for all Sales Headers

Business Scenario – Steps involved

7. Add Request Reply (get all Sales Orders for each Product)
8. Add XSLT Mapping (remove namespaces)
9. Add General Splitter (split Sales Order for iteration)
10. Add Content Modifier (add Sales Order ID, Item to Exchange Property)

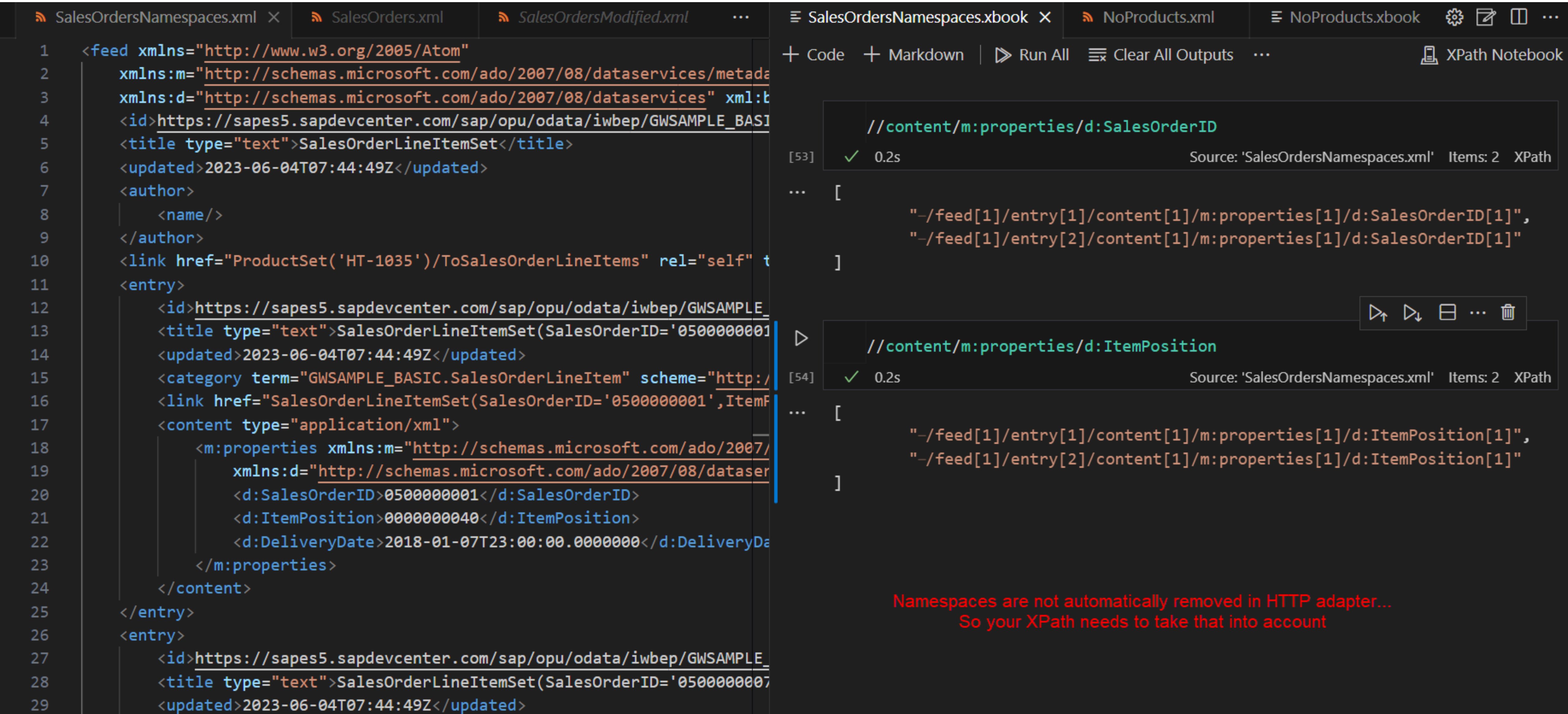
Business Scenario - Task flow


Send a list of Product IDs (TIMER)
Is Product ID in DB ? Do this for all products
Get all Sales Orders for Product ID Do this for all products in DB
Get Sales Header for Sales Order Do this for all Sales Orders
Select Customer ID and write to Data Store Do this for all Sales Headers

Business Scenario – Steps involved

11. Add Request Reply (get Sales Header for each Sales Order)
12. Add XSLT Mapping (remove namespaces)
13. Add Content Modifier (add Customer ID to Exchange Property)

HTTP Adapter



SalesOrdersNamespaces.xml | SalesOrders.xml | SalesOrdersModified.xml

```xml
<feed xmlns="http://www.w3.org/2005/Atom"
    xmlns:m="http://schemas.microsoft.com/ado/2007/08/dataservices/metada
    xmlns:d="http://schemas.microsoft.com/ado/2007/08/dataservices" xml:b
    <id>https://sapes5.sapdevcenter.com/sap/opu/odata/iwbep/GWSAMPLE_BASI
    <title type="text">SalesOrderLineItemSet</title>
    <updated>2023-06-04T07:44:49Z</updated>
    <author>
        <name/>
    </author>
    <link href="ProductSet('HT-1035')/ToSalesOrderLineItems" rel="self" t
    <entry>
        <id>https://sapes5.sapdevcenter.com/sap/opu/odata/iwbep/GWSAMPLE_
        <title type="text">SalesOrderLineItemSet(SalesOrderID='0500000001
        <updated>2023-06-04T07:44:49Z</updated>
        <category term="GWSAMPLE_BASIC.SalesOrderLineItem" scheme="http:/
        <link href="SalesOrderLineItemSet(SalesOrderID='0500000001',ItemP
        <content type="application/xml">
            <m:properties xmlns:m="http://schemas.microsoft.com/ado/2007/
                xmlns:d="http://schemas.microsoft.com/ado/2007/08/dataser
                <d:SalesOrderID>0500000001</d:SalesOrderID>
                <d:ItemPosition>0000000040</d:ItemPosition>
                <d:DeliveryDate>2018-01-07T23:00:00.0000000</d:DeliveryDa
            </m:properties>
        </content>
    </entry>
    <entry>
        <id>https://sapes5.sapdevcenter.com/sap/opu/odata/iwbep/GWSAMPLE_
        <title type="text">SalesOrderLineItemSet(SalesOrderID='0500000007
        <updated>2023-06-04T07:44:49Z</updated>
```

SalesOrdersNamespaces.xbook | NoProducts.xml | NoProducts.xbook

+ Code + Markdown | Run All | Clear All Outputs ... XPath Notebook

```
//content/m:properties/d:SalesOrderID
[53] ✓ 0.2s                Source: 'SalesOrdersNamespaces.xml'  Items: 2  XPath
[
        "~/feed[1]/entry[1]/content[1]/m:properties[1]/d:SalesOrderID[1]",
        "~/feed[1]/entry[2]/content[1]/m:properties[1]/d:SalesOrderID[1]"
]
```

```
//content/m:properties/d:ItemPosition
[54] ✓ 0.2s                Source: 'SalesOrdersNamespaces.xml'  Items: 2  XPath
[
        "~/feed[1]/entry[1]/content[1]/m:properties[1]/d:ItemPosition[1]",
        "~/feed[1]/entry[2]/content[1]/m:properties[1]/d:ItemPosition[1]"
]
```

Namespaces are not automatically removed in HTTP adapter…
So your XPath needs to take that into account

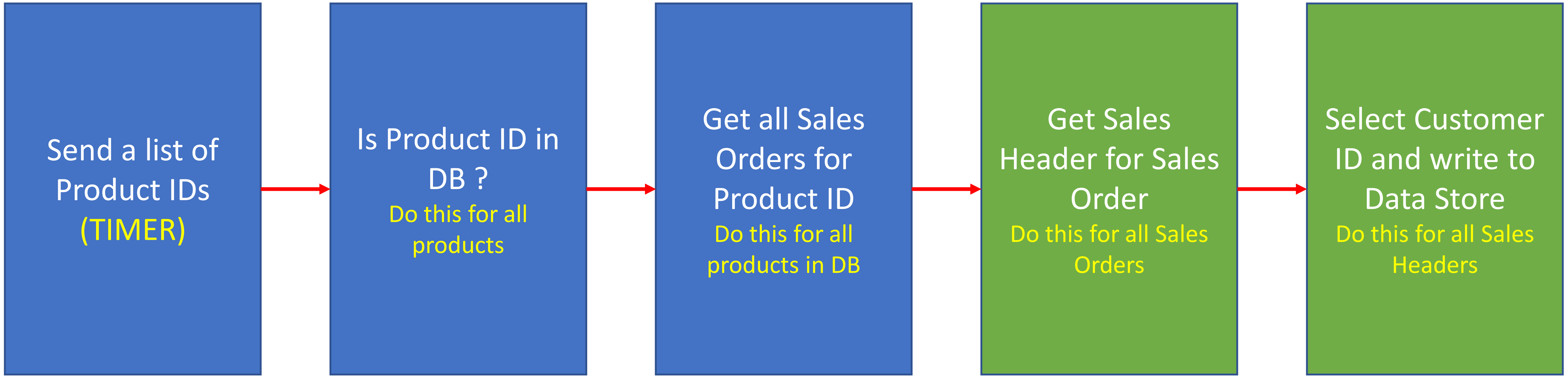
Mappings

- Message Mapping
  - Mapping editor provides tools to map XML or JSON messages
- XSLT Mapping
  - Language designed for transforming XML docs to other formats
  - Stylesheet is processed by an XSLT processor (Xalon or Saxon)
- Mapping with scripting
- Operation Mapping from Enterprise Service Repository (On-Premise)

Mappings

- Process of converting source format into different target formats
- Message Mapping offers context handling, UDF, testing functions
- XSLT Mapping requires XML as input
  - Can create more target formats
  - Useful for creating attachments
- Mapping via scripting offers most flexibility

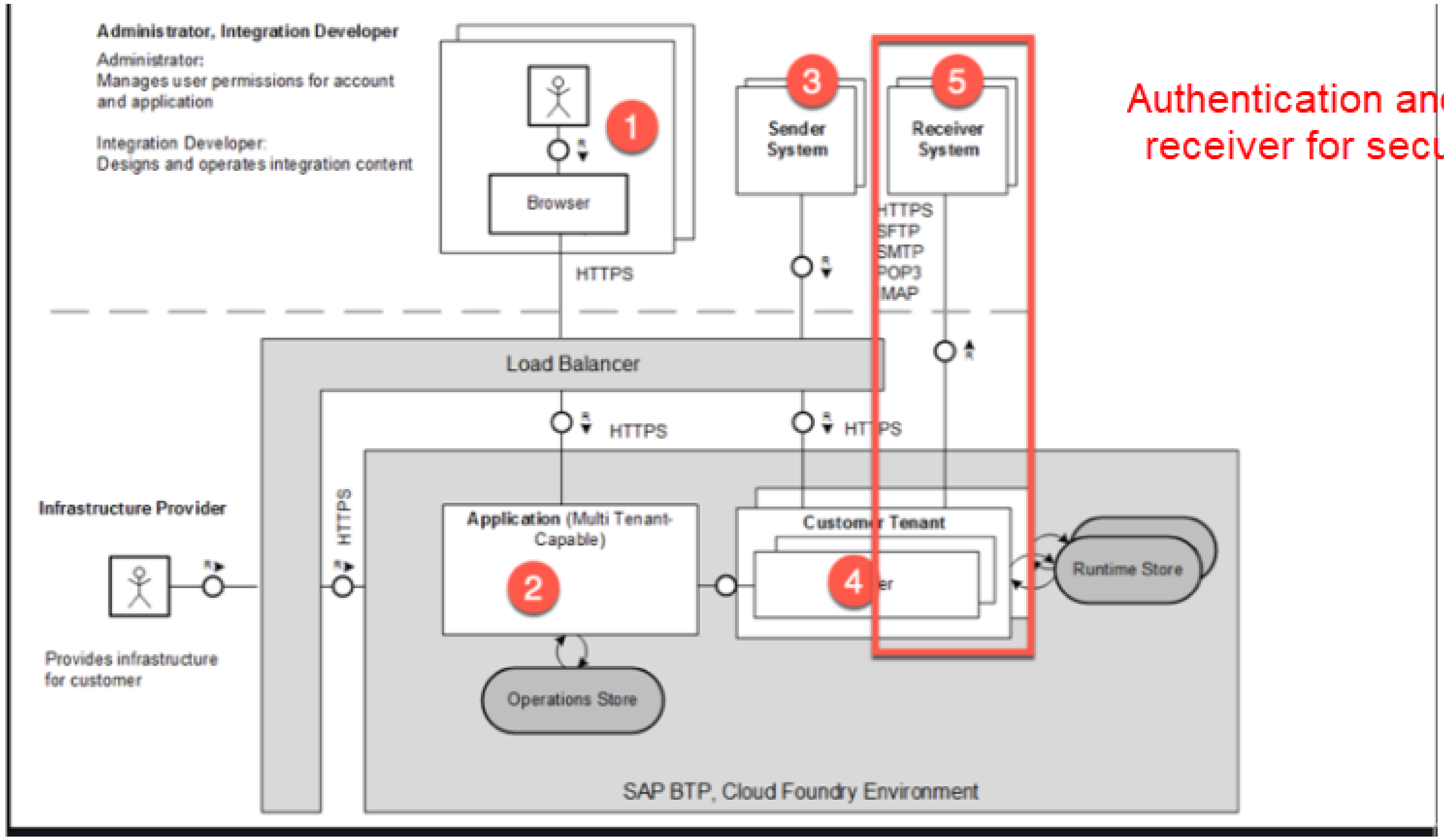
Business Scenario - Task flow


Send a list of Product IDs (TIMER)
Is Product ID in DB ? Do this for all products
Get all Sales Orders for Product ID Do this for all products in DB
Get Sales Header for Sales Order Do this for all Sales Orders
Select Customer ID and write to Data Store Do this for all Sales Headers

Business Scenario – Steps involved

14. Add Data Store (write Customer ID to Data Store)

Using Adapter Outbound Security


Administrator, Integration Developer
Administrator:
Manages user permissions for account and application
Integration Developer:
Designs and operates integration content
1
Browser
HTTPS
3
Sender System
5
Receiver System
HTTPS
SFTP
SMTP
POP3
IMAP
Load Balancer
HTTPS
HTTPS
HTTPS
Infrastructure Provider
Provides infrastructure for customer
Application (Multi Tenant-Capable)
2
Customer Tenant
4
Runtime Store
Operations Store
SAP BTP, Cloud Foundry Environment

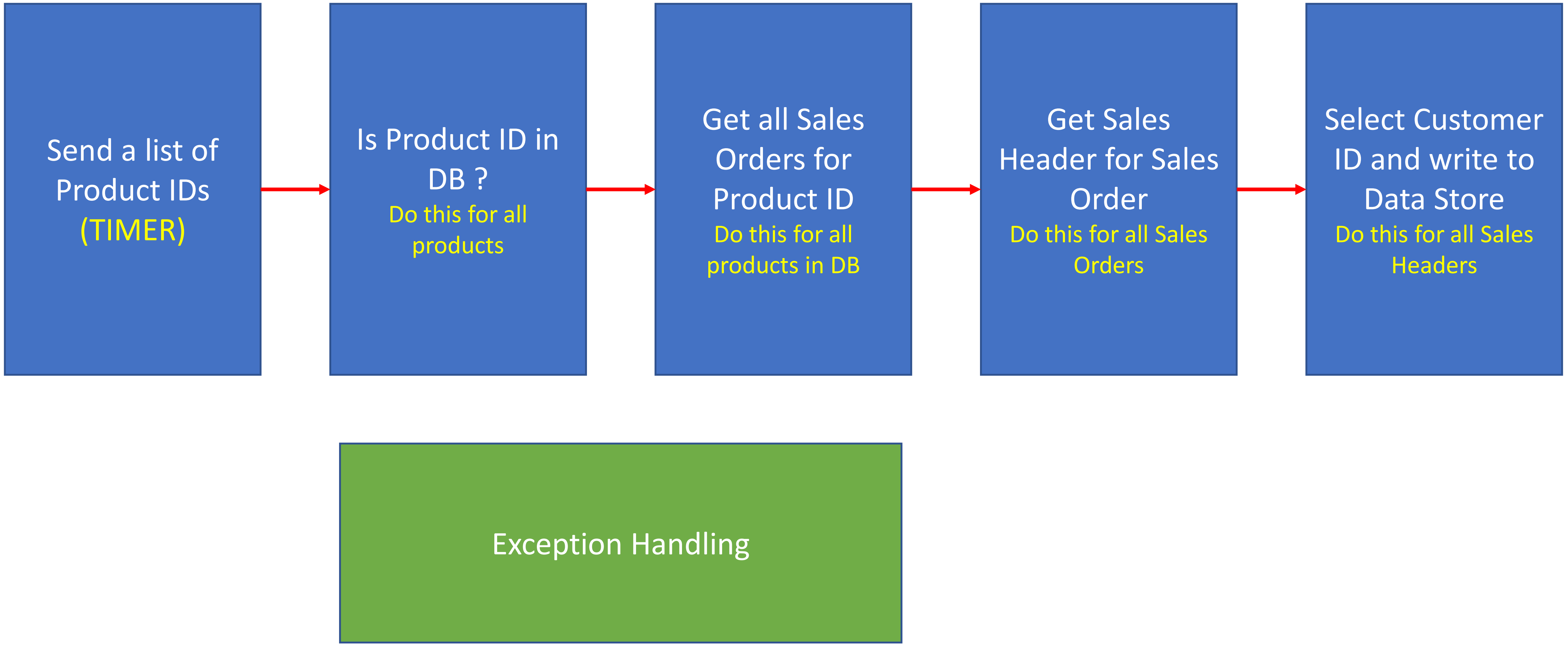
Authentication and Authorization with receiver for secure communication

Example of a Direct Receiver and Sender Adapter

Options for Authentication / Authorization

- Basic
- Client Certificate
- None
- OAuth2 Client Credentials
- OAuth2 SAML Bearer Assertion

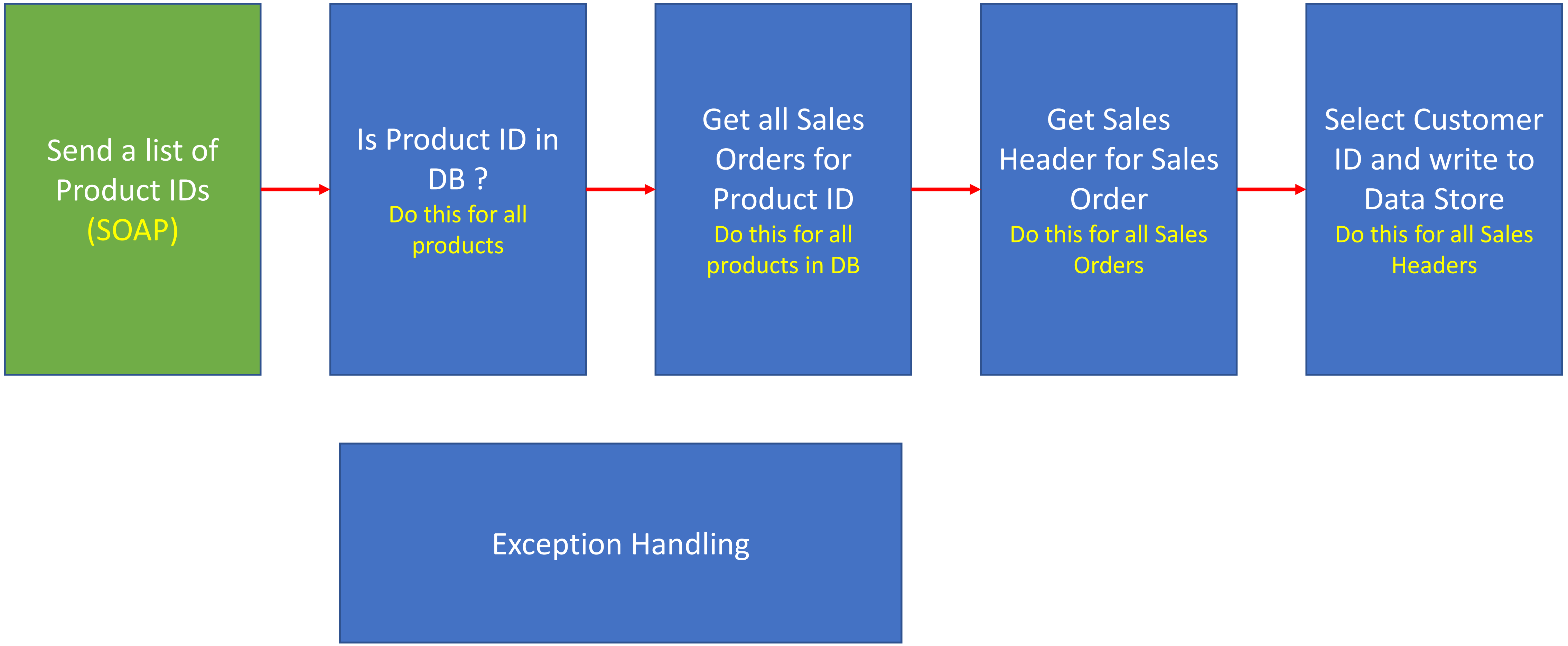
Business Scenario - Task flow


Send a list of Product IDs
(TIMER)
Is Product ID in DB ?
Do this for all products
Get all Sales Orders for Product ID
Do this for all products in DB
Get Sales Header for Sales Order
Do this for all Sales Orders
Select Customer ID and write to Data Store
Do this for all Sales Headers
Exception Handling

Business Scenario – Steps involved

15. Add Exception Subprocess (barebone)
16. Add Groovy Script (read exception messages to payload)

Business Scenario - Task flow

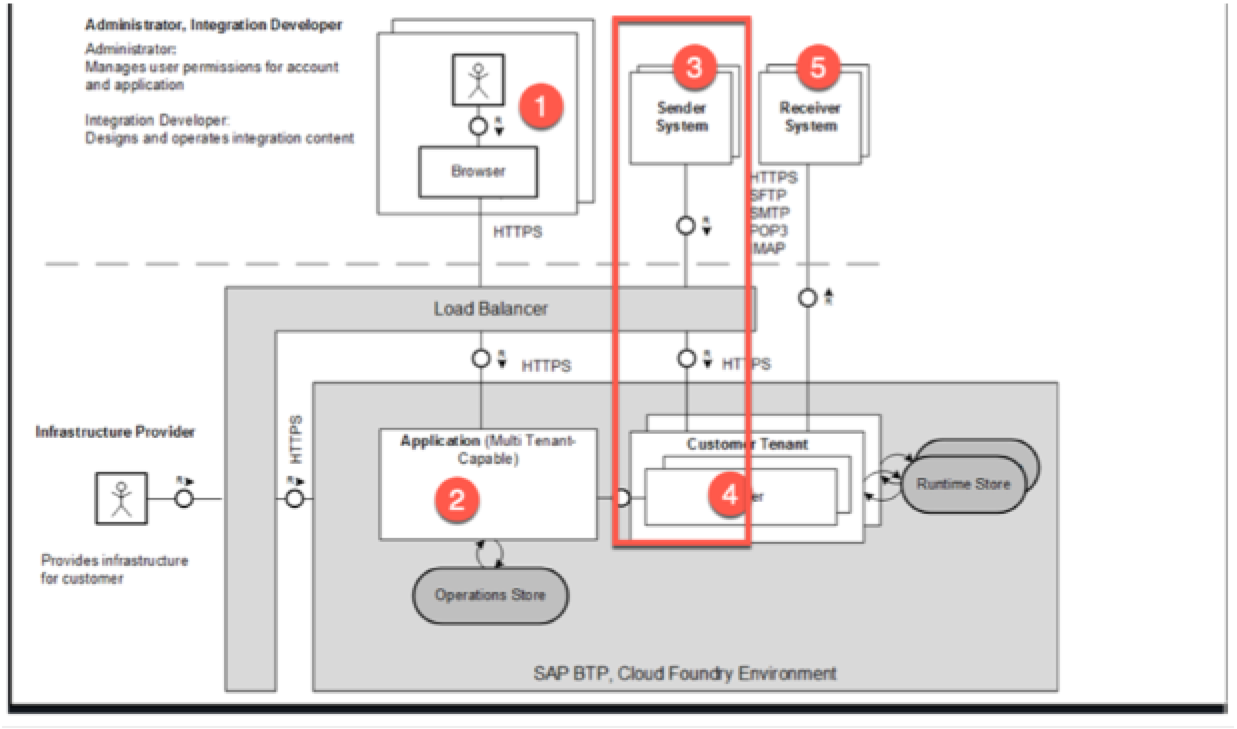

Send a list of Product IDs (SOAP)
Is Product ID in DB ?
Do this for all products
Get all Sales Orders for Product ID
Do this for all products in DB
Get Sales Header for Sales Order
Do this for all Sales Orders
Select Customer ID and write to Data Store
Do this for all Sales Headers
Exception Handling

Business Scenario – Steps involved

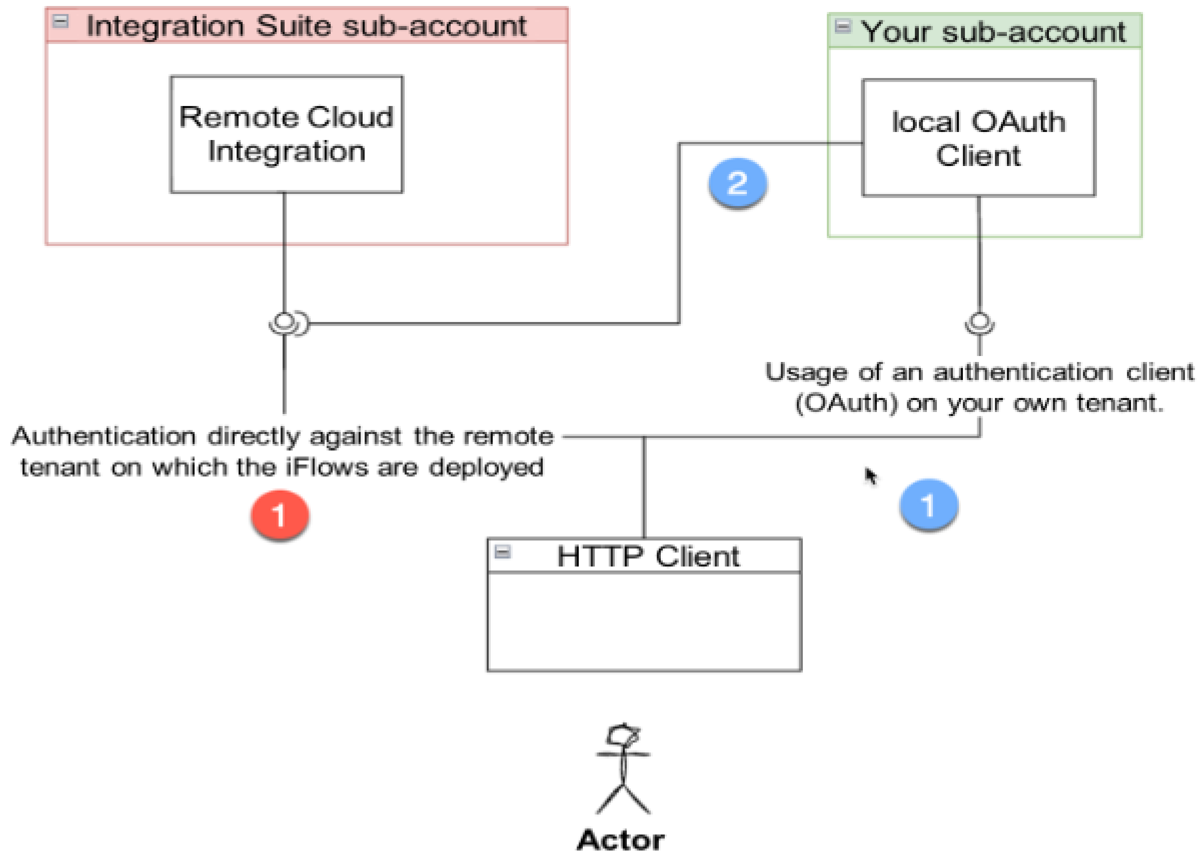
17. Add Message Start (remove Timer)
18. Add SOAP adapter (asynchronous call)
19. Remove $top clause (power of asynchronous call)

Using Adapter Inbound Security

- Certificates between sender and load balancer for HTTPS connection
- Sender’s authorization validated against Integration flow endpoint


Administrator, Integration Developer
Administrator:
Manages user permissions for account and application
Integration Developer:
Designs and operates integration content
1
Browser
HTTPS
3
Sender System
5
Receiver System
HTTPS
SFTP
SMTP
POP3
IMAP
Load Balancer
HTTPS
HTTPS
HTTPS
Infrastructure Provider
Provides infrastructure for customer
Application (Multi Tenant-Capable)
2
Customer Tenant
4
Runtime Store
Operations Store
SAP BTP, Cloud Foundry Environment

Authorization of sender


Integration Suite sub-account
Remote Cloud Integration
Your sub-account
local OAuth Client
2
Usage of an authentication client (OAuth) on your own tenant.
Authentication directly against the remote tenant on which the iFlows are deployed
1
1
HTTP Client
Actor

Sender Authorization

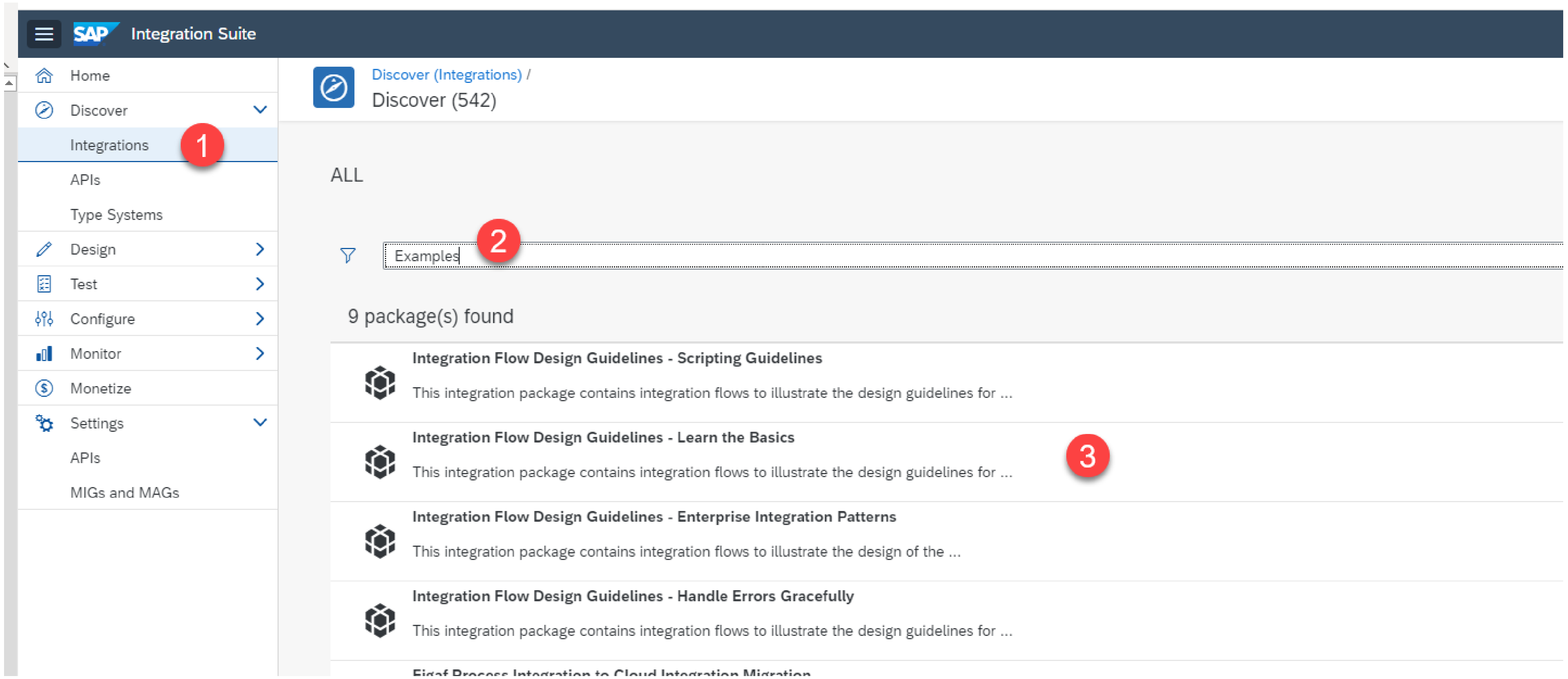
Authentication against remote endpoint

- Assign user role ESBMessaging.send
- Not recommended for production use

Authentication (Oauth) client on your own Tenant

- Set up Process Integration Runtime instance
- Supports
    - Authorization code
    - Client credentials
    - Password
    - Refresh Token
    - SAML2 Bearer
    - JWT Bearer

Sample Integration Flows



9 package(s) found

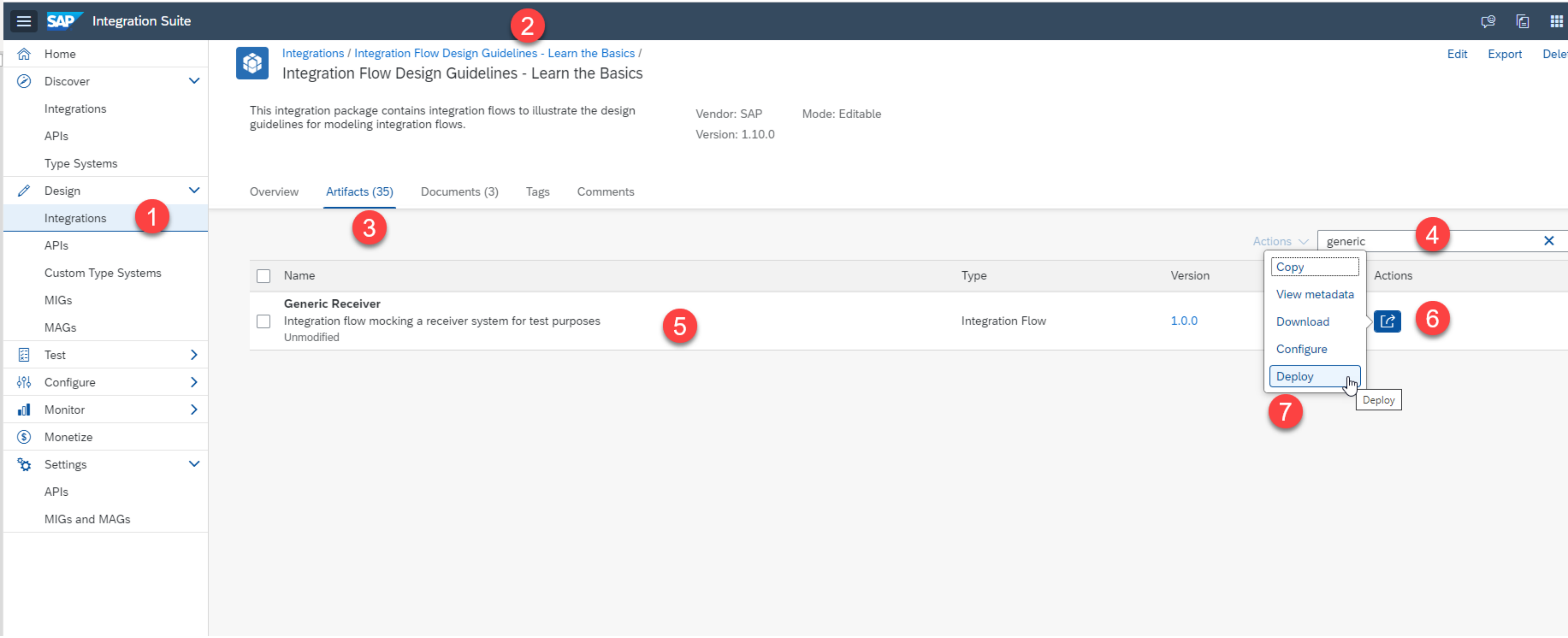
Integration Flow Design Guidelines - Scripting Guidelines
This integration package contains integration flows to illustrate the design guidelines for ...

Integration Flow Design Guidelines - Learn the Basics
This integration package contains integration flows to illustrate the design guidelines for ...

Integration Flow Design Guidelines - Enterprise Integration Patterns
This integration package contains integration flows to illustrate the design of the ...

Integration Flow Design Guidelines - Handle Errors Gracefully
This integration package contains integration flows to illustrate the design guidelines for ...

Figaf Process Integration to Cloud Integration Migration



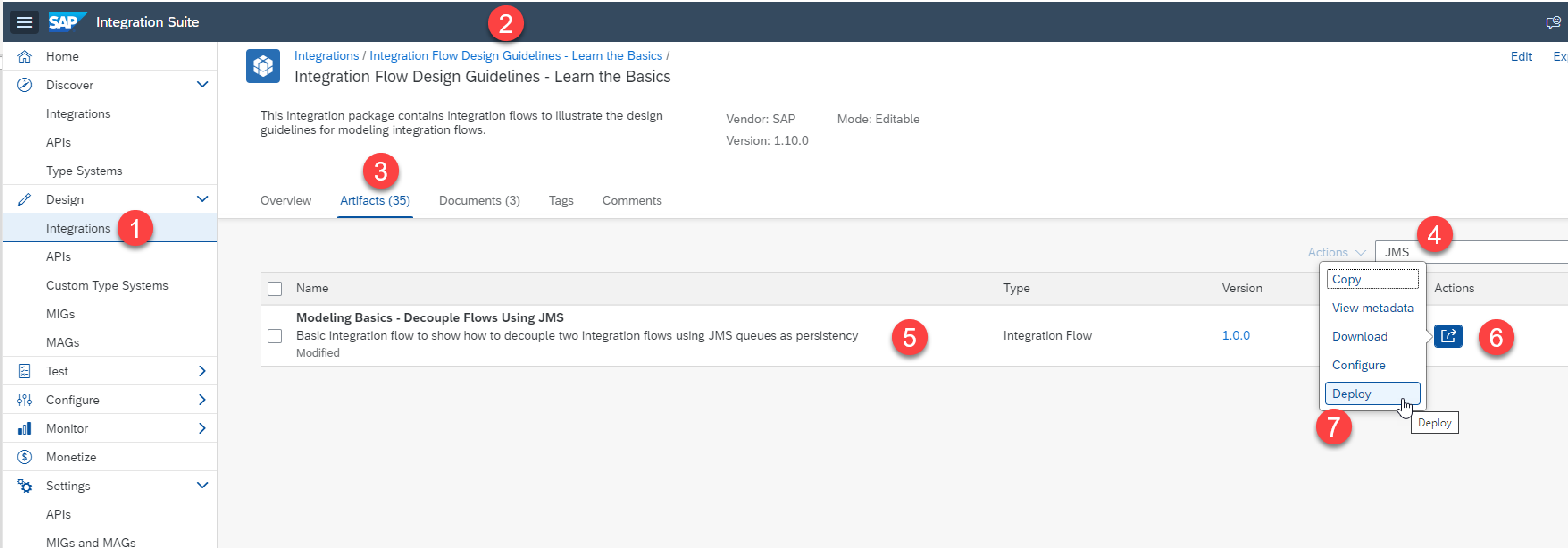
Integrations / Integration Flow Design Guidelines - Learn the Basics /

Integration Flow Design Guidelines - Learn the Basics

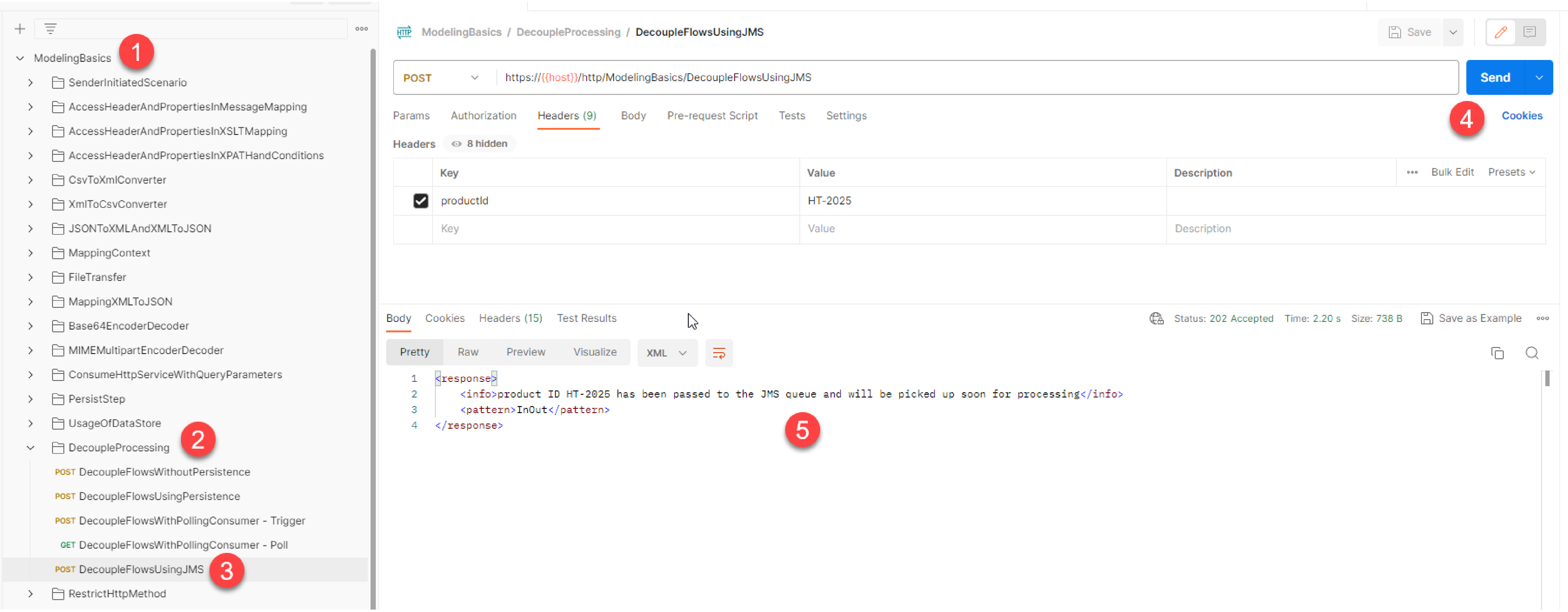
This integration package contains integration flows to illustrate the design guidelines for modeling integration flows.

Vendor: SAP
Version: 1.10.0
Mode: Editable

	Name	Type	Version	Actions
☐	**Generic Receiver** Integration flow mocking a receiver system for test purposes Unmodified	Integration Flow	1.0.0	



	Name	Type	Version	Actions
☐	**Modeling Basics - Decouple Flows Using JMS** Basic integration flow to show how to decouple two integration flows using JMS queues as persistency Modified	Integration Flow	1.0.0	



ModelingBasics
- SenderInitiatedScenario
- AccessHeaderAndPropertiesInMessageMapping
- AccessHeaderAndPropertiesInXSLTMapping
- AccessHeaderAndPropertiesInXPATHandConditions
- CsvToXmlConverter
- XmlToCsvConverter
- JSONToXMLAndXMLToJSON
- MappingContext
- FileTransfer
- MappingXMLToJSON
- Base64EncoderDecoder
- MIMEMultipartEncoderDecoder
- ConsumeHttpServiceWithQueryParameters
- PersistStep
- UsageOfDataStore
- DecoupleProcessing
  - POST DecoupleFlowsWithoutPersistence
  - POST DecoupleFlowsUsingPersistence
  - POST DecoupleFlowsWithPollingConsumer - Trigger
  - GET DecoupleFlowsWithPollingConsumer - Poll
  - POST DecoupleFlowsUsingJMS
- RestrictHttpMethod

ModelingBasics / DecoupleProcessing / DecoupleFlowsUsingJMS

POST https://{{host}}/http/ModelingBasics/DecoupleFlowsUsingJMS

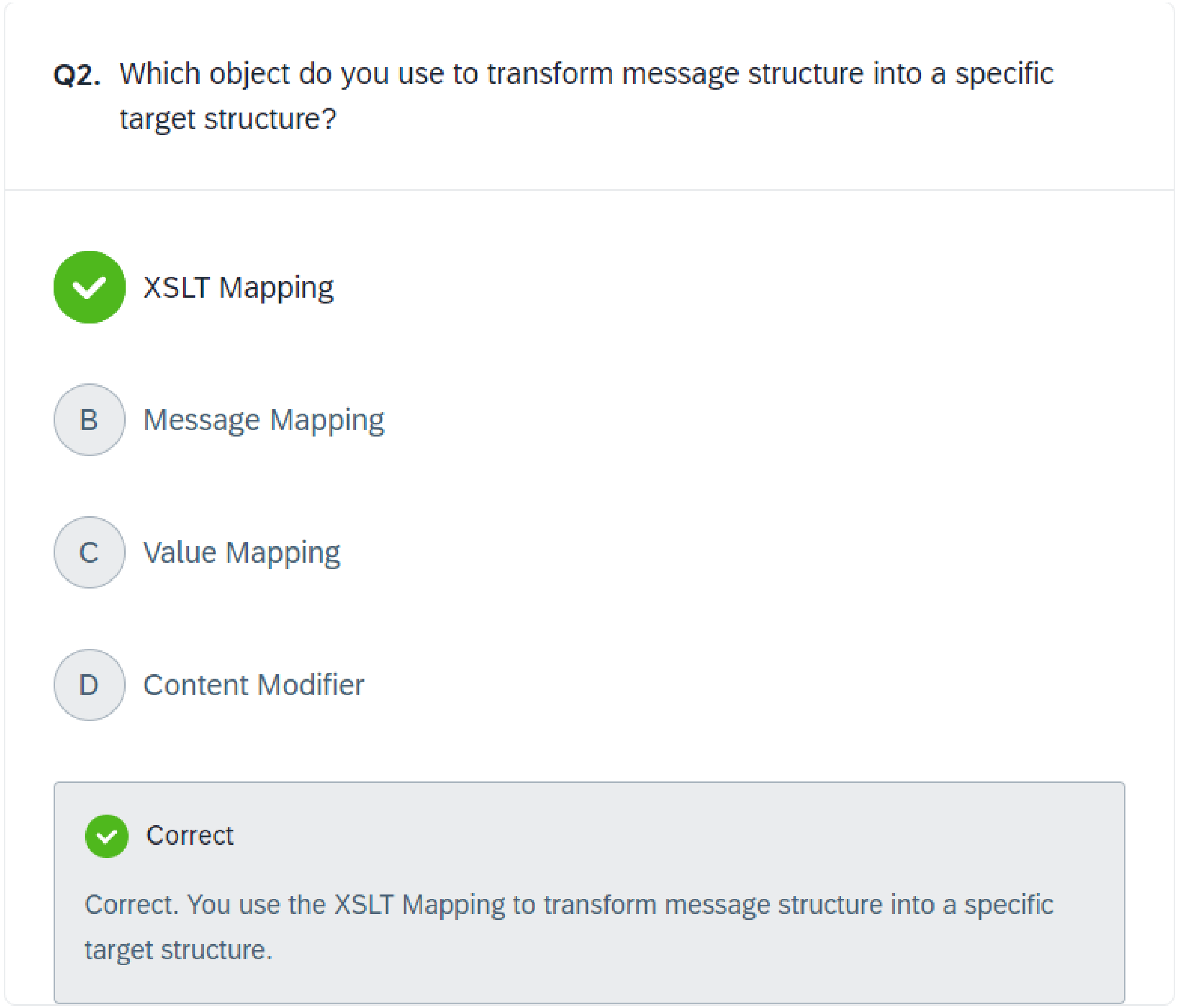
Headers (9) — 8 hidden

	Key	Value	Description
✓	productId	HT-2025	
	Key	Value	Description

Status: 202 Accepted Time: 2.20 s Size: 738 B

```xml
<response>
    <info>product ID HT-2025 has been passed to the JMS queue and will be picked up soon for processing</info>
    <pattern>InOut</pattern>
</response>
```

Key Summary Points – Unit 5



Q2. Which object do you use to transform message structure into a specific target structure?

[✓ XSLT Mapping]

B Message Mapping

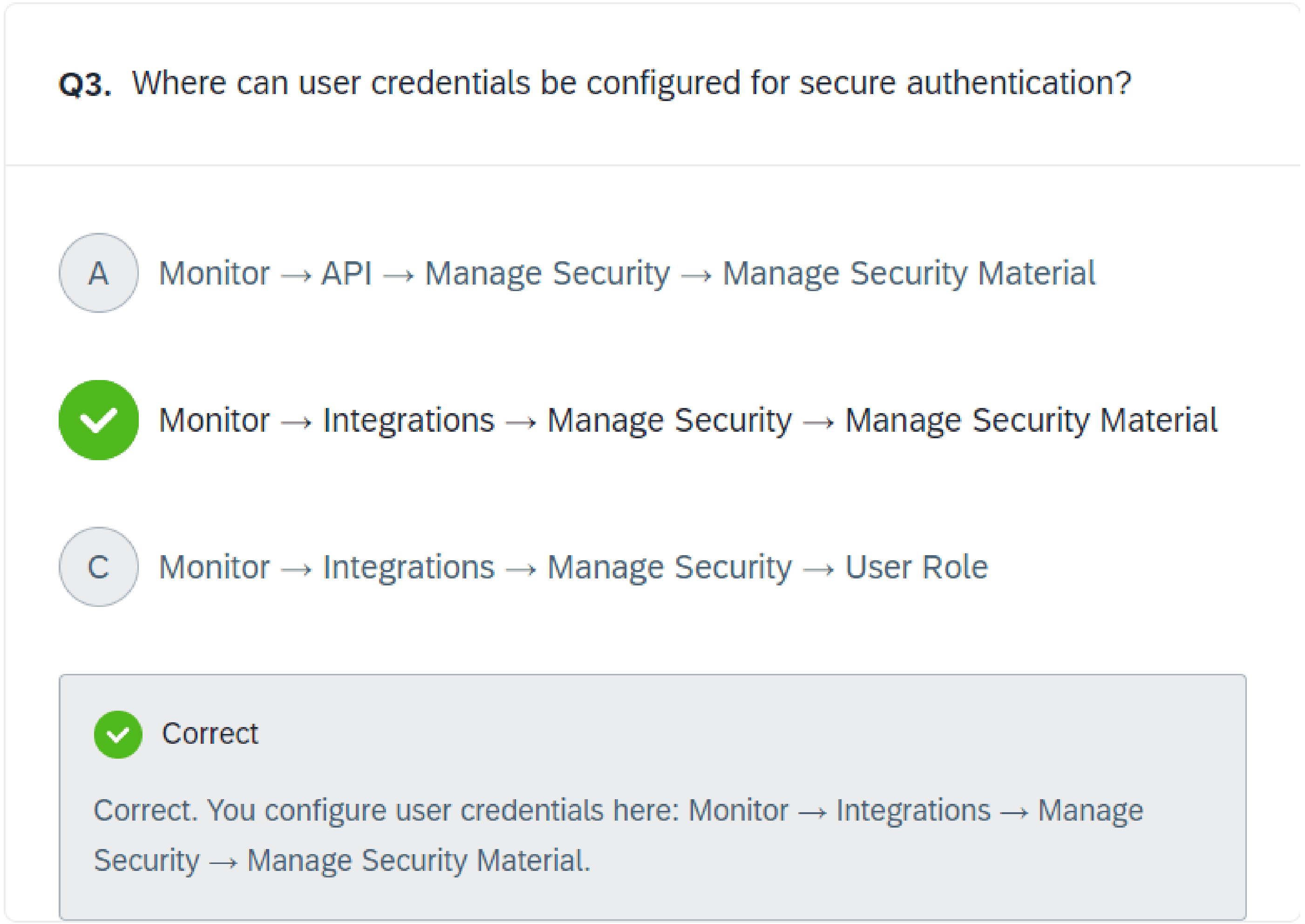
C Value Mapping

D Content Modifier

✓ Correct

Correct. You use the XSLT Mapping to transform message structure into a specific target structure.

Key Summary Points – Unit 5



Q3. Where can user credentials be configured for secure authentication?

A Monitor → API → Manage Security → Manage Security Material

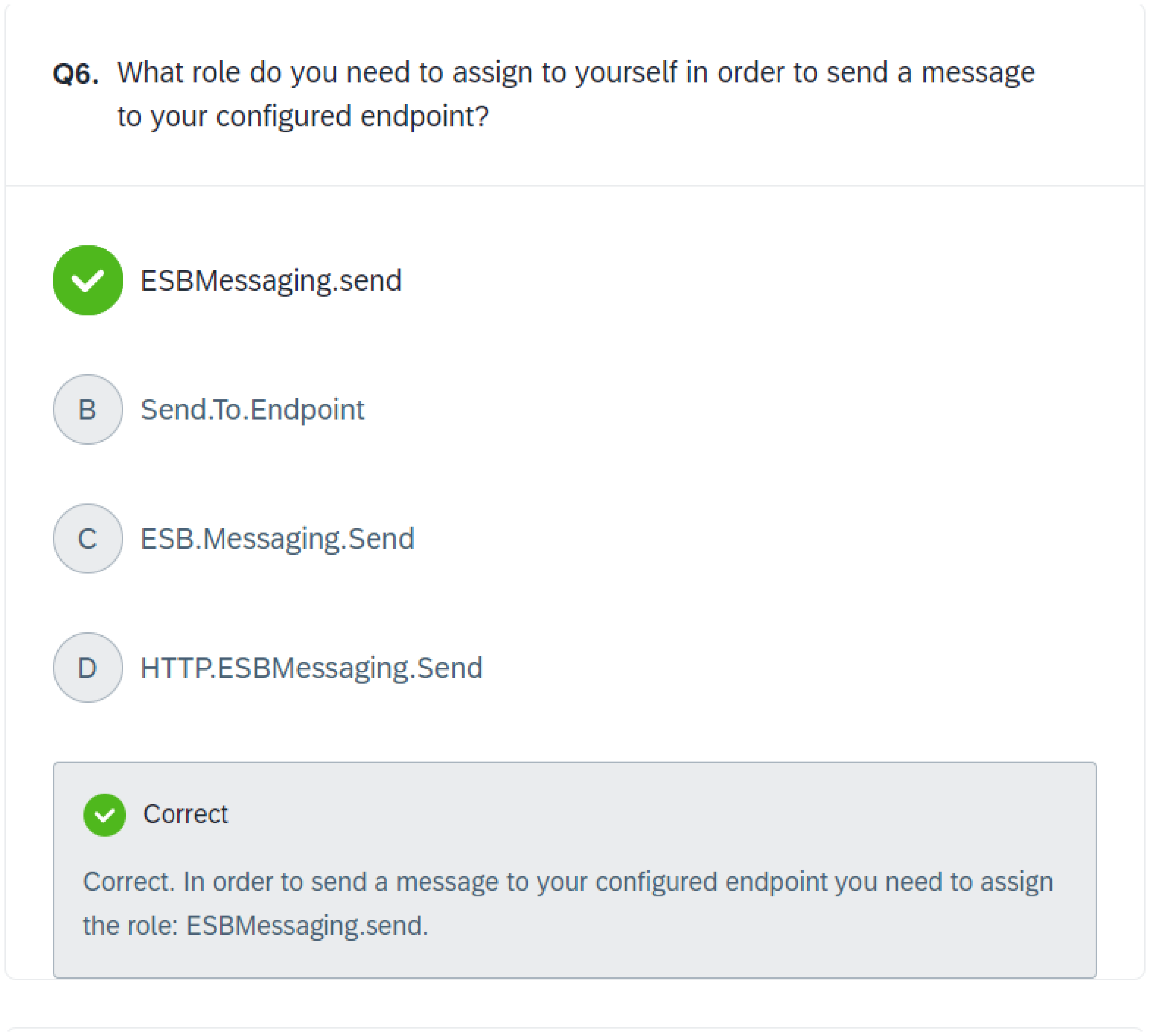
[✓ Monitor → Integrations → Manage Security → Manage Security Material]

C Monitor → Integrations → Manage Security → User Role

✓ Correct

Correct. You configure user credentials here: Monitor → Integrations → Manage Security → Manage Security Material.

Key Summary Points – Unit 5



Q6. What role do you need to assign to yourself in order to send a message to your configured endpoint?

✓ ESBMessaging.send

B Send.To.Endpoint

C ESB.Messaging.Send

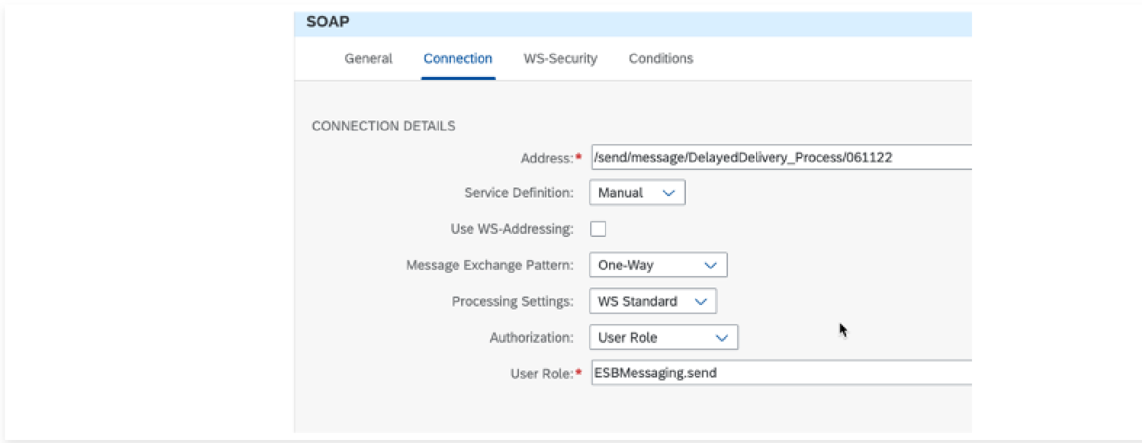
D HTTP.ESBMessaging.Send

✓ Correct

Correct. In order to send a message to your configured endpoint you need to assign the role: ESBMessaging.send.

Key Summary Points – Unit 5

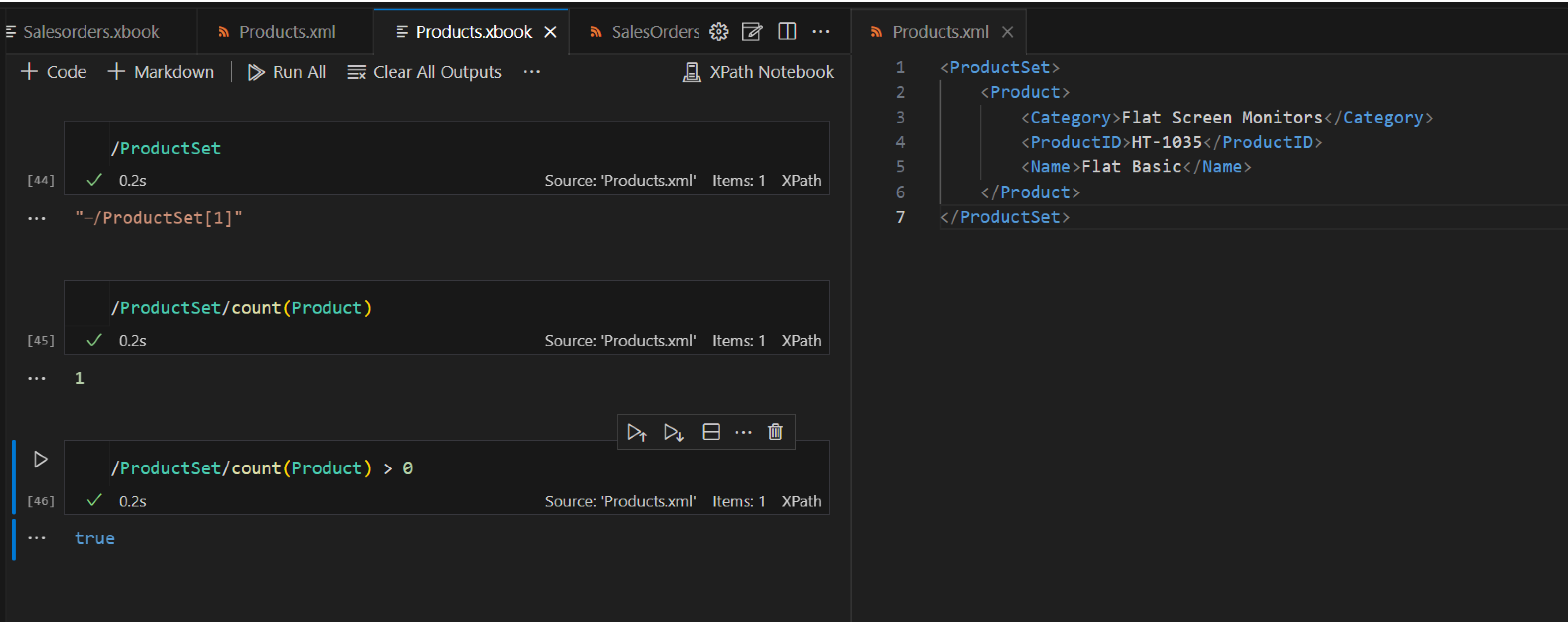
Field Name	Input Data
Address	/send/message/DelayedDelivery_Process/timestamp (must be unique)
Service Definition	Manual
Message Exchange Pattern	One-Way (starting asynchronous)
Processing Settings	WS standard
Authorization	User Role
User Role	ESBMessaging.send



CONNECTION DETAILS

Address:*	/send/message/DelayedDelivery_Process/061122
Service Definition:	Manual
Use WS-Addressing:	☐
Message Exchange Pattern:	One-Way
Processing Settings:	WS Standard
Authorization:	User Role
User Role:*	ESBMessaging.send

XPath Expressions

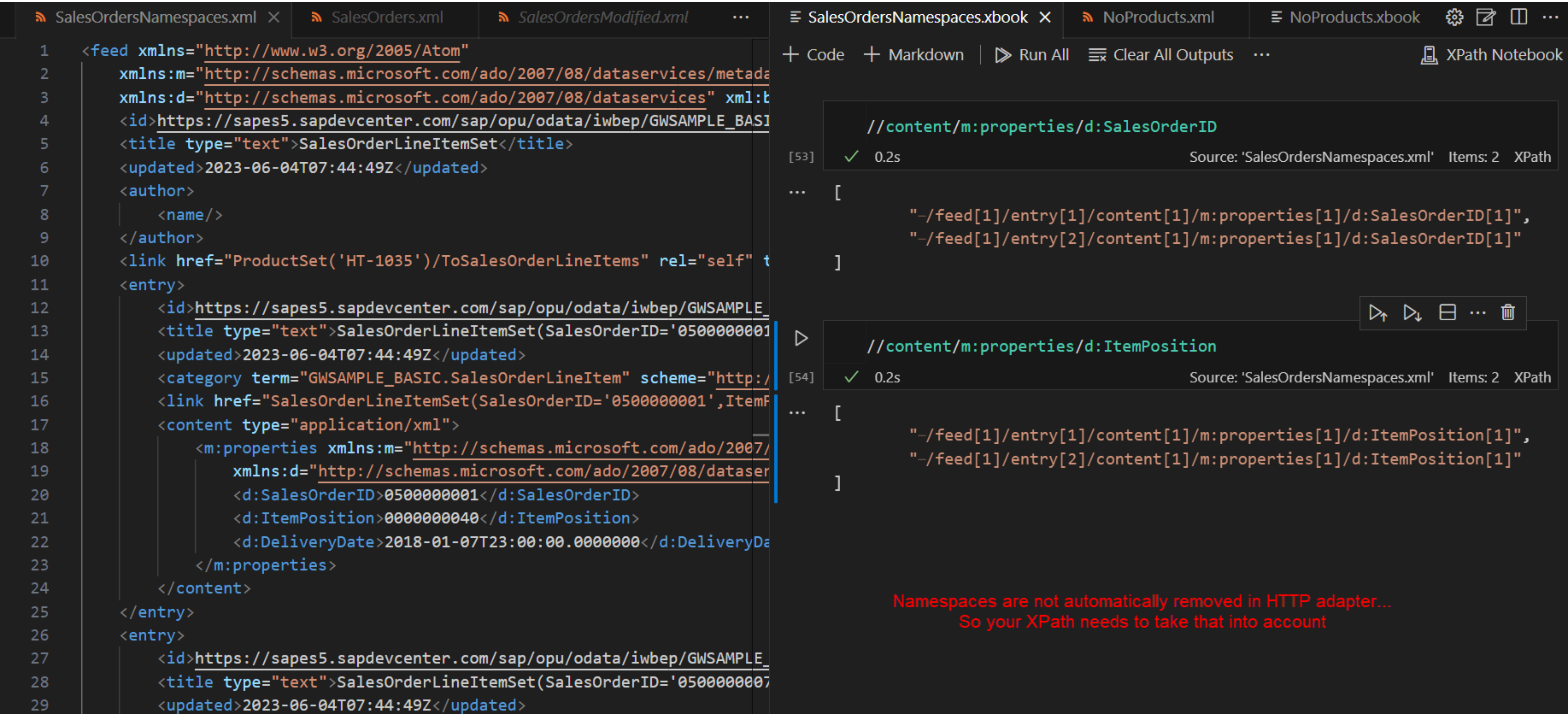


```
/ProductSet
[44] ✓ 0.2s                Source: 'Products.xml'   Items: 1   XPath
"~/ProductSet[1]"

/ProductSet/count(Product)
[45] ✓ 0.2s                Source: 'Products.xml'   Items: 1   XPath
1

/ProductSet/count(Product) > 0
[46] ✓ 0.2s                Source: 'Products.xml'   Items: 1   XPath
true
```

Products.xml

```xml
<ProductSet>
    <Product>
        <Category>Flat Screen Monitors</Category>
        <ProductID>HT-1035</ProductID>
        <Name>Flat Basic</Name>
    </Product>
</ProductSet>
```

HTTP Adapter



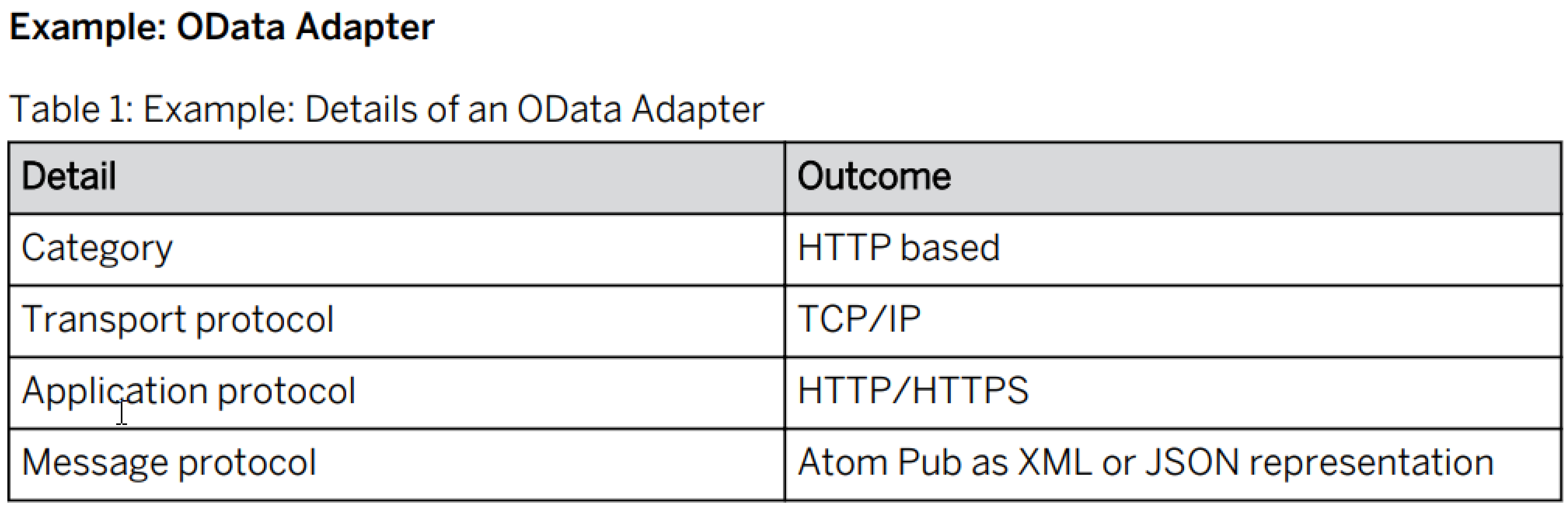
```xml
<feed xmlns="http://www.w3.org/2005/Atom"
    xmlns:m="http://schemas.microsoft.com/ado/2007/08/dataservices/metada
    xmlns:d="http://schemas.microsoft.com/ado/2007/08/dataservices" xml:b
    <id>https://sapes5.sapdevcenter.com/sap/opu/odata/iwbep/GWSAMPLE_BASI
    <title type="text">SalesOrderLineItemSet</title>
    <updated>2023-06-04T07:44:49Z</updated>
    <author>
        <name/>
    </author>
    <link href="ProductSet('HT-1035')/ToSalesOrderLineItems" rel="self" t
    <entry>
        <id>https://sapes5.sapdevcenter.com/sap/opu/odata/iwbep/GWSAMPLE_
        <title type="text">SalesOrderLineItemSet(SalesOrderID='0500000001
        <updated>2023-06-04T07:44:49Z</updated>
        <category term="GWSAMPLE_BASIC.SalesOrderLineItem" scheme="http:/
        <link href="SalesOrderLineItemSet(SalesOrderID='0500000001',ItemP
        <content type="application/xml">
            <m:properties xmlns:m="http://schemas.microsoft.com/ado/2007/
                xmlns:d="http://schemas.microsoft.com/ado/2007/08/dataser
                <d:SalesOrderID>0500000001</d:SalesOrderID>
                <d:ItemPosition>0000000040</d:ItemPosition>
                <d:DeliveryDate>2018-01-07T23:00:00.0000000</d:DeliveryDa
            </m:properties>
        </content>
    </entry>
    <entry>
        <id>https://sapes5.sapdevcenter.com/sap/opu/odata/iwbep/GWSAMPLE_
        <title type="text">SalesOrderLineItemSet(SalesOrderID='0500000007
        <updated>2023-06-04T07:44:49Z</updated>
```

```
//content/m:properties/d:SalesOrderID
[53] ✓ 0.2s    Source: 'SalesOrdersNamespaces.xml'  Items: 2  XPath
[
        "~/feed[1]/entry[1]/content[1]/m:properties[1]/d:SalesOrderID[1]",
        "~/feed[1]/entry[2]/content[1]/m:properties[1]/d:SalesOrderID[1]"
]
```

```
//content/m:properties/d:ItemPosition
[54] ✓ 0.2s    Source: 'SalesOrdersNamespaces.xml'  Items: 2  XPath
[
        "~/feed[1]/entry[1]/content[1]/m:properties[1]/d:ItemPosition[1]",
        "~/feed[1]/entry[2]/content[1]/m:properties[1]/d:ItemPosition[1]"
]
```

Namespaces are not automatically removed in HTTP adapter…
So your XPath needs to take that into account

Details of OData Adapter



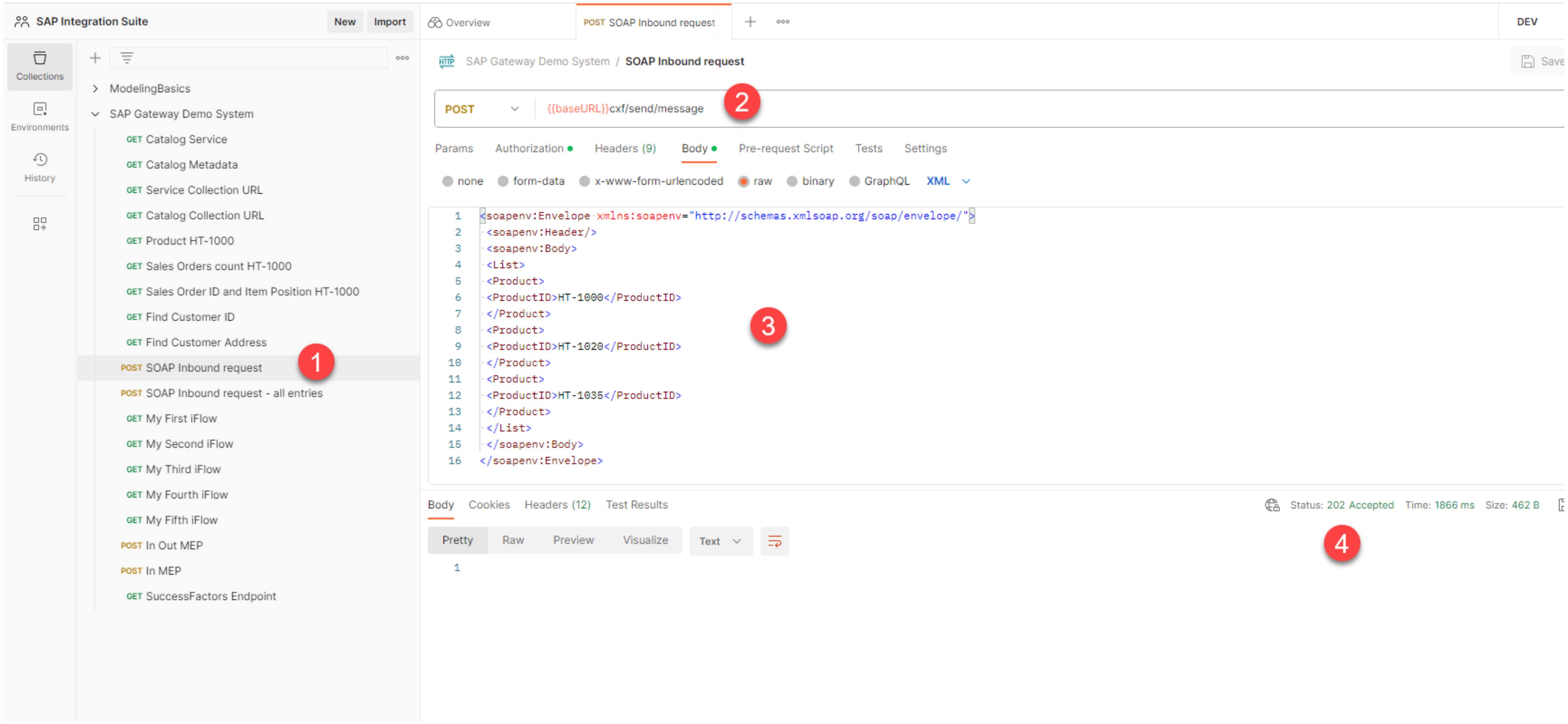
Example: OData Adapter

Table 1: Example: Details of an OData Adapter

Detail	Outcome
Category	HTTP based
Transport protocol	TCP/IP
Application protocol	HTTP/HTTPS
Message protocol	Atom Pub as XML or JSON representation

Features of OData Adapter

- Query wizard
  - Navigate the interface to be accessed with metadata document
- Page Processing mode
  - Read entries in multiple pages which are processed sequentially
  - Overcome challenges with large number of entries
- Automatically removing namespaces
  - Remove namespaces and prefixes automatically

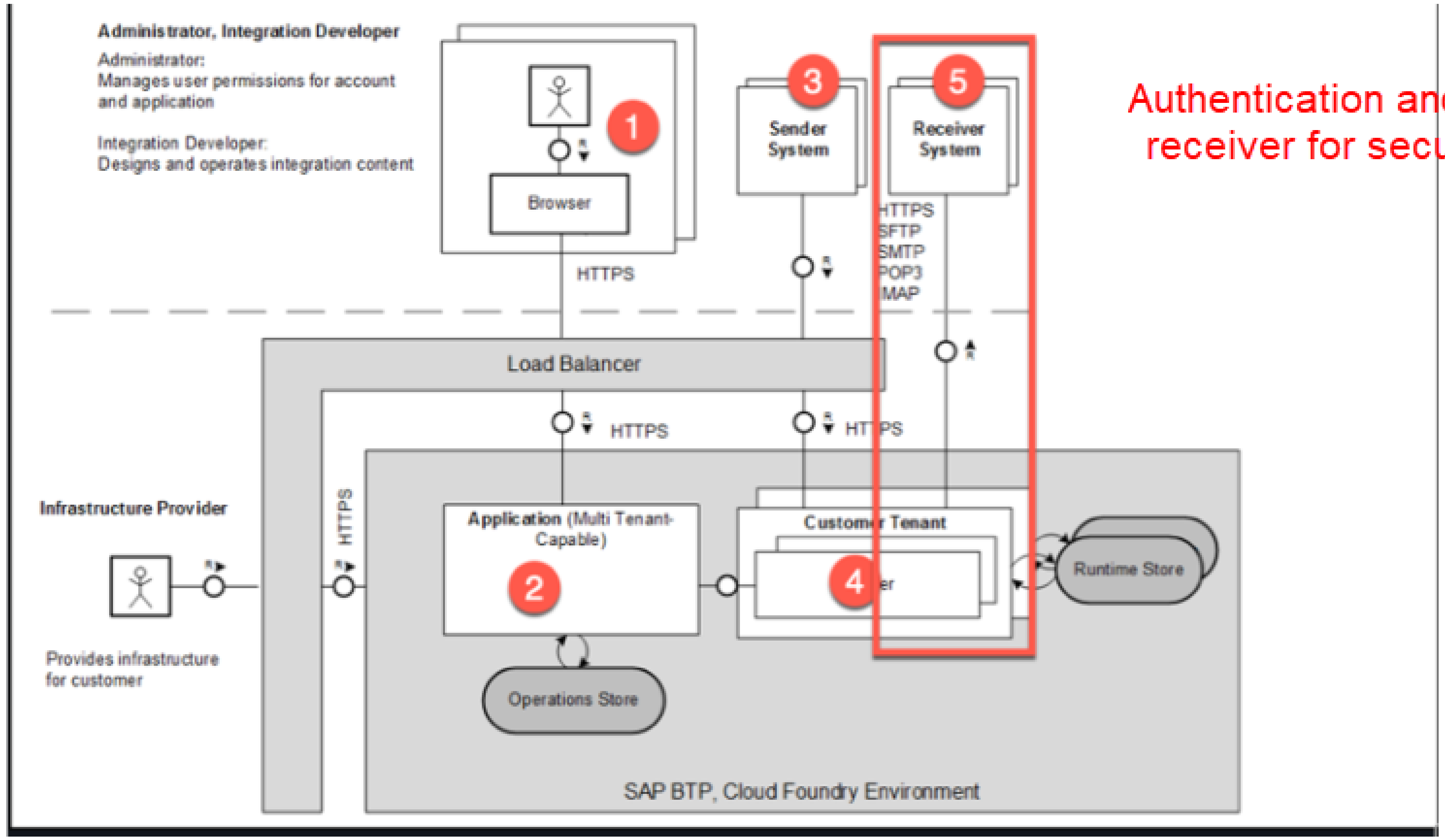


POST {{baseURL}}cxf/send/message

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
 <soapenv:Header/>
 <soapenv:Body>
 <List>
 <Product>
 <ProductID>HT-1000</ProductID>
 </Product>
 <Product>
 <ProductID>HT-1020</ProductID>
 </Product>
 <Product>
 <ProductID>HT-1035</ProductID>
 </Product>
 </List>
 </soapenv:Body>
</soapenv:Envelope>
```

Status: 202 Accepted Time: 1866 ms Size: 462 B

Using Adapter Outbound Security


Administrator, Integration Developer
Administrator:
Manages user permissions for account and application
Integration Developer:
Designs and operates integration content
1
Browser
HTTPS
3
Sender System
5
Receiver System
HTTPS
SFTP
SMTP
POP3
IMAP
Load Balancer
HTTPS
HTTPS
HTTPS
Infrastructure Provider
Provides infrastructure for customer
Application (Multi Tenant-Capable)
2
Customer Tenant
4
Runtime Store
Operations Store
SAP BTP, Cloud Foundry Environment

Authentication and Authorization with receiver for secure communication

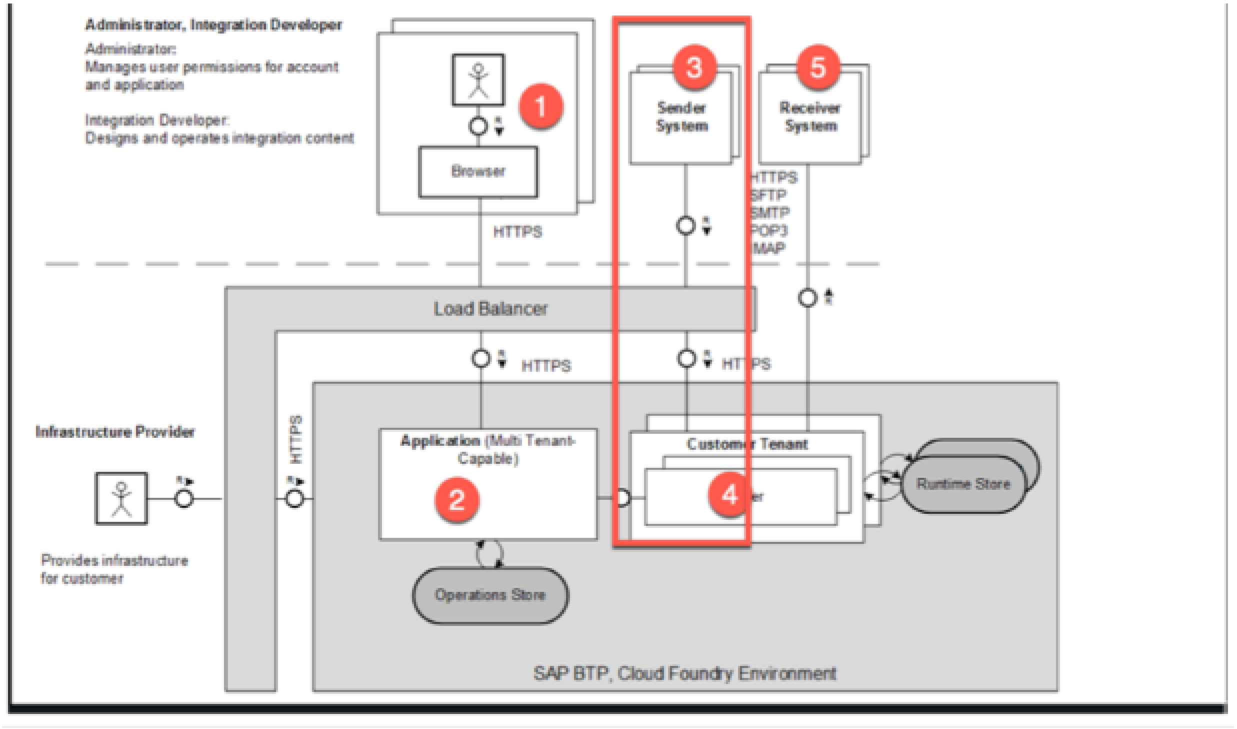
Example of a Direct Receiver and Sender Adapter

Options for Authentication / Authorization

- Basic
- Client Certificate
- None
- OAuth2 Client Credentials
- OAuth2 SAML Bearer Assertion

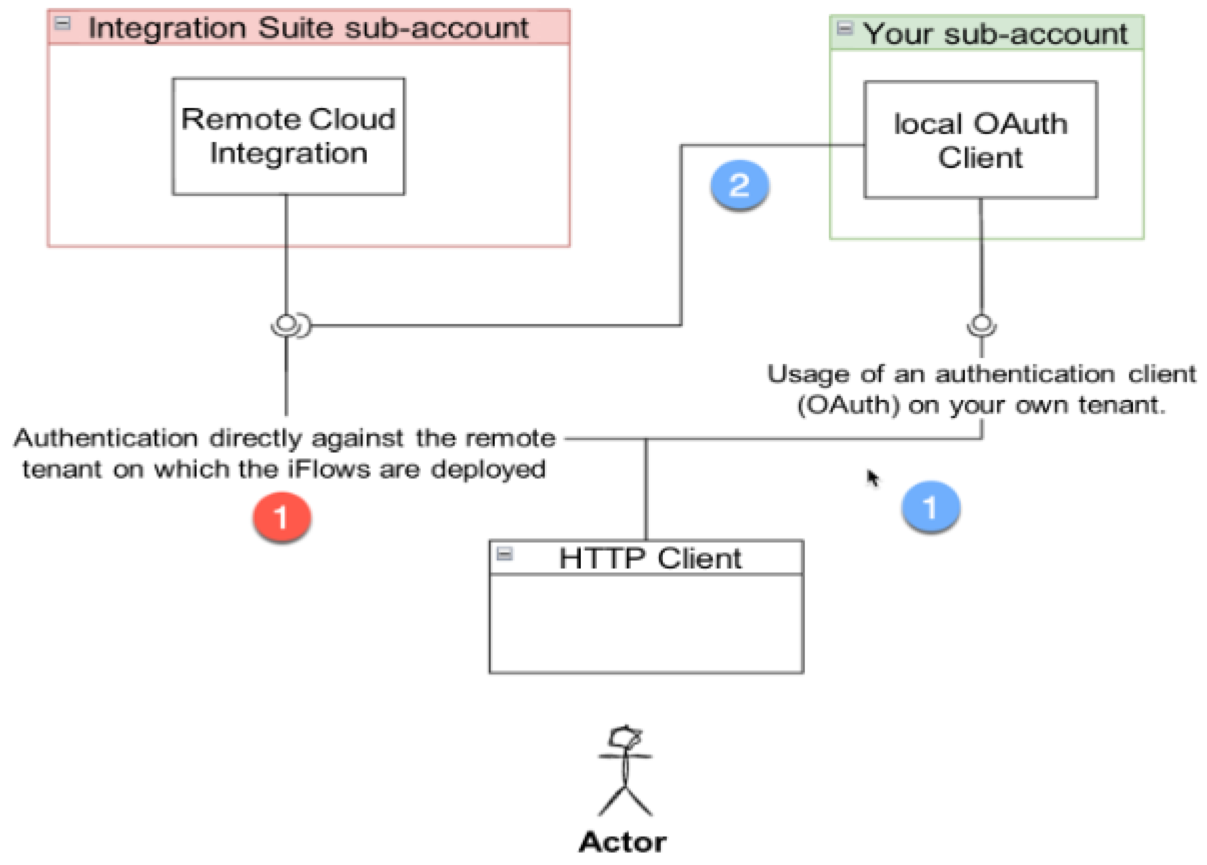
Using Adapter Inbound Security

- Certificates between sender and load balancer for HTTPS connection
- Sender’s authorization validated against Integration flow endpoint



Administrator, Integration Developer
Administrator:
Manages user permissions for account and application
Integration Developer:
Designs and operates integration content
1
Browser
HTTPS
3
Sender System
5
Receiver System
HTTPS
SFTP
SMTP
POP3
IMAP
Load Balancer
HTTPS
HTTPS
HTTPS
Infrastructure Provider
Provides infrastructure for customer
Application (Multi Tenant-Capable)
2
Customer Tenant
4
Runtime Store
Operations Store
SAP BTP, Cloud Foundry Environment

Authorization of sender


Integration Suite sub-account
Remote Cloud Integration
Your sub-account
local OAuth Client
2
Usage of an authentication client (OAuth) on your own tenant.
Authentication directly against the remote tenant on which the iFlows are deployed
1
1
HTTP Client
Actor

Sender Authorization

Authentication against remote endpoint

- Assign user role ESBMessaging.send
- Not recommended for production use

Authentication (OAuth) client on your own Tenant

- Set up Process Integration Runtime instance
- Supports
  - Authorization code
  - Client credentials
  - Password
  - Refresh Token
  - SAML2 Bearer
  - JWT Bearer

Usage of an Authentication (OAuth) Client on your own Tenant

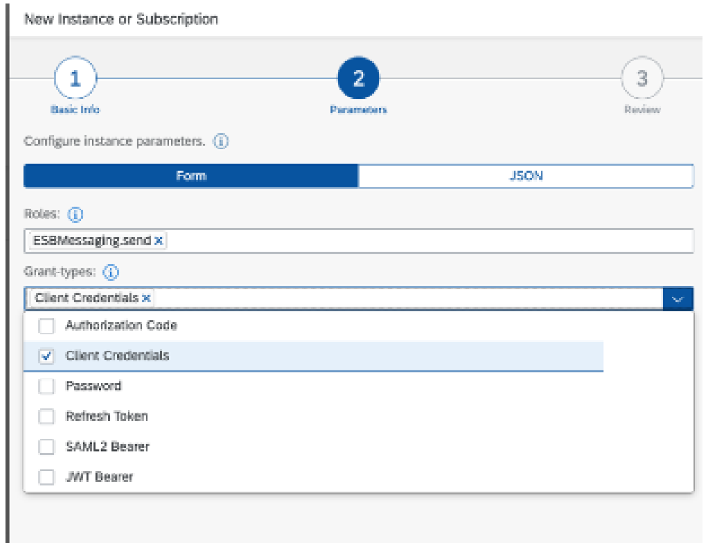
The method of directly calling an integration flow via the role-based approach shown uses personalized users and basic authentication, which are not suitable for productive purposes. For better authentication methods, we need to use a self-configured OAuth2.0 client that can be created on our own subaccount.

To accomplish this, we need to set up a Process Integration Runtime instance on our subaccount, and associate it with the integration flow plan. This instance can then be customized with various client credentials. These correspond to No. 1 and No. 2, marked in blue in the picture above.

You can choose the following grand-types:

- Authorization Code
- Client Credentials
- Password
- Refresh Token
- SAML2 Bearer
- JWT Bearer

Selection of grand types when configuring the local *Process integration Runtime* instance.


New Instance or Subscription

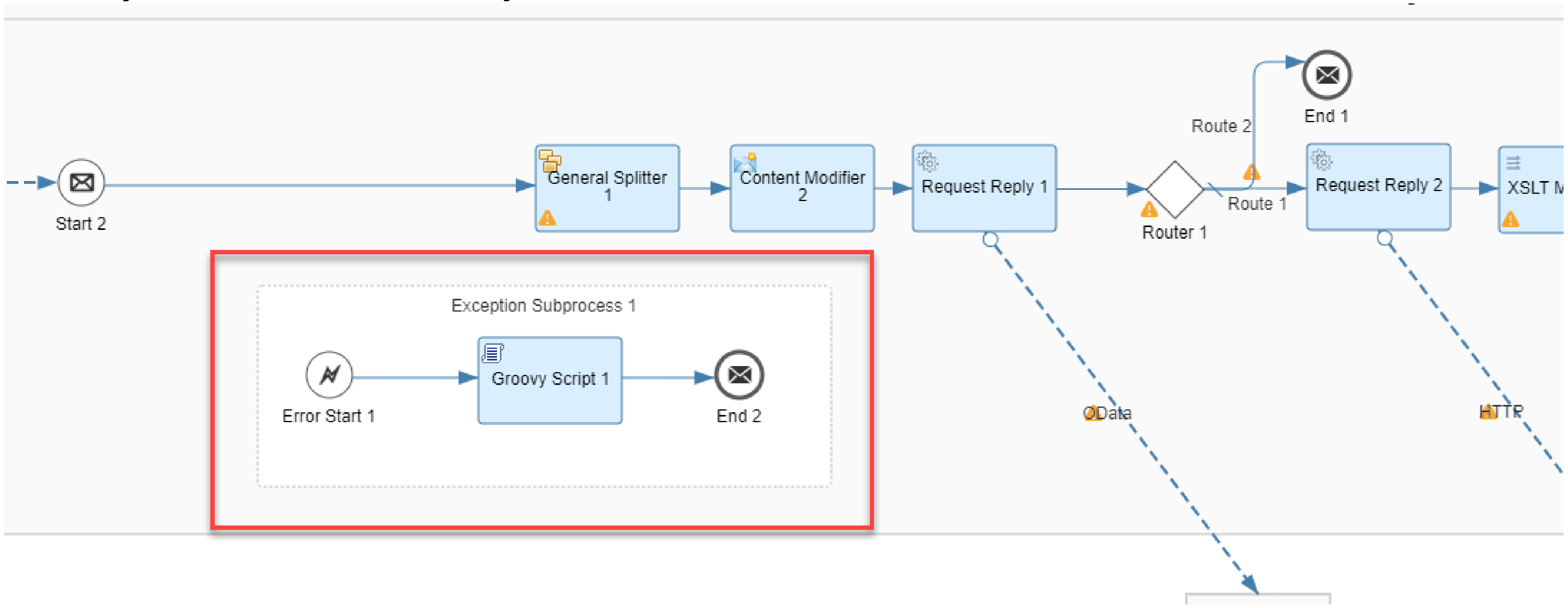
Configure instance parameters.

Roles: ESBMessaging.send

Grant-types: Client Credentials

- [ ] Authorization Code
- [✓] Client Credentials
- [ ] Password
- [ ] Refresh Token
- [ ] SAML2 Bearer
- [ ] JWT Bearer

Key Summary Points – Unit 5



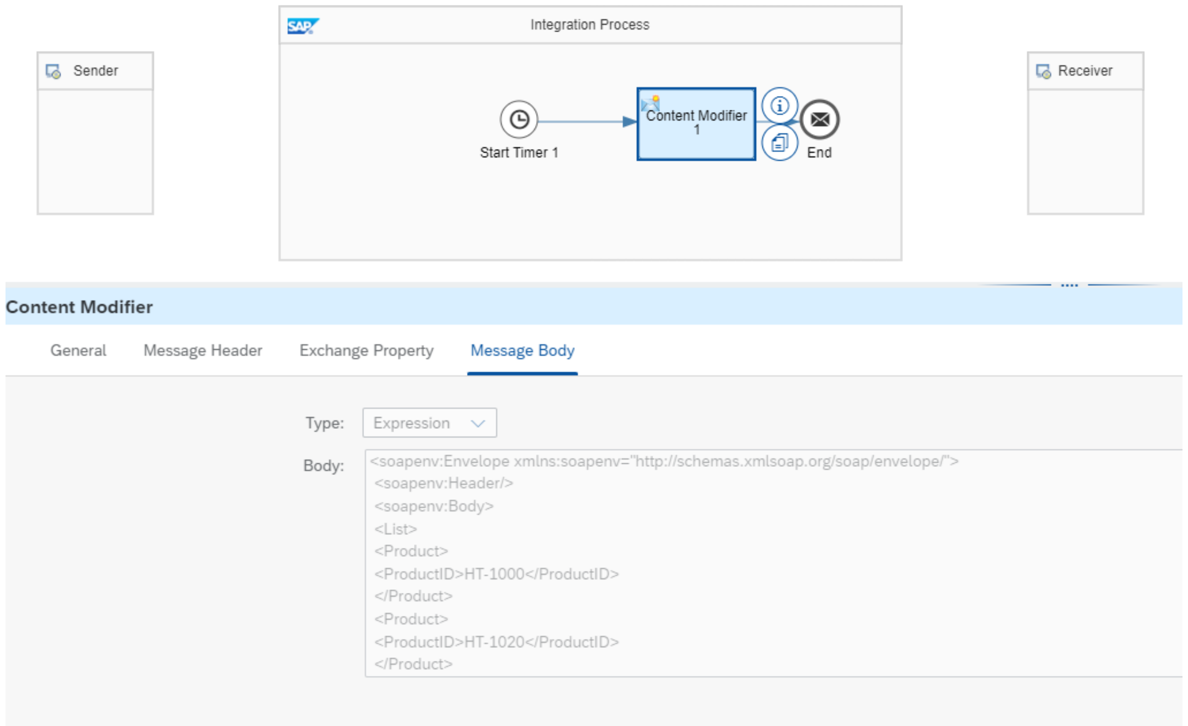
Summary

A special error subprocess can intercept an unexpected error using an Exception Start Event. After interception, various processing steps can be implemented. For instance, it would be appropriate to store process values or message content following an error. Additionally, informing the sender about the error can also be configured.

Key Summary Points – Unit 5

Developer Test with Real Deployment and Debugging of your Integration Flow

Before examining the integration flow, it needs to be deployed in the monitoring environment. The graphical model is converted into a Java application and placed in the runtime, allowing the integration flow to be started. If the deployment is successful, the integration flow will either execute immediately if a timer event is used, or it will wait for an incoming message. Cloud integration offers a trace log level that provides insight into the processing of each integration flow component.

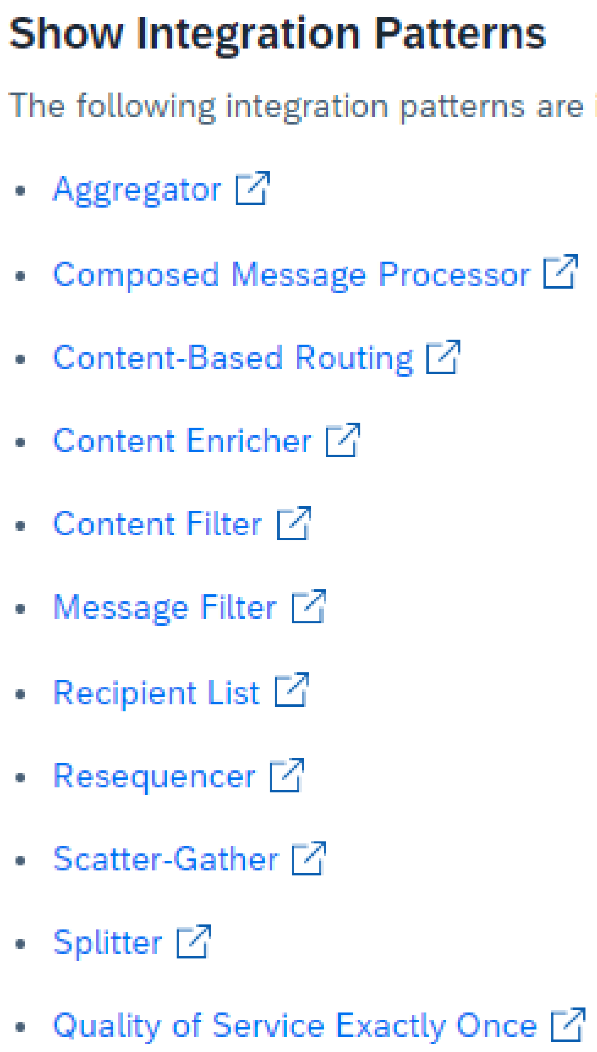


Type: Expression

Body:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
 <soapenv:Header/>
 <soapenv:Body>
 <List>
 <Product>
 <ProductID>HT-1000</ProductID>
 </Product>
 <Product>
 <ProductID>HT-1020</ProductID>
 </Product>
```

Key Summary Points – Unit 5

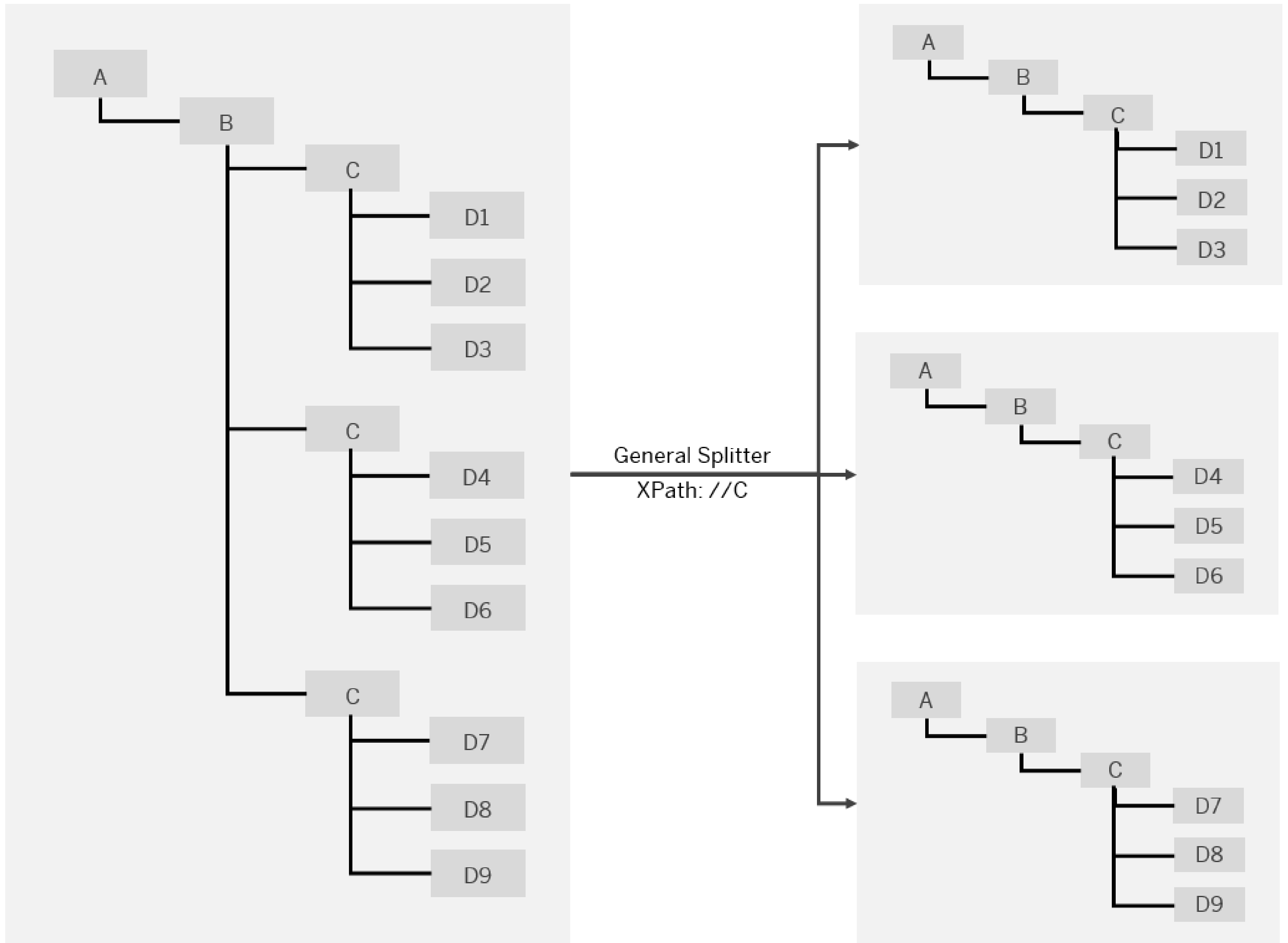


Show Integration Patterns

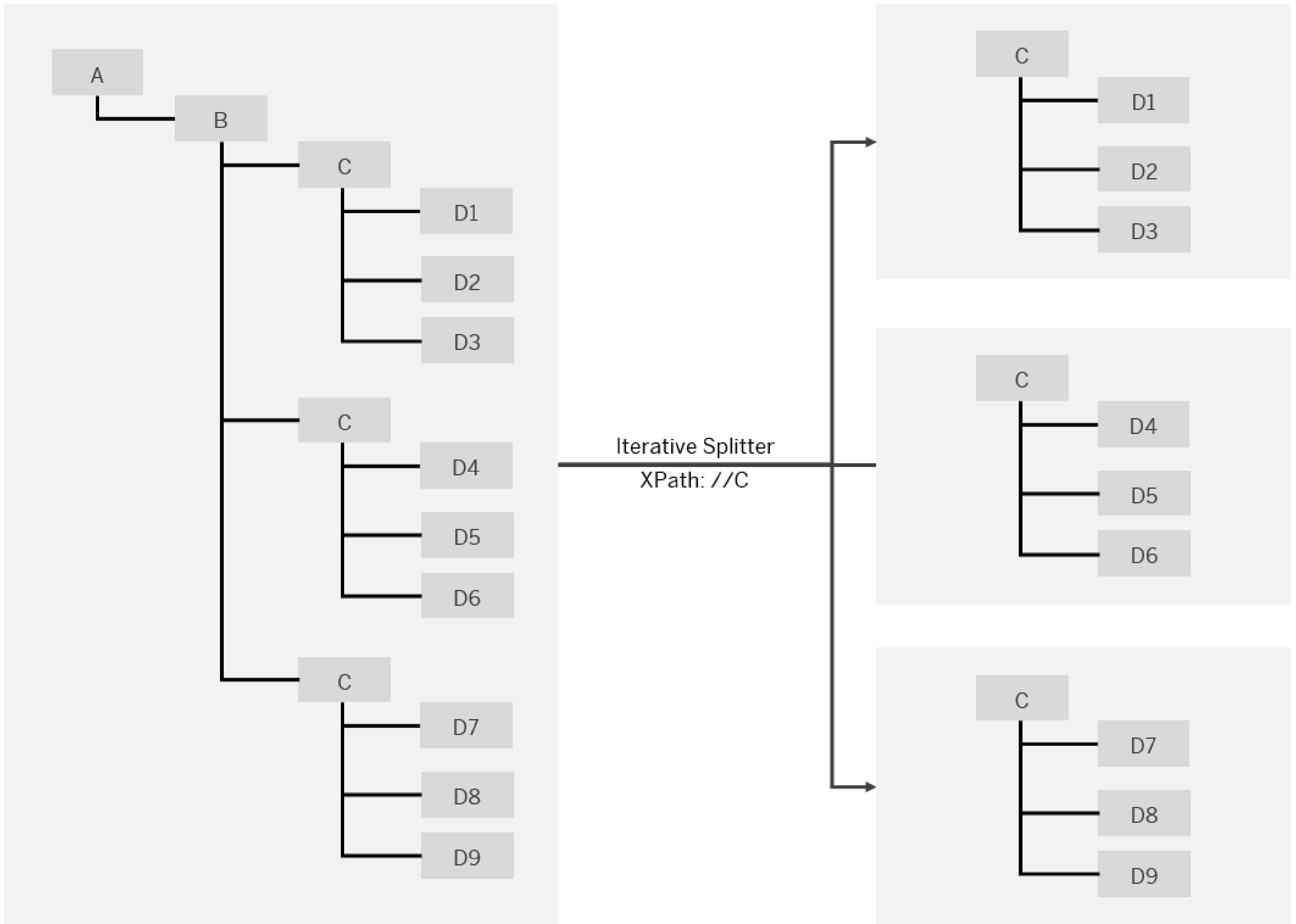
The following integration patterns are included in the example package:

- Aggregator
- Composed Message Processor
- Content-Based Routing
- Content Enricher
- Content Filter
- Message Filter
- Recipient List
- Resequencer
- Scatter-Gather
- Splitter
- Quality of Service Exactly Once

Key Summary Points – Unit 5



Key Summary Points – Unit 5


A
B
C
D1
D2
D3
C
D4
D5
D6
C
D7
D8
D9
Iterative Splitter
XPath: //C
C
D1
D2
D3
C
D4
D5
D6
C
D7
D8
D9