Oracle® Cloud

Using B2B for Oracle Integration 3

F45541-11
April 2024

Oracle Cloud Using B2B for Oracle Integration 3,

F45541-11

Primary Author: Oracle Corporation

Contents

Preface

1 Introduction to B2B for Oracle Integration

2 Use B2B for Oracle Integration in Trading Partner Mode

Preface

This guide describes how to use B2B for Oracle Integration.

> **Note:**
>
> The use of this adapter may differ depending on the features you have, or whether your instance was provisioned using Standard or Enterprise edition. These differences are noted throughout this guide.

Topics:

- Audience
- Documentation Accessibility
- Diversity and Inclusion
- Related Resources
- Conventions

Audience

This guide is intended for developers who want to use B2B for Oracle Integration.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc`.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` or visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` if you are hearing impaired.

Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation. We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

Related Resources

See these Oracle resources:

- Oracle Cloud at `http://cloud.oracle.com`
- *Using Integrations in Oracle Integration 3*
- *Using the Oracle Mapper with Oracle Integration 3*
- Oracle Integration documentation on the Oracle Help Center.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1

Introduction to B2B for Oracle Integration

Learn about how Oracle Integration provides e-commerce support with B2B for Oracle Integration.

Topics:

- Business-to-Business E-Commerce
- What Is B2B for Oracle Integration
- Supported Protocols and Standards
- Two Patterns For Using B2B for Oracle Integration

Business-to-Business E-Commerce

In B2B e-commerce, an enterprise extends its business processes to reach trading partners. B2B e-commerce represents classic business processes, mature business documents, and industry-tempered messaging services. It requires a unified business process platform, end-to-end instance tracking, visibility and auditing, integrated process intelligence, process and service governance, and centralized security.

What Is B2B for Oracle Integration

Oracle Integration provides support for B2B e-commerce with B2B for Oracle Integration.

B2B for Oracle Integration represents a collective set of features inside Oracle Integration to support processing, including:

- A trading partner management interface.
- A schema and document editor for customization.
- A specialized B2B tracking user interface for viewing messages exchanged with trading partners.
- Adapters to support B2B protocols such as AS2, RosettaNet, and FTP. See Transport Protocols Supported in B2B for Oracle Integration.
- Actions you place within your integrations for EDI processing. See Business Protocols Supported in B2B for Oracle Integration.

Supported Protocols and Standards

B2B for Oracle Integration supports the following protocols and standards.

Topics:

- About Electronic Data Interchange
- About X12 HIPAA

- About RosettaNet
- Business Protocols Supported in B2B for Oracle Integration
- Transport Protocols Supported in B2B for Oracle Integration
- REST Support
- Open Applications Group (OAGIS) Support
- Custom XML Support
- Positional Flat or Fixed Length Payloads Support
- Delimited Payload Support

About Electronic Data Interchange

Electronic Data Interchange (EDI) covers a set of data formats used to exchange formal business documents between companies.

These data formats have a long history of usage in the industry starting from the 1980s. EDI formats are standardized by various governing bodies. The two major formats are:

- EDI X12 format, governed by the ANSI X12 standards body. EDI X12 is a data format used to exchange specific data between trading partners. The term trading partner may represent an organization, group of organizations, or some other entity. It is typically just an organization or company.
- UN/EDIFACT format, governed by the United Nations Economic Commission for Europe (UNECE).

The governing bodies define formats for several types of business documents, such as a purchase order, invoice, insurance claim, and so on that are used across many vertical industries, such as supply chain, finance, transportation, and insurance. While EDI has a dominant place in B2B (business to business) communications, other formats based on XML are also in use today. B2B for Oracle Integration currently supports the EDI X12 and UN/EDIFACT formats.

Here's an example purchase order in the ANSI EDI X12 format.

```
ISA|00|          |00|          |01|111111T        |01|22222          |
190312|0845|U|00401|000001894|0|T|~
GS|PO|0124578|ACME|20190312|084515|0123456|X|004010
ST|850|1234
BEG|00|NE|45030000000||20190312
CUR|2L|USD
REF|IT|999|Global Chips
REF|WO|P8923.5
REF|KY||Standard Terms and conditions will apply
FOB|PC|OR|ORIGIN FREIGHT COLLECT|02|FOB
ITD||3|||||10|||||P03R
DTM|002|20190329
TD5|B||05||Ground
N1|BT|Global Chips
N2|ACCOUNTS PAYABLE
N3|P.O. BOX 1111
N4|NEW YORK|NY|10001|US
PER|BD|JAMES SMITH|TE|1112223333|EM|jamess@globalchips.com
```

```
PO1|00001|1|EA|74.99|PE|BP|5566|VN|AB-1264
PID|F||||AB-1264 BRACKET ASSY WITH SPRING
PO1|00002|1|EA|25.00|PE|BP|7264|VN|DE-1834
PID|F||||DE-1834 GEAR BOX PACKAGE
CTT|2|2
AMT|TT|99.99
SE|22|1234
GE|1|0123456
IEA|1|000001894
```

Here's an example purchase order in the ANSI EDI UN/EDIFACT format.

```
UNB+UNOA:1+US::US+50138::THEM+140531:0305+001934++ORDERS'
UNH+1+ORDERS:91:2:UN'
BGM+220+A761902+4:20140530:102+9'
RFF+CT:EUA01349'
RFF+AAV::C'
TXT+THIS IS WHAT AN EDI MESSAGE WOULD LOOK LIKE... '
NAD+BY++OUR NAME PLC::::+++++EW4 34J'
CTA+PD'
COM+01752 253939:TE+01752 253939:FX+0:TL'
CTA+OC+:A.SURNAME'
COM+2407:EX'
CTA+TI+:B.BROWN'
COM+0:EX'
CTA+SU'
COM+0161 4297476:TE+01752 670633:FX'
UNT+15+1'
UNZ+1+001934'
```

About X12 HIPAA

B2B for Oracle Integration supports the X12 HIPAA standard to exchange documents over the internet. X12 HIPAA is a subset of the X12 standard and is used in the healthcare industry for the exchange of data about health insurance claims and payments.

B2B for Oracle Integration translates inbound X12 HIPAA messages into XML format for processing and converts XML to X12 HIPAA version 2 format for outbound delivery. You create an X12 HIPAA schema and document and a trading partner in which the document can be used in both inbound and outbound agreements.

You define an AS2 Adapter as a trigger and invoke connection to handle X12 HIPAA data. The AS2 Adapter provides the following capabilities:

- Trigger (inbound) direction:
  - The AS2 Adapter trigger connection receives the data in native X12 HIPAA version 2 format from the trading partner, unpacks it, and delivers it for translation to XML format by the B2B action in an integration.
  - The XML is sent to the backend integration for further processing.
- Invoke (outbound) direction:
  - The XML data is translated in the backend integration to native X12 HIPAA version 2 format.

  - The AS2 Adapter invoke connection packs the received X12 HIPAA format and sends it to the target trading partner.

B2B for Oracle Integration supports the following X12 HIPAA version 2.0 standards:

- 005010X212 Health Care Claim Status Request and Response
- 005010X214 Health Care Claim Acknowledgment
- 005010X217 Health Care Services Review - Request for Review and Response
- 005010X218 Payroll Deducted and Other Group Premium Payment For Insurance Products
- 005010X220 Benefit Enrollment and Maintenance
- 005010X221 Health Care Claim Payment Advice
- 005010X222 Health Care Claim Professional
- 005010X223 Health Care Claim Institutional
- 005010X224 Health Care Claim Dental
- 005010X231 Implementation Acknowledgment for Health Care Insurance
- 005010X279 Health Care Eligibility Benefit Inquiry and Response

About RosettaNet

B2B for Oracle Integration supports the nonproprietary, XML-based RosettaNet standard to exchange documents over the internet. RosettaNet standards prescribe when to exchange, acknowledge, or confirm information, and how to package and physically exchange messages between trading partners.

- RosettaNet Concepts
- How B2B for Oracle Integration Supports RosettaNet

RosettaNet Concepts

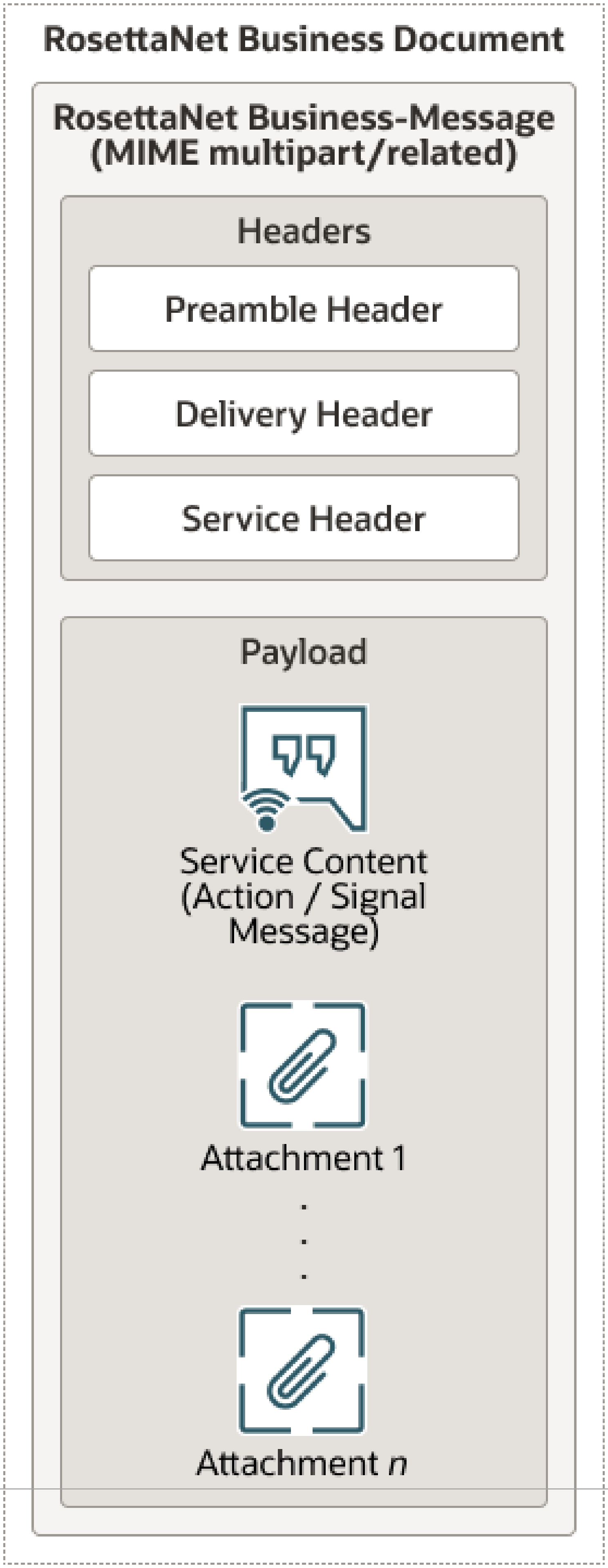
RosettaNet is a global standard for electronic B2B communication and collaboration in the supply chain management industry. RosettaNet includes a set of standardized message formats, business processes, and implementation frameworks to facilitate electronic data exchange and collaboration between trading partners.

A complete RosettaNet implementation consists of three core elements:

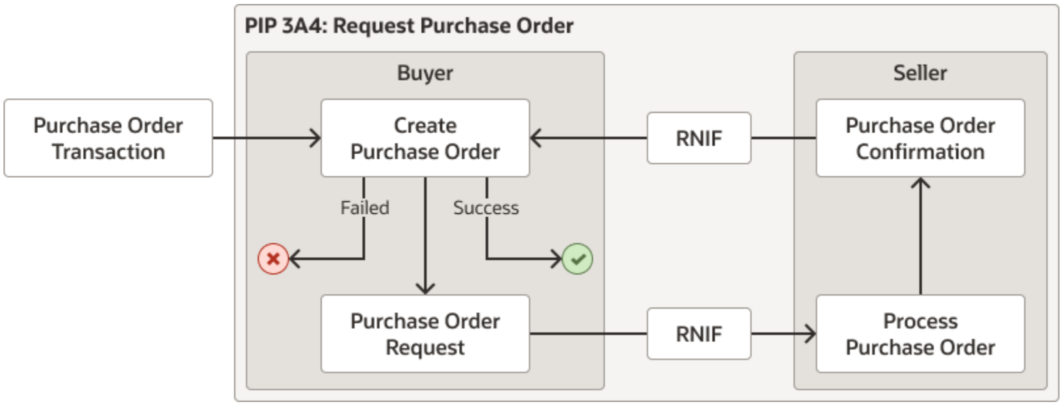
- RosettaNet Implementation Framework (RNIF)
  RNIF is a collection of specifications used to implement message exchange between trading partners over the internet. RNIF provides the following guidelines for RosettaNet partner interface processes (PIPs) messages and business signals:
  - Transport
  - Packaging
  - Authorization
  - Authentication
  - Encryption
  - Compression
  - Digital signature

  - Message digest
  - Nonrepudiation (so trading partners cannot dispute receiving a message)

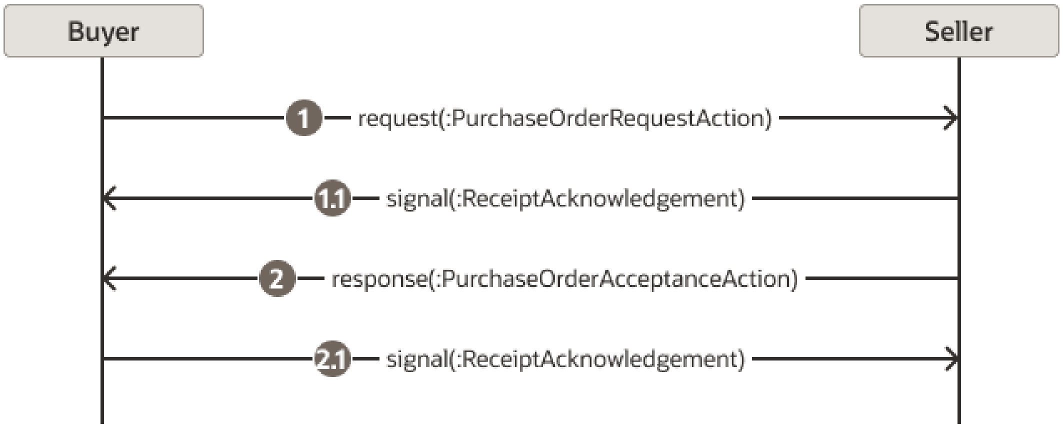
A RosettaNet business document is structured as follows:


RosettaNet Business Document
RosettaNet Business-Message (MIME multipart/related)
Headers
Preamble Header
Delivery Header
Service Header
Payload
Service Content (Action / Signal Message)
Attachment 1
.
.
.
Attachment *n*

- Partner Interface Processes (PIPs)
  RosettaNet PIPs define the sequence of steps required to execute a business process between two trading partners. B2B for Oracle Integration supports all PIPs. For example, the message exchange sequence for PIP 3A4 (Request Purchase Order) is as follows. This is a double-action PIP. For a double-action PIP, the initiator receives a response message and replies with an acknowledgment.

  A functional view of PIP 3A4 in action is as follows.

  
  Buyer
Seller
1 request(:PurchaseOrderRequestAction)
1.1 signal(:ReceiptAcknowledgement)
2 response(:PurchaseOrderAcceptanceAction)
2.1 signal(:ReceiptAcknowledgement)

  For a single-action PIP, the only response is an acknowledgment signal message.

  See PIP Directory.
- Business and technical dictionaries
  B2B for Oracle Integration Rosetta support doesn't interpret the RosettaNet dictionaries. The dictionaries are left to interpretation by the backend applications.

How B2B for Oracle Integration Supports RosettaNet

You follow this workflow to use RosettaNet in B2B for Oracle Integration:

- You configure the RosettaNet Adapter connection in the trigger direction and invoke direction. See RosettaNet Adapter Capabilities in *Using the RosettaNet Adapter with Oracle Integration 3*. This configuration is similar to the AS2 Adapter.

- You create a RosettaNet schema (by uploading the required PIP) and document. See B2B Schemas and B2B Documents.
- You correlate the message sequence by creating an XPath expression in the document. This ensures that the business document messages and wire messages are in the proper order at runtime on the Track B2B messages page.
- You assign a RosettaNet-required DUNS business identifier with a nine digit, alphanumeric character value when defining the trading partners and select RosettaNet as the transport type. See Manage B2B Trading Partners.
- You track the business document and wire messages for the B2B integration during runtime. See B2B Tracking.
  A **Collaboration** tab is provided for viewing. This tab enables you to ensure that the sequence of messages is correct at runtime. See Work with Business Messages.

Business Protocols Supported in B2B for Oracle Integration

B2B for Oracle Integration supports the EDI X12, X12 HIPAA, EDIFACT, and RosettaNet business protocols for the exchange of business documents between Oracle Integration and a trading partner.

Each business protocol supports the following:

- EDI X12 supports versions 4010 through 8040 (includes all document types within each version).
- X12 HIPAA supports version 2.0 standards. See About X12 HIPAA.
- EDIFACT supports versions D00A through S04A D96A (includes all document types within each version).
- RosettaNet version 2.0 supports all PIP versions.

Transport Protocols Supported in B2B for Oracle Integration

The transport protocol defines the underlying transport used to send and receive data between trading partners. B2B for Oracle Integration supports the following protocols:

- AS2 (Applicability Statement 2)
- FTP and SFTP
- RosettaNet version 2.0
- REST

The following protocols are available for use in B2B for Oracle Integration, but do not support trading partner mode.

- AQ
- File
- REST
- SOAP
- JMS

REST Support

You can exchange inbound and outbound messages with trading partners over the REST transport in B2B for Oracle Integration. The REST transport is useful for trading partners that don't use EDI or RosettaNet and can exchange messages using web forms or WebUI.

You send and receive messages over REST calls using existing REST Adapter connections configured in Oracle Integration. You can also exchange EDI, X12, and custom XML documents over the REST transport.

How B2B for Oracle Integration Supports REST

You follow this workflow to use REST in B2B for Oracle Integration:

- You configure the REST Adapter as a trigger (inbound) connection and invoke (outbound) connection. These connections are used for connectivity to your trading partners. See Create a REST Adapter Connection in *Using the REST Adapter with Oracle Integration 3*. You can configure these connections either outside of B2B for Oracle Integration or within B2B for Oracle Integration during REST transport configuration.
- You select REST as the transport to use. See Define Transports.
- You specify the REST Adapter connections to receive inbound messages and send outbound messages when configuring the REST transport.

> **Note:**
>
> Only XML documents are currently tested and certified.

- You track the business document and wire messages for the B2B integration during runtime. See B2B Tracking.

Open Applications Group (OAGIS) Support

B2B for Oracle Integration supports Open Applications Group (OAGIS) documents. OAGIS is a supply chain domain standard that specifies a protocol for passing data in and out of applications using XML format. Under the OAGIS standard, related information is bundled into a structure called a business object document (BOD).

B2B for Oracle Integration provides support for the following:

- OAGIS schema and document definitions. The following document versions are supported:

7.2	7.2.1	10.1
SHOW_SHIPMENT_005 (I)	PROCESS_PO_007 (I)	ProcessInvoice (I)
SHOW_SHIPMENT_002 (I)	PROCESS_INVOICE_002 (I/O)	ProcessShipment (I)
CHANGE_PO_006 (O)	ACKNOWLEDGE_PO_008	ConfirmBOD (I)
PROCESS_PO_007(O)	CHANGE_PO_006	AcknowledgePurchaseOrder (I)
ACKNOWLEDGE_PO_008	CONFIRM_BOD_004	ChangePurchaseOrder (O)
CONFIRM_BOD_004	- -	ProcessPurchaseOrder (O)
PROCESS_INVOICE_002	- -	CancelPurchaseOrder (O)

7.2	7.2.1	10.1
SHOW_SALESORDER_006	- -	- -
SHOW_SHIPMENT_005	- -	- -
SYNC_CATALOG_003	- -	- -

Where:

  - (I): Input
  - (O): Output

- Inbound and outbound documents.
- Functional acknowledgment (FA) and document validation against the schema definition.

Custom XML Support

B2B for Oracle Integration supports custom XML documents.

You can perform the following actions:

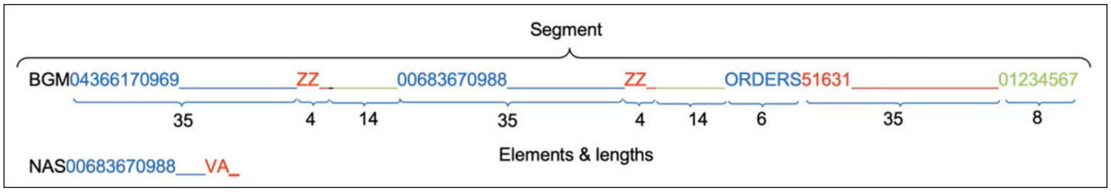
- Upload and edit custom XML schemas.
- Create an XML schema as a separate document type.
- Create the XML file schema by importing an XSD file.
  - The XSD can be within a ZIP file and support XML made up of multiple XSDs.
  - You can provide a document version and transaction type value.
- Use the schema to create the B2B document.
  - Define the root element for the XML document.
  - Define elements from the XML as business identifiers.
- Choose that XML document in their trading partner agreements.
- Use the document as part of B2B data transformations (mappings) and process it as per the developed integration.
- Use basic XML validations. B2B message tracking lets you visualize the processed XML transactions, filter on the document type, view the payload, and perform normal observability actions as other document types.
- Download the schema from the user interface.

Positional Flat or Fixed Length Payloads Support

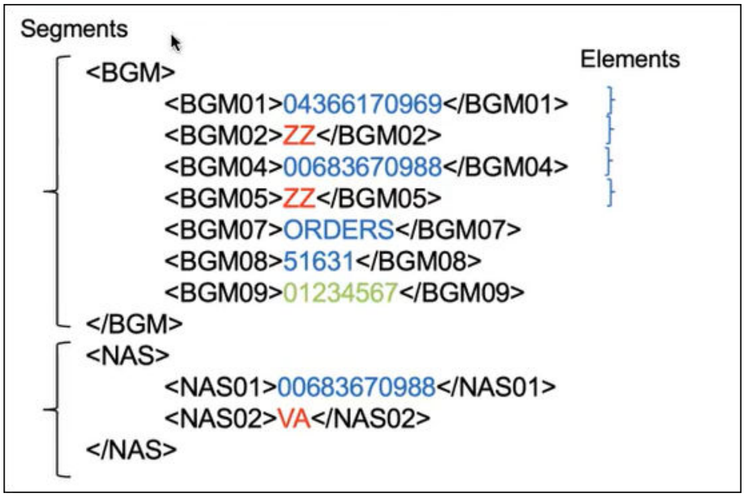
B2B for Oracle Integration can consume positional flat or fixed length payloads (also called data files in native format) for inbound and outbound message translations. These files can be of fixed length for various fields, CSV, delimited, and other formats. B2B for Oracle Integration enables you to define a B2B schema and document from these various formats and include the document in a trading partner agreement.

The following example describes how B2B for Oracle Integration handles a positional flat payload.

A positional flat payload includes segments (for this example, `BGM` and `NAS`). Each segment includes a row that consists of multiple elements. Each element is defined by its length. For example, the `BGM` segment includes the element `04366170969` that can be a length of up to 35 characters, the element `ZZ` that can be a length of up to 4 characters, and so on.


Segment
BGM04366170969_____________ZZ_____________00683670988_____________ZZ_____________ORDERS51631_____________01234567
35
4
14
35
4
14
6
35
8
Elements & lengths
NAS00683670988___VA_

After translation by the B2B action in an integration, the XML format looks as follows. Each segment consists of child elements corresponding to a unique row in the payload.



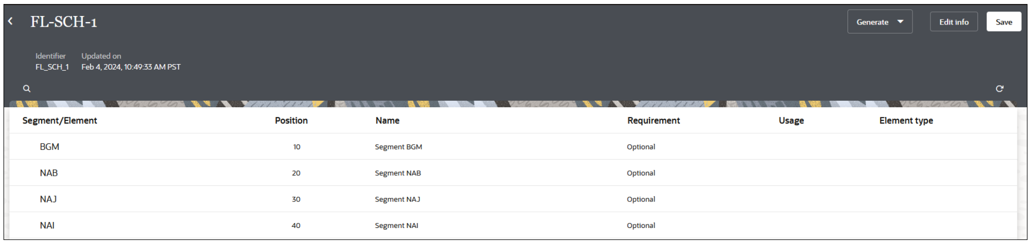
```
<BGM>
        <BGM01>04366170969</BGM01>
        <BGM02>ZZ</BGM02>
        <BGM04>00683670988</BGM04>
        <BGM05>ZZ</BGM05>
        <BGM07>ORDERS</BGM07>
        <BGM08>51631</BGM08>
        <BGM09>01234567</BGM09>
</BGM>
<NAS>
        <NAS01>00683670988</NAS01>
        <NAS02>VA</NAS02>
</NAS>
```

Let's look at another example. Assume your positional flat payload file consists of the following segments and rows.

```
BGM94366179969                         ZZ
006836700988                       ZZ                 ORDERS51631
NAB568                92 Ritorazione e XXXXXXX Per XXXXXXX Srl presso Scuola
XXXXXXX XXXXXXP.zza Vittorio Veneto 1 XXXXXXXX MI,
NAJ568                92 Ritorazione e XXXXXXX Per XXXXXXX Srl presso Scuola
XXXXXXX XXXXXXP.zza Vittorio Veneto 1 XXXXXXXX MI,
NAI04366170969        VA Ritorazione e XXXXXXX Per XXXXXXX
Srl                             Via Roma, 19
```

You drag the positional flat payload file into the user interface when creating a B2B schema and specify the number of characters to scan to identify the segments. For this example, `3` is specified to create the schema segments from the first three characters in each row.

From this payload, four segments are created with default position and requirement options.



Identifier: FL_SCH_1
Updated on: Feb 4, 2024, 10:49:33 AM PST

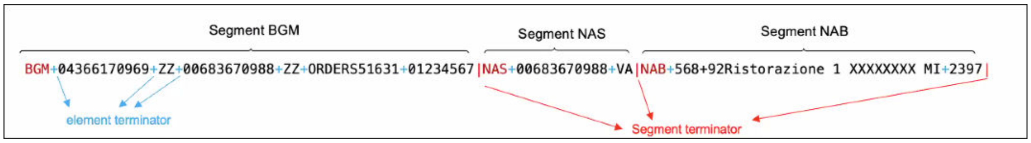
Segment/Element	Position	Name	Requirement	Usage	Element type
BGM	10	Segment BGM	Optional		
NAB	20	Segment NAB	Optional		
NAJ	30	Segment NAJ	Optional		
NAI	40	Segment NAI	Optional		

You can select **Actions** ▪ ▪ ▪ to add child elements, new child elements, additional segments, and loops.

Delimited Payload Support

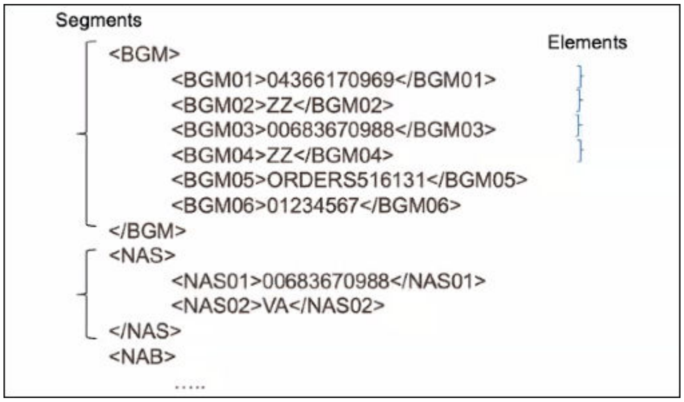
B2B for Oracle Integration supports delimited payloads in inbound and outbound transactions. You create a schema and document from a delimited payload to use in integrations. During runtime, the delimited payload is translated to XML (and vice versa) based on the schema you created. Delimited payload support is provided in both standalone and trading partner modes.

The following example describes how B2B for Oracle Integration handles a flat delimited payload. You must provide segment and element terminators in your delimited payload to separate the data and successfully create a schema on the Schemas page in B2B for Oracle Integration. For this example, segments are identified by a pipe (|) terminator and elements within each segment are identified by a plus sign (+) terminator.



```
BGM+04366170969+ZZ+00683670988+ZZ+ORDERS51631+01234567|NAS+00683670988+VA|NAB+568+92Ristorazione 1 XXXXXXXX MI+2397|
```

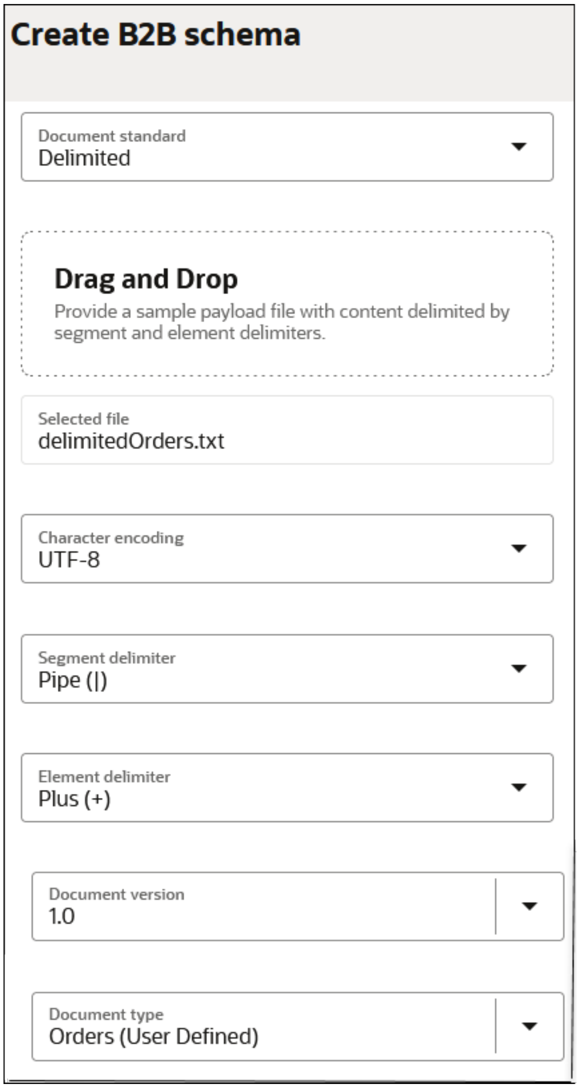
The corresponding translated XML in B2B for Oracle Integration at runtime looks as follows. Three segments (`BGM`, `NAS`, and `NAB`) are created, each with their individual elements.



```
<BGM>
        <BGM01>04366170969</BGM01>
        <BGM02>ZZ</BGM02>
        <BGM03>00683670988</BGM03>
        <BGM04>ZZ</BGM04>
        <BGM05>ORDERS516131</BGM05>
        <BGM06>01234567</BGM06>
</BGM>
<NAS>
        <NAS01>00683670988</NAS01>
        <NAS02>VA</NAS02>
</NAS>
<NAB>
        .....
```

When you create the schema on the Schemas page, you specify the following key details:

- Select **Delimited** as the document standard.
- Drag and drop the delimited payload.
- Select the segment and element terminators (delimiters) that you specified in the delimited payload.
- Manually specify values for the document version and type or select existing values.



Create B2B schema

Document standard: Delimited

Drag and Drop
Provide a sample payload file with content delimited by segment and element delimiters.

Selected file: delimitedOrders.txt

Character encoding: UTF-8

Segment delimiter: Pipe (|)

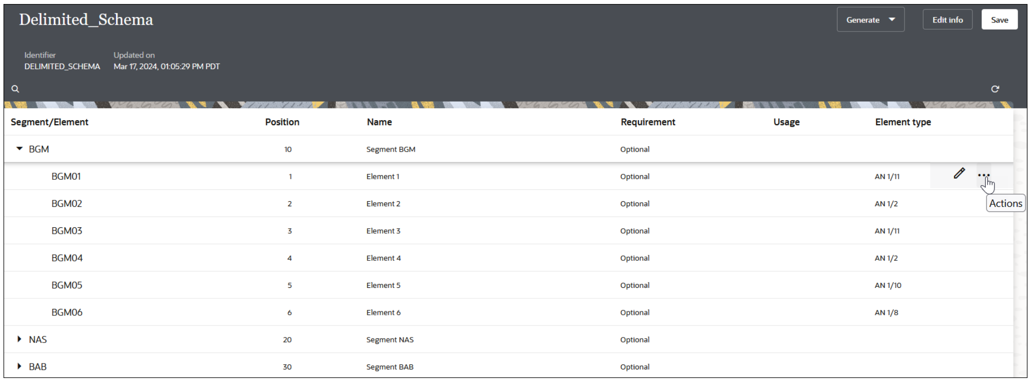
Element delimiter: Plus (+)

Document version: 1.0

Document type: Orders (User Defined)

> **Note:**
>
> The segment and element delimiters cannot be the same value.

This creates a schema consisting of the segments and elements from the delimited payload. You can further customize the schema from the **Actions** menu by adding new segments, adding new elements, defining loops, and more. You can also view the defined terminators for the schema by selecting **Edit info** at the top of the page.



Identifier: DELIMITED_SCHEMA
Updated on: Mar 17, 2024, 01:05:29 PM PDT

Segment/Element	Position	Name	Requirement	Usage	Element type
BGM	10	Segment BGM	Optional		
BGM01	1	Element 1	Optional		AN 1/11
BGM02	2	Element 2	Optional		AN 1/2
BGM03	3	Element 3	Optional		AN 1/11
BGM04	4	Element 4	Optional		AN 1/2
BGM05	5	Element 5	Optional		AN 1/10
BGM06	6	Element 6	Optional		AN 1/8
NAS	20	Segment NAS	Optional		
BAB	30	Segment BAB	Optional		

After schema definition is complete, you define a document from that schema to use in integrations on the Documents page, including:

- Defining identifiers for document identifications at runtime.
- Defining XPath expressions for business identifiers in the document.

Two Patterns For Using B2B for Oracle Integration

B2B for Oracle Integration can be used in two different ways. This section describes these two patterns for processing B2B messages.

- Standalone mode:
  Uses the basic building blocks such as the AS2, FTP, SOAP, or REST Adapters along with the B2B action to process B2B messages. This is the same as any other integration. You build the B2B integrations yourself and track the processing of B2B messages as integration instances on the Instances page. In this mode, the B2B trading partner definitions are not used.

  See Design a B2B Standalone Mode Integration.
- B2B trading partner mode:
  Extends standalone mode and provides a fully declarative and configuration-driven setup with trading partner management and a specialized B2B message tracking user interface (the Track B2B messages page). You define trading partner profiles at design-time and use them during runtime. You only create integrations to interface with your backend applications. Behind the scenes, however, message processing still occurs through integrations. The integrations are automatically created for you using templates available with B2B for Oracle Integration. In short, this mode provides you with a ready-to-use solution that you simply configure and use.

The differences between these two modes are as follows. Overall, the B2B trading partner mode is the preferred pattern to use because it is fully-declarative and simpler to use. This mode was introduced recently when compared to the standalone mode.

Feature	Standalone Mode	B2B Trading Partner Mode
Processing basic B2B messages, including EDI processing	Available. You design and build the B2B integrations using any technology adapters and actions available in Oracle Integration. For instance, you use the AS2, FTP, REST, or SOAP Adapter to interface with your trading partner. To integrate with backend (or back-office) applications, you also use any available application adapters, such as Oracle ERP Cloud Adapter or Oracle NetSuite Adapter.	Available. You start by defining trading partners and other objects in a specialized B2B user interface. The B2B integrations are automatically created for you using a transport type you choose (for example, AS2 or FTP). You separately create the backend (or back-office) integrations using any available application adapters and link to them inside the trading partner profile.
Tracking B2B message activity	Limited capability. B2B message traffic is tracked solely as integration instances. Each incoming or outgoing message is one integration instance. You use the Instances page to view B2B messages. Tracking and troubleshooting requires deeper skills, likely requiring knowledge about how the integrations were built.	Specialized tracking is available. In this mode, B2B messages are persisted separately and a specialized B2B tracking user interface (the Track B2B messages page) is provided to view them. This page is less technical and designed for use by B2B operations or help desk staff for routine tracking and troubleshooting.
On-boarding new trading partners	To on-board a new trading partner, you manually create and activate additional integrations to receive and send messages from the new trading partner. Design your integrations to support document-based routing. This mode does not use B2B trading partner definitions.	To on-board a new trading partner, you add a trading partner definition in the user interface and input additional configuration. This automatically creates additional integrations to receive and send messages from the new partner. Document-based routing is defined as part of trading partner definitions.
Using any adapter to interface with a trading partner	Because the standalone mode only provides basic build blocks, it gives you the power to design the B2B integration any way you want using any technology adapter available in Oracle Integration. For example, you can use the AS2, FTP, REST, or SOAP Adapter to communicate with a trading partner even though some adapters do not currently provide a B2B trading partner mode.	At this time, only the AS2, RosettaNet, FTP Adapters support the B2B trading partner mode. For example, you cannot currently use the REST or SOAP protocols in this mode.

If you initially built integrations in standalone mode, you can continue to use them as-is. You cannot mix and match the two modes; they work separately and their input/output schemas are different.

More details about each pattern are provided in subsequent chapters.

2 Use B2B for Oracle Integration in Trading Partner Mode

This chapter describes the concepts and tasks for using B2B for Oracle Integration in trading partner mode.

Topics:

- Trading Partner Mode Concepts
- Manage B2B Trading Partners
- Create B2B Integrations for Receiving and Sending
- Create Backend Integrations
- Inbound Message Processing
- Outbound Message Processing
- Use the B2B Action In Trading Partner Mode

Trading Partner Mode Concepts

This section describes trading partner mode concepts in B2B for Oracle Integration.

Topics:

- Host Company
- Trading Partners
- Integrations Used for B2B Message Processing
- Message Routing Between Integrations
- Transports for Communication
- Agreements
- B2B Documents and Schemas
- Message Persistence and Tracking

Host Company

There are two parties involved in any document exchange in B2B for Oracle Integration: your company (the host company) and an external trading partner.

1. In the navigation pane, click **B2B**, then **Host profile**.
2. View the configuration page you must complete on behalf of your company.
   You only need to enter B2B identifiers for the host profile. You enter your company's identity information such as the EDI Interchange ID and the AS2 Identifier. Other types of configuration, such as signing certificates for your company, are entered later.

When you send electronic documents to an external trading partner, some of the B2B identifiers you enter in the host profile are inserted into the message. This enables the recipient party to identify your company as the sender of the message.

See Define the Host Profile.

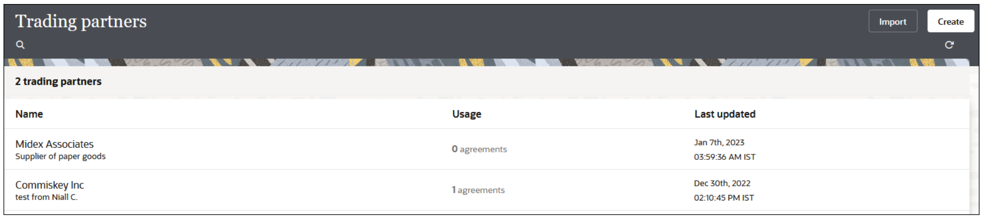
Trading Partners

A trading partner is the external business entity with which your company interacts to send or receive business documents, such as orders and invoices, in electronic form.

1. In the navigation pane, click **B2B**, then **Trading partners**.
2. View the contents.
   You can access pages to register all your external trading partners and enter information on their behalf from this location. The trading partners cannot access these pages in your Oracle Integration instance. Therefore, as the B2B system administrator, you gather information offline from your external trading partners and enter the information in these pages.

   Similar to B2B for Oracle Integration, your external trading partner has a B2B application of their own. Your trading partners collect information offline about your company and enter it into their B2B application, which you cannot directly access.

   Once setup is completed and tested on both ends, the two parties (your company and the external trading partner) are ready to send and receive documents.

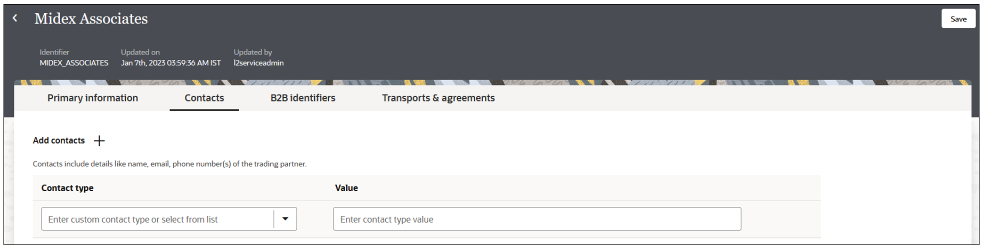


2 trading partners

Name	Usage	Last updated
Midex Associates Supplier of paper goods	0 agreements	Jan 7th, 2023 03:59:36 AM IST
Commiskey Inc test from Niall C.	1 agreements	Dec 30th, 2022 02:10:45 PM IST

   You create and register an external trading partner by entering information in the following tabs, each of which are explained in subsequent sections.

   - **Primary information**
   - **Contacts**
   - **B2B identifiers**
   - **Transports & agreements**



Identifier MIDEX_ASSOCIATES
Updated on Jan 7th, 2023 03:59:36 AM IST
Updated by l2serviceadmin

Contacts include details like name, email, phone number(s) of the trading partner.

Contact type	Value
Enter custom contact type or select from list	Enter contact type value

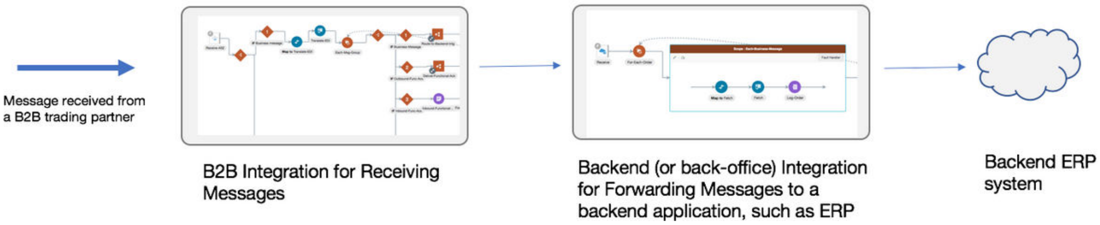
See Create Trading Partners.

Integrations Used for B2B Message Processing

The actual processing of B2B messages occurs in integrations. B2B for Oracle Integration uses a two-integration design pattern for better modularity.

Inbound Message

A B2B message received from an external trading partner is called an inbound message. The following diagram shows how an inbound message is processed through two integrations.


Message received from a B2B trading partner
B2B Integration for Receiving Messages
Backend (or back-office) Integration for Forwarding Messages to a backend application, such as ERP
Backend ERP system

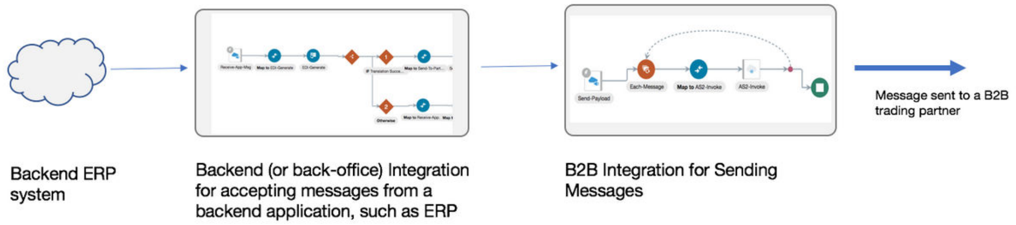
In the two-integration pattern, the B2B integration for receiving messages performs the following steps:

1. Receives the message from the trading partner and performs various validity checks, for example:
   - Is the message received from a known (that is, registered) trading partner? Verify this based on the authentication credentials, SSL certificates, HTTP headers, and so on.
   - Is the message signed or encrypted? If so, verify the signature and decrypt the message. This step is called unpackaging, which is similar to removing an object from its packaging.
2. Sends a transport level acknowledgment back to the trading partner, if asked by the trading partner.
3. Detects the type of payload. If it is a payload that requires translation (for example, an EDI message), parse and translate the message.
4. Sends a translator level acknowledgment, if configured.
5. The backend integration converts the message into a format that a backend application, such as ERP, can directly consume (for example, XML, JSON, CSV, and so on) and forwards the message to a backend system for further processing.

Outbound Message

A B2B message sent to an external trading partner is called as an outbound message.

The following diagram shows how an outbound message is processed through two integrations.



In the two-integration pattern, the backend integration performs these steps:

1. Receives an event from a backend application such as ERP for a business document that must be sent to an external trading partner.
2. Translates the message to an industry-standard B2B format (for example, Electronic Data Interchange (EDI) format).

The B2B integration for sending messages performs these steps:

1. Adds headers, encrypts, signs, and compresses the payload, as per your required configuration. This step is called packaging, which is similar to wrapping an object into an envelope and making it ready for delivery.
2. Transmits the message to the external trading partner's endpoint.
3. If the trading partner responds with a transport level acknowledgment, it updates the status of the transmitted message accordingly.

The following table summarizes the two-integration pattern for B2B message processing:

Concept	B2B Integrations for Receiving and Sending Messages	Backend Integration
Purpose	Handles the low-level technical interaction with an external trading partner, including B2B layer acknowledgments. This is provided by Oracle.	Handles the complete interaction with your backend application, such as ERP, and the message transformation between B2B canonical format and the backend application format. This is developed by your company and is typically specific to your backend application. There may be recipes and/or accelerators available for specific backend applications to get you started. See Get Started with Integration Recipes and Accelerators in *Getting Started with Oracle Integration Generation 2*.

Concept	B2B Integrations for Receiving and Sending Messages	Backend Integration
Creation	These integrations are automatically created for each trading partner from a template. Details about how to automatically create integrations are provided. See Create B2B Integrations for Receiving and Sending.	You create this integration manually and use any available application adapters to interface with your backend application. Details about how to build this integration are provided. See Create Backend Integrations.
What does it do	• Handles packaging/ unpackaging of a message and performs receipt or transmission from or to a trading partner using an agreed-upon protocol. • Performs handshakes with the trading partner's system, generating or consuming acknowledgments as needed by a typical B2B gateway. For an inbound message, the B2B integration for receiving messages also translates the message (for example, from EDI to XML). If the inbound message contains batched transactions, those are split individually before handing them to the backend integration. However, for an outbound message, the translation is not performed in the B2B integration for sending messages. (It is performed in the backend integration.)	Handles interfacing with a backend application, such as ERP, both for the inbound and outbound sides. For an inbound message, the backend integration gets a B2B message that is already translated to a canonical format. It must convert it further and forward it to your backend application. For an outbound message, the backend integration is the entry point where an event notification from a backend application triggers this integration to perform a translation to a B2B format (for example, XML to EDI) and call the B2B integration for sending messages.

Concept	B2B Integrations for Receiving and Sending Messages	Backend Integration
How many integrations are needed	A pair of integrations are required per B2B trading partner and transport. For example, if you have 10 B2B trading partners, each with one transport, then 20 integrations are needed (one each for receiving and sending messages). These integrations are automatically created for you. See Trading Partner Mode Concepts.	A pair of integrations are required per business document type. For example, if you have three types of business documents (purchase orders, invoices, and shipment notifications), you need six integrations (three on the inbound side and three on the outbound side), regardless of the number of trading partners. Typically, you share these integrations across many trading partners as long as they use the same document format. You may actually need fewer than six integrations if some types of business documents are only used in one direction. For example, if purchase orders are only received, but never sent, you need only one backend integration for inbound purchase orders and none for outbound purchase orders. You may also need fewer integrations if you design the integrations to handle multiple document types. This is because in the simplest case, the assumption is that one backend integration handles only one type of business document. You need to create these integrations manually. See Create Backend Integrations.

Message Routing Between Integrations

The B2B integrations and backend integrations are not hard-wired to each other. They are instead dynamically wired to follow routing rules that you add declaratively. This section describes how the routing rules work using the concept of inbound and outbound agreements.

Inbound

Assume one of your trading partners is sending multiple types of documents: purchase orders and invoices. For this trading partner, one B2B integration receives messages, and it receives both types of documents. You define routing rules by adding two inbound agreements to this trading partner. When you add an inbound agreement, you must select one B2B document and one backend integration.

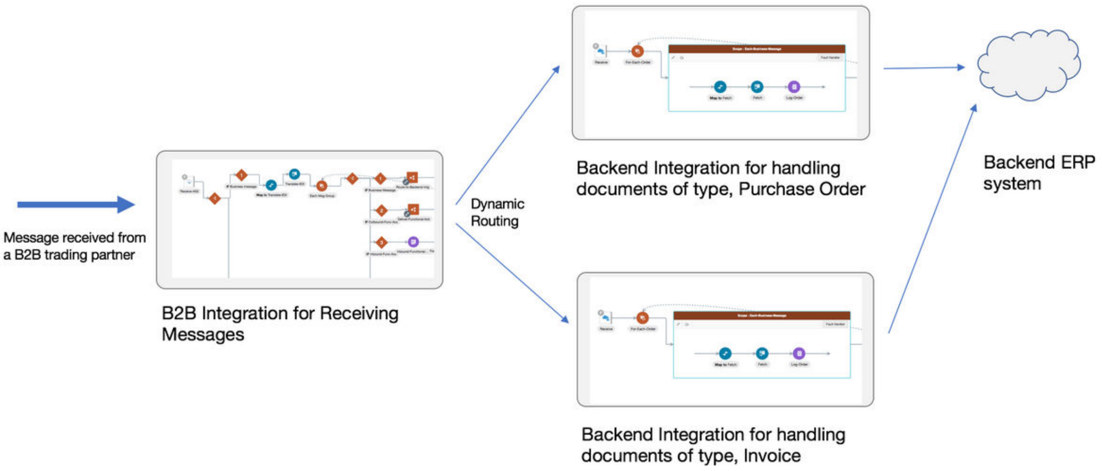
You add the following two inbound agreements:

- Inbound agreement 1 for selecting a purchase order type document and a backend integration called X.
- Inbound agreement 2 for selecting an invoice type document and a backend integration called Y.

These two agreements express the following intentions to the system:

- When a purchase order type document is received from this trading partner, it's routed to backend integration X.
- When an invoice type document is received from this trading partner, it's routed to backend integration Y.

While inbound agreements do more than routing, the routing is a primary concept. The B2B integration for receiving messages honors the routing rules at runtime and dynamically invokes backend integration X or Y, depending on the identified document type, as shown below.


Message received from a B2B trading partner
B2B Integration for Receiving Messages
Dynamic Routing
Backend Integration for handling documents of type, Purchase Order
Backend Integration for handling documents of type, Invoice
Backend ERP system

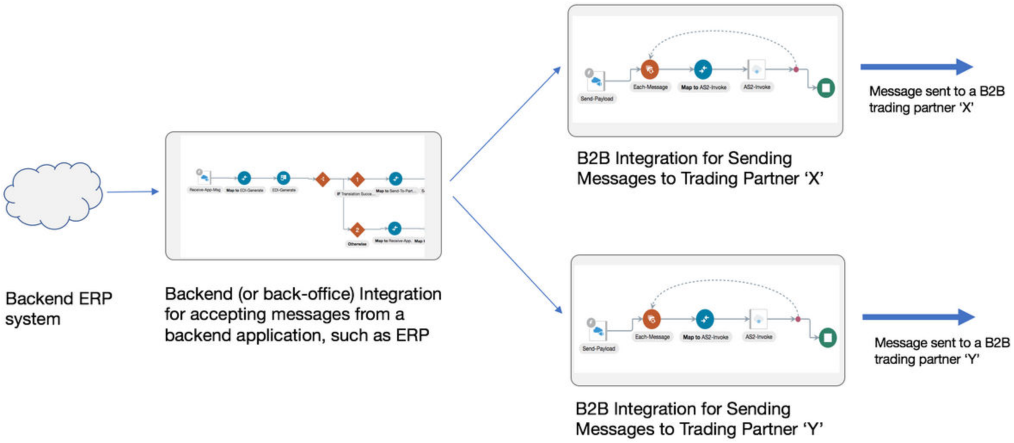
Note:

You can create a single, generic backend integration that handles multiple types of documents, if you think there is enough commonality and it is easier for maintenance. In this case, select the same backend integration in the two inbound agreements.

Outbound

Assume you have two trading partners and want to route a purchase order document to one of those trading partners. In this case, the backend integration must make a routing choice to select the correct B2B integration for sending messages between two integrations.

In the outbound case, inside your backend integration, you must specify the trading partner to which you want to deliver the message. This is specified by providing a trading partner ID or an application partner ID. See Outbound Message Processing.


Send-Payload
Each-Message
Map to AS2-Invoke
AS2-Invoke
Message sent to a B2B trading partner 'X'
B2B Integration for Sending Messages to Trading Partner 'X'
Backend ERP system
Backend (or back-office) Integration for accepting messages from a backend application, such as ERP
Send-Payload
Each-Message
Map to AS2-Invoke
AS2-Invoke
Message sent to a B2B trading partner 'Y'
B2B Integration for Sending Messages to Trading Partner 'Y'

You define routing rules by adding two outbound agreements, one per trading partner. When you add an outbound agreement, you must select one B2B document and one transport. See Transports for Communication. Be aware for now that a transport maps to B2B integrations for receiving and sending messages for a given trading partner.

You add the following two outbound agreements:

- Outbound agreement 1 for selecting a purchase order type document and transport X for trading partner X.
- Outbound agreement 2 for selecting a purchase order type document and transport Y for trading partner Y.

These two agreements express the intention to the system that when a purchase order type document is sent to trading partner X, route it to the B2B integration for sending messages X. When the document is sent to trading partner Y, route it to the B2B integration for sending messages Y.

Outbound agreements do more than routing, but this section only focuses on the routing aspect.

Further concepts about agreements are provided. See Agreements.

Details are provided for creating agreements. See Create Agreements.

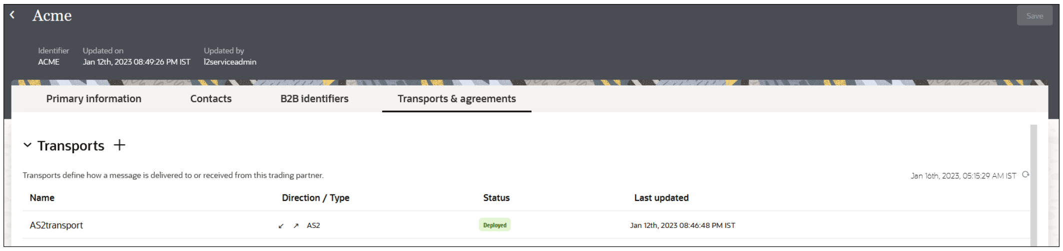
> **Note:**
>
> In the outbound case also, you can create a common backend integration that handles multiple types of documents, if that is your design preference.

Transports for Communication

Transports are configuration objects that represent a concrete communication channel to a trading partner using a specific protocol such as AS2 or FTP. You add one or more transports to a trading partner to send or receive business documents from the partner.

AS2 and FTP (which includes SFTP) are the currently supported protocols for B2B trading partner mode. If you want to use another protocol adapter in B2B for Oracle Integration, you can do so only using the standalone mode.

1. In the navigation pane, click **B2B**, then **Trading partners**.
2. Click the **Transports & agreements** tab.
   The following example shows a trading partner with an AS2 transport.



Identifier: ACME
Updated on: Jan 12th, 2023 08:49:26 PM IST
Updated by: l2serviceadmin

Transports define how a message is delivered to or received from this trading partner.

Jan 16th, 2023, 05:15:29 AM IST

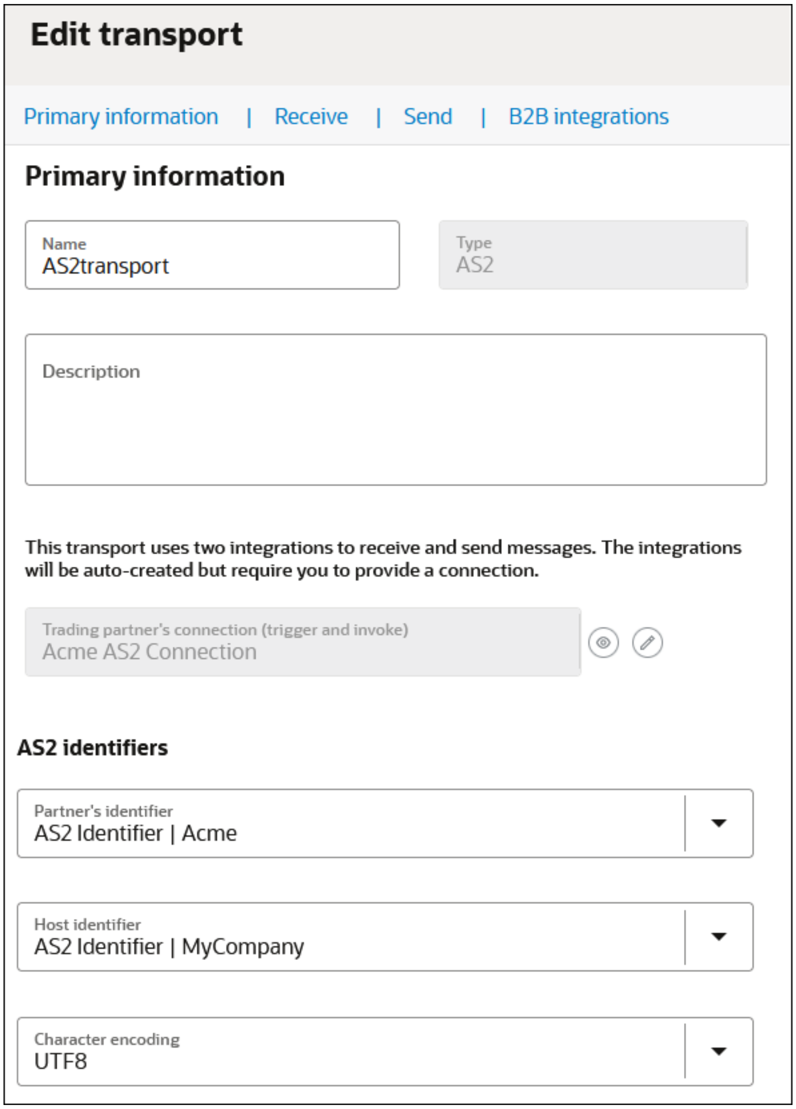
Name	Direction / Type	Status	Last updated
AS2transport	↙ ↗ AS2	Deployed	Jan 12th, 2023 08:46:48 PM IST

A transport requires the following entities to work:

- A connection that you create and configure to point to the external trading partner's endpoint. As part of this configuration, you provide the host name, port, URL, username/password credentials, and other details that your trading partner provides to you.
- Additional configurations (for example, a directory path for FTP or an encryption algorithm for AS2).
- A pair of integrations, one for receiving messages and another for sending messages. These integrations are automatically created for you. See Integrations Used For B2B Processing.

You enter the following configuration details when you define a transport (for this example, AS2).

- **Primary Information** tab:



Edit transport

Primary information

Name: AS2transport

Type: AS2

Description

This transport uses two integrations to receive and send messages. The integrations will be auto-created but require you to provide a connection.

Trading partner's connection (trigger and invoke): Acme AS2 Connection

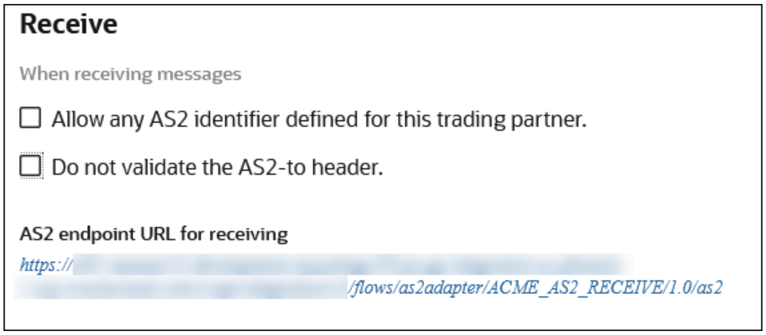
AS2 identifiers

Partner's identifier: AS2 Identifier | Acme

Host identifier: AS2 Identifier | MyCompany

Character encoding: UTF8

- **Receive** tab:



Receive

When receiving messages

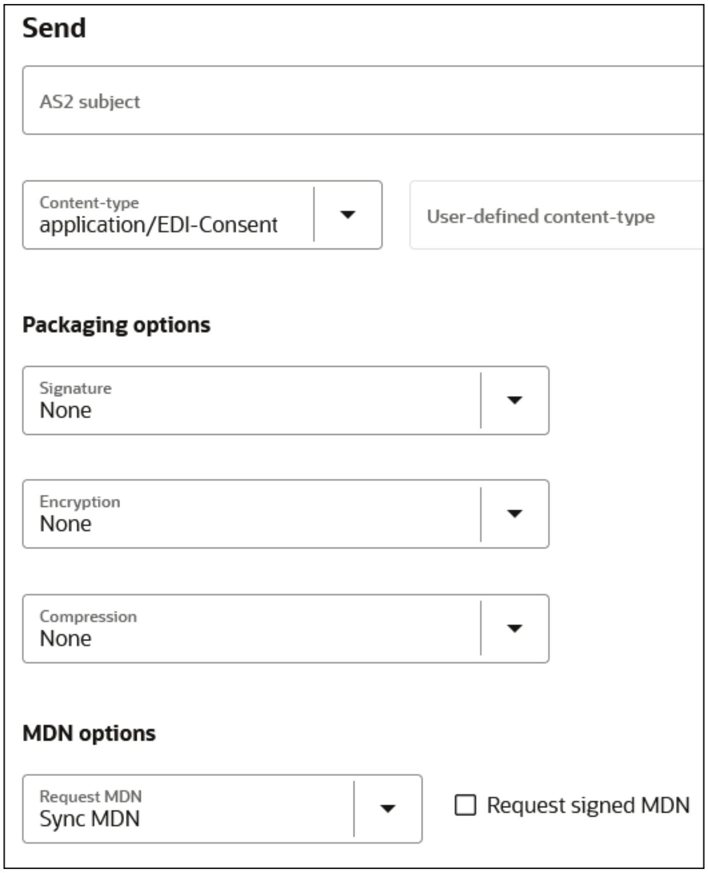
☐ Allow any AS2 identifier defined for this trading partner.

☐ Do not validate the AS2-to header.

AS2 endpoint URL for receiving

https://[redacted]/flows/as2adapter/ACME_AS2_RECEIVE/1.0/as2

- **Send** tab:



Send

AS2 subject

Content-type: application/EDI-Consent

User-defined content-type

Packaging options

Signature: None

Encryption: None

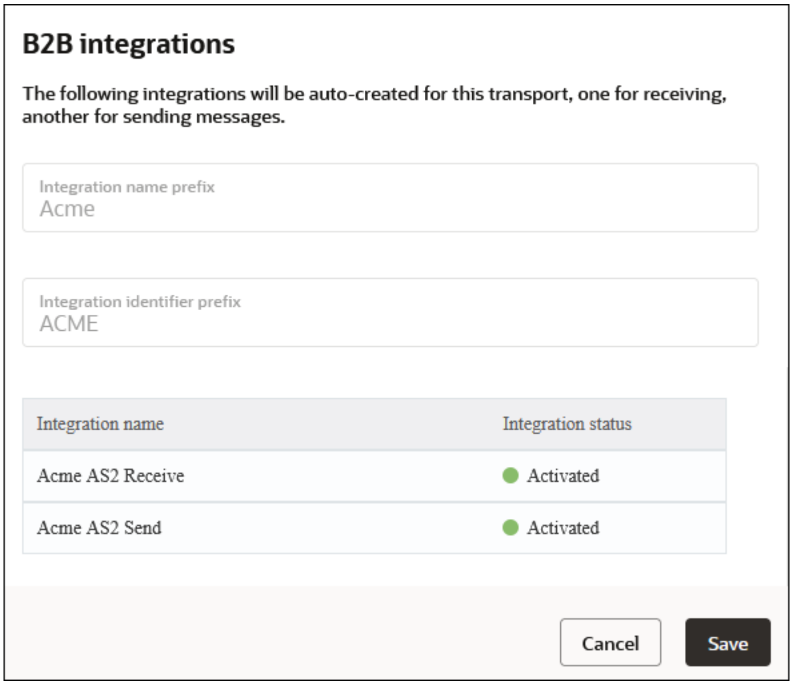
Compression: None

MDN options

Request MDN: Sync MDN

□ Request signed MDN

- **B2B integrations** tab:



B2B integrations

The following integrations will be auto-created for this transport, one for receiving, another for sending messages.

Integration name prefix: Acme

Integration identifier prefix: ACME

Integration name	Integration status
Acme AS2 Receive	● Activated
Acme AS2 Send	● Activated

The transport life cycle consists of the following actions:

- Create a transport: Adds a definition for the transport. The pair of B2B integrations are automatically created and permanently linked to this transport.
- Deploy a transport: Makes the transport visible for runtime processing and also activates the B2B integrations. This transport can now receive and send messages.
- Redeploy a transport: Applies configuration changes to the runtime on-the-fly without disrupting message processing (with some restrictions).
- Undeploy a transport: Hides the transport from runtime processing and also deactivates the B2B integrations. This transport can no longer receive and send messages.
- Delete a transport: Removes the definition and also deletes the B2B integrations.

See Define Transports.

AS2

Applicability Statement 2 (AS2) is an HTTP-based protocol designed for B2B that adds a comprehensive set of data security features around data confidentiality, data integrity/authenticity, and nonrepudiation.

The AS2 specification is covered by RFC 4130. B2B for Oracle Integration supports AS2 versions 1.0 and 1.1.

An AS2 transport offers configuration options specific to AS2 that work in conjunction with the AS2 connection and the certificate management user interface.

For example, if you want to sign and encrypt the outbound messages:

1. Use the Oracle Integration certificate management capabilities by uploading your certificate. In the navigation pane, click **Settings**, then **Certificates**..
2. Enter the signing and encryption certificate alias in the AS2 Adapter connection selected in the AS2 transport.
3. Select an encryption and signing algorithm in the AS2 transport configuration.

The simplest AS2 communication uses no encryption, signing, and compression. If you are learning about AS2, you can start simple and add the security layers later.

There is the concept of an electronic read receipt in AS2, officially known as Message Disposition Notification (MDN). It is a transport level acknowledgment used as a confirmation that the other party has received your message intact. B2B for Oracle Integration generates and consumes MDN messages (when enabled) and correlates them to the original transmissions. The Track B2B Messages page, described later, enables you to view the AS2 messages and MDN acknowledgments.

More details about the AS2 transport are provided. See Define Transports.

FTP

File Transfer Protocol (FTP) and Secure File Transfer Protocol (SFTP) are commonly used for B2B communications. An FTP transport also works in conjunction with an FTP Adapter connection.

In an FTP Adapter connection, you specify the hostname, port, credentials, and other security configurations. In the FTP transport, you enter the input and output directories, file name pattern, and other details.

One important aspect of an FTP transport is that the receiving side polls the input directory on a time-based schedule. You can set up the schedule by selecting the **Receive schedule** option of the actions menu for an FTP transport.

More details about the FTP transport are provided. See Define Transports.

RosettaNet

RosettaNet is a global standard for electronic B2B communication and collaboration in the supply chain management industry. RosettaNet includes a set of standardized message formats, business processes, and implementation frameworks to facilitate electronic data exchange and collaboration between trading partners.

The RosettaNet standards are described in GS1 US RosettaNet Standards. B2B for Oracle Integration supports all PIPs.

RosettaNet supports nonrepudiation. This feature enables B2B for Oracle Integration to verify the integrity and origin of a RosettaNet 2.0 business message that is transmitted and received from a trading partner. This includes nonrepudiation of origin and content and nonrepudiation of receipt. No trading partner can deny receiving the message. The Track B2B messages page, described later, enables you to view the RosettaNet messages.

More details about the RosettaNet transport are provided. See Define Transports.

REST

You can exchange inbound and outbound messages with trading partners over the REST transport in B2B for Oracle Integration. The REST transport is useful for trading partners that don't use EDI or RosettaNet and can exchange messages using web forms or WebUI.

More details about the REST transport are provided. See Define Transports.

Agreements

You define one or more agreements for a B2B trading partner with an intent to send or receive only certain types of business documents to or from that trading partner. An agreement literally means that your company and the external trading partner have formally agreed upon the terms for the exchange of specific business documents.

An agreement has the following purposes:

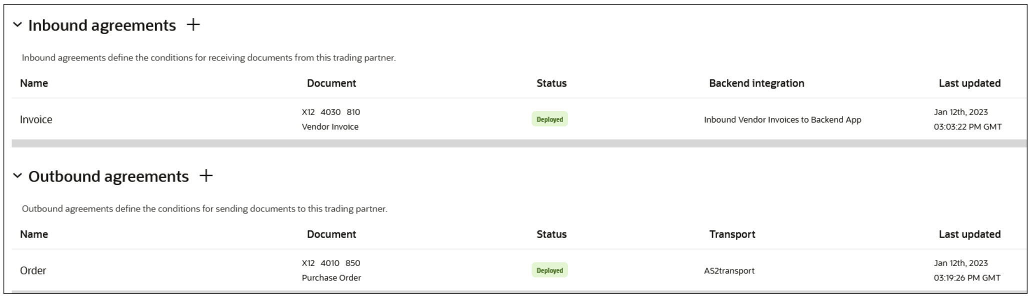
- Armed with the knowledge of which documents to expect from a given trading partner, B2B only accepts one of the agreed-upon documents. Any unexpected document type is rejected both while receiving from or sending to that trading partner.
- Defines the data format for the documents exchanged. For example, for an EDI document, syntax validations and parsing is done based on the B2B document selected in an agreement. Both parties, your company and the external trading partner, must decide in advance on the data format to use for interoperability to work. One of the parties typically shares an implementation guide containing the data format definition and the other party complies with it and creates an equivalent data format definition on their side.
- Defines behavior for message processing. For example, syntax validations on the data format can be turned on or off in an agreement, among other settings.
- Defines rules for the routing of documents. For example, when receiving documents, an inbound agreement defines the backend integration to which to route a document, based

on its type. While sending documents, an outbound agreement defines which B2B sending integration to hand over a specific document for delivery to a trading partner.

The agreement lifecycle consists of the following actions:

- Create an agreement: Adds the definition in the design-time only (but unless deployed, the new agreement is not enforced at runtime).
- Deploy and redeploy an agreement: Makes the agreement visible for runtime processing and is immediately enforced.
- Undeploy an agreement: Hides the agreement from runtime processing, making it no longer effective, starting immediately.
- Delete an agreement: Removes it from design time.

The following example shows inbound and outbound agreements defined for a trading partner:



Inbound agreements

Inbound agreements define the conditions for receiving documents from this trading partner.

Name	Document	Status	Backend integration	Last updated
Invoice	X12 4030 810 Vendor Invoice	Deployed	Inbound Vendor Invoices to Backend App	Jan 12th, 2023 03:03:22 PM GMT

Outbound agreements

Outbound agreements define the conditions for sending documents to this trading partner.

Name	Document	Status	Transport	Last updated
Order	X12 4010 850 Purchase Order	Deployed	AS2transport	Jan 12th, 2023 03:19:26 PM GMT

See Create Agreements.

B2B Documents and Schemas

A B2B document is a mandatory object required by an agreement that specifies a data format and some additional configuration pertaining to the data format. A data format is specified using the following properties.

- Document standard: The X12, X12 HIPPA, EDIFACT EDI, RosettaNet, OAG, fixed length, and custom standards are currently supported.
- Document version: A version as defined by the standards body.
- Document type: A type as defined by the standards body.
- Document schema: A standard or customized variant defined in a B2B schema object. Standard means the pristine schema defined in the data dictionary published by the standards body.

Additional configuration enables one or more business identifiers for the B2B runtime to be extracted and displayed in the Track B2B messages page.

1. In the navigation pane, click **Observability**, then **B2B tracking**.
2. Select **Business messages**.

Additional details about B2B documents and B2B schemas are provided. See B2B Documents and schemas.

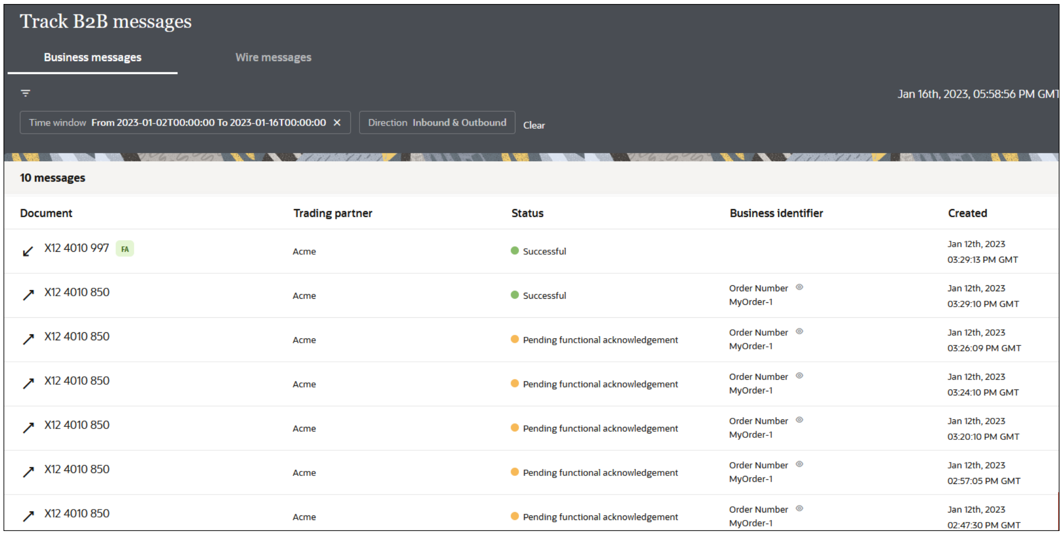
Additional details about agreements are provided. See Create Agreements.

Message Persistence and Tracking

As messages flow through the B2B integrations for receiving and sending messages, each inbound and outbound message is persisted separately, in addition to the usual integration instance tracking.

Persisted B2B messages can be viewed from a specialized B2B tracking page.

1. In the navigation pane, click **Observability**, then **B2B tracking**.



Track B2B messages

Jan 16th, 2023, 05:58:56 PM GMT

Time window From 2023-01-02T00:00:00 To 2023-01-16T00:00:00 ✕ | Direction Inbound & Outbound | Clear

10 messages

Document	Trading partner	Status	Business identifier	Created
↙ X12 4010 997 FA	Acme	● Successful		Jan 12th, 2023 03:29:13 PM GMT
↗ X12 4010 850	Acme	● Successful	Order Number MyOrder-1	Jan 12th, 2023 03:29:10 PM GMT
↗ X12 4010 850	Acme	● Pending functional acknowledgement	Order Number MyOrder-1	Jan 12th, 2023 03:26:09 PM GMT
↗ X12 4010 850	Acme	● Pending functional acknowledgement	Order Number MyOrder-1	Jan 12th, 2023 03:24:10 PM GMT
↗ X12 4010 850	Acme	● Pending functional acknowledgement	Order Number MyOrder-1	Jan 12th, 2023 03:20:10 PM GMT
↗ X12 4010 850	Acme	● Pending functional acknowledgement	Order Number MyOrder-1	Jan 12th, 2023 02:57:05 PM GMT
↗ X12 4010 850	Acme	● Pending functional acknowledgement	Order Number MyOrder-1	Jan 12th, 2023 02:47:30 PM GMT

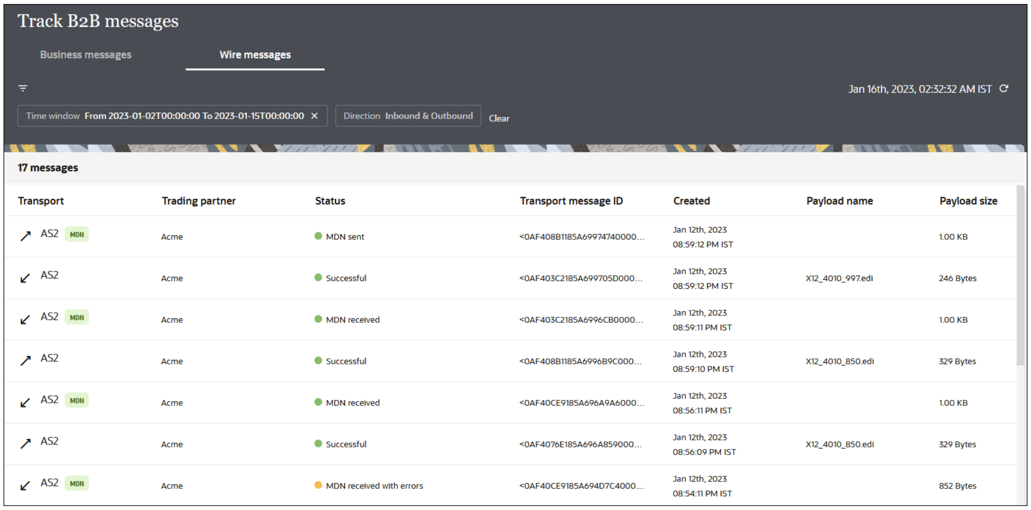
Business Messages

In this view, you see B2B messages as individual business transactions. Rows marked FA denote functional acknowledgments. Assume you receive an inbound message containing batched transactions. This view shows individual rows for each individual transaction within that batch.

Wire Messages

An alternate, low-level technical view of B2B messages is provided with the wire messages. In this view, you see messages in the form in which they are transmitted to or received from a trading partner. This view is useful for troubleshooting, in case the message failed to be delivered to a trading partner or a received message failed signature verification. Rows marked with MDN denote AS2 MDNs (electronic return receipts).

In the case where you receive an inbound message containing a batched transaction, you only see one row corresponding to the actual batch message that was received.



Track B2B messages

Jan 16th, 2023, 02:32:32 AM IST

Time window From 2023-01-02T00:00:00 To 2023-01-15T00:00:00 ✕ | Direction Inbound & Outbound | Clear

17 messages

Transport	Trading partner	Status	Transport message ID	Created	Payload name	Payload size
↗ AS2 MDN	Acme	● MDN sent	<0AF408B1185A69974740000...	Jan 12th, 2023 08:59:12 PM IST		1.00 KB
↙ AS2	Acme	● Successful	<0AF403C2185A699705D000...	Jan 12th, 2023 08:59:12 PM IST	X12_4010_997.edi	246 Bytes
↙ AS2 MDN	Acme	● MDN received	<0AF403C2185A6996CB0000...	Jan 12th, 2023 08:59:11 PM IST		1.00 KB
↗ AS2	Acme	● Successful	<0AF408B1185A6996B9C000...	Jan 12th, 2023 08:59:10 PM IST	X12_4010_850.edi	329 Bytes
↙ AS2 MDN	Acme	● MDN received	<0AF40CE9185A696A9A6000...	Jan 12th, 2023 08:56:11 PM IST		1.00 KB
↗ AS2	Acme	● Successful	<0AF4076E185A696A859000...	Jan 12th, 2023 08:56:09 PM IST	X12_4010_850.edi	329 Bytes
↙ AS2 MDN	Acme	● MDN received with errors	<0AF40CE9185A694D7C4000...	Jan 12th, 2023 08:54:11 PM IST		852 Bytes

More B2B tracking details are provided. See B2B Tracking.

Manage B2B Trading Partners

This section describes how to configure trading partners in B2B for Oracle Integration.

- Define the Host Profile
- Create Trading Partners
- Define Transports
- Create Agreements
- Export and Import a Trading Partner

Trading partner concepts are provided. See Trading Partners.

Define the Host Profile

This section describes how to define the host profile (host company) in B2B for Oracle Integration.

Host trading partner concepts are provided. See Host Company.

- Define Identifiers in the Host Profile
- Create the Host Profile

Define Identifiers in the Host Profile

You specify the host identifier and value when configuring the host profile. Understand the following details about how the identifier is used.

- Where the identifiers are used:
  The following table lists where each type of identifier is used and its purpose.

Identifier Type	Where Used	Purpose	Value Restrictions
Application Partner ID	Not used.	This identifier from the host profile is not used.	Not used.
AS2 Identifier	Under **Trading Partners**, then **Transports & agreements**, then **AS2**, then **AS2 identifiers**, and then **Host identifier**.	Mandatory only when the AS2 transport protocol is used. This identifier type is not used for the FTP transport protocol. For an outbound message, this value is inserted as the `AS2-From` HTTP header of the AS2 message. For an inbound message, this value is used for validation against the `AS2-To` header of the incoming message.	Up to 128 printable ASCII characters except double quotes or back slashes. This value can be any value agreed upon with the trading partner as a value with which they can identify your company. The value is case sensitive.
EDI Interchange ID	Under **Trading Partner**, then **Transports & agreements**, then **Outbound agreement**, and then **Select host identifiers**.	Mandatory for all EDI data formats. This identifier is used as the Interchange Sender ID field of the interchange envelope (ISA segment for X12 and UNB segment for EDIFACT). For an outbound message, this value is inserted as the Interchange Sender ID. For an inbound message, this identifier (from the host profile) is not used.	Up to 15 characters for X12. Up to 35 characters for EDIFACT. The value is case sensitive and must not contain any delimiter characters.

Identifier Type	Where Used	Purpose	Value Restrictions
EDI Interchange ID Qualifier	Same as above.	Mandatory for X12 and optional for EDIFACT. This identifier is used as the Interchange Sender ID Qualifier field of the interchange envelope of the EDI payload. It's a code to indicate the category of the value specified in the EDI Interchange ID (for example, DUNS number, IATA number, and so on). For an outbound message, this value is inserted as the Interchange Sender ID Qualifier. For an inbound message, the value (from the host profile) is not used.	Must be exactly two characters for X12. The value must be from an X12 code list. See EDI Standards Reference. A generic sample value for X12 is `ZZ`. Up to four characters for EDIFACT. The value must be from an EDIFACT code list (https://www.gefeg.com/jswg/cl/v3/11b/cl3.htm). A generic sample value for EDIFACT is `ZZZ`. The value is case sensitive and must not contain any delimiter characters.
EDI Interchange Internal ID	Same as above.	Only used for EDIFACT. In that, it is also optional. EDIFACT defines this field as an identification (for example, a division, branch, or computer system/process) specified by the sender of the interchange to be included if agreed to by the recipient in response interchanges to facilitate internal routing.	Up to 35 characters for EDIFACT. The value is case sensitive and must not contain any delimiter characters.
EDI Interchange Internal Sub ID	Same as above.	Only used for EDIFACT syntax version 4. In that, it is also optional. EDIFACT defines this field as the sublevel of sender internal identification when further sublevel identification is required.	Up to 35 characters for EDIFACT. The value is case sensitive and must not contain any of the delimiter characters.

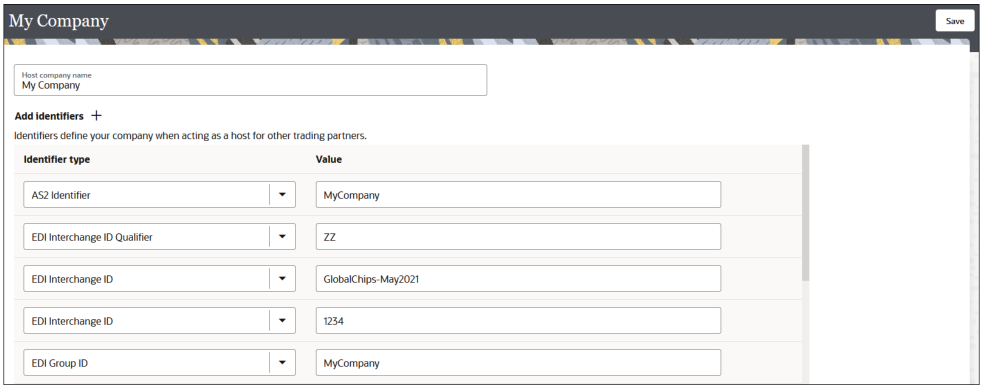
Identifier Type	Where Used	Purpose	Value Restrictions
EDI Group ID	Same as above.	Mandatory for X12 and optional for EDIFACT. EDIFACT defines this field as an identification of the sender's division, department, and so on from which a group of messages is sent. X12 has a similar definition. For an outbound X12 message, this value is inserted in the GS segment as the Application Sender's Code. For an outbound EDIFACT message, this value is inserted in the UNG segment as the sender identification (a UNG is only generated when this identifier is selected in an outbound agreement; otherwise not). For an inbound message, the value from the host profile is not used.	Minimum of two characters and up to 15 characters for X12. Up to 35 characters for EDIFACT. The value is case sensitive and must not contain any delimiter characters.
EDI Group ID Qualifier	Same as above.	Only used for EDIFACT. In that, it is also optional. It's a code to indicate the category of the value specified in the EDI Group ID (for example, DUNS number, IATA number, and so on).	Up to four characters for EDIFACT. The value must be from an EDIFACT code list (https://www.gefeg.com/jswg/cl/v3/11b/cl3.htm). A generic sample value is `ZZZ`. The value is case sensitive and must not contain any delimiter characters.
DUNS	Same as above.	Only used for RosettaNet.	A nine digit, numerical value.
LocationID	Same as above.	This is a free text field and used to populate the Service Header if it is provided.	Free text field.

- Host identifiers in use are protected:
  You cannot delete host identifiers referenced in transports or agreements. The associations must be removed from the transports or agreements *before* you can delete the host identifier from the host profile page.

- Multiple identifiers of the same type:
  You may add multiple identifiers of the same identifier type. For example, you may want to add identifiers based on X12, X12 HIPAA, EDIFACT, or RosettaNet usage, based on business units within your company, or based on an environment such as development, test, production, and so on. The combination of identifier type and value must be unique, and this validation is enforced in the user interface.

  Even though you may define multiple identifiers of the same type, in outbound agreements, you must choose specific unique types so that B2B for Oracle Integration knows exactly which identifiers to insert in an outbound message without ambiguity.

  In the following example, two different rows, both with EDI Interchange ID, are defined.



My Company

Host company name: My Company

Identifiers define your company when acting as a host for other trading partners.

Identifier type	Value
AS2 Identifier	MyCompany
EDI Interchange ID Qualifier	ZZ
EDI Interchange ID	GlobalChips-May2021
EDI Interchange ID	1234
EDI Group ID	MyCompany

- Updating values and applying changes to runtime:
  You can update existing identifier values any time. However, they are not used in runtime B2B processing until:
  - Each transport where an identifier value was referenced is redeployed.
  - Each outbound agreement where an identifier value was referenced is redeployed.

  Redeployment is a life cycle action available for transports and agreements. It applies changes to the runtime on-the-fly, without disruption of message processing.

Create the Host Profile

1. In the navigation pane, click **B2B**, then **Host profile**.
2. Click **Create**.
3. In the **Host company name** field, enter your company name. The name is currently only for reference and not used elsewhere.
4. Select a host identifier or, if none are defined, click **Add** +. You add identifiers to the host profile on behalf of your company. This is typically a one-time activity that you perform before adding your first trading partner.
   Host identifiers define your company when acting as the host interacting with other trading partners. They identify and validate the source of the document when sent by the host. Identifiers defined here are used in two places:
   - Transports

- Outbound agreements

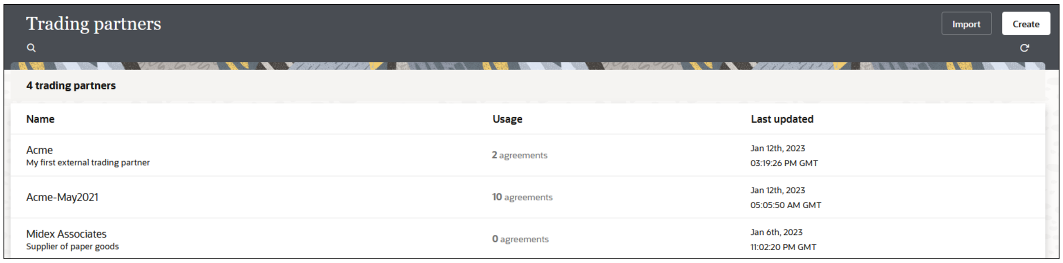
5. Select a host identifier name and specify a value. You can specify multiple name and value pairs for the host.
6. Click **Save**.
   If you change the host identifier value used in a deployed agreement, the changes only take effect after you explicitly redeploy the agreement between the host and the trading partner.
7. If you want to delete any unneeded host identifiers, click **Delete** 🗑. You cannot delete host identifiers referenced by active agreements. The associations must be removed from the agreements before you can delete the host identifier from the host profile page.

Create Trading Partners

You can create and manage trading partners. A trading partner is the external business entity with which your company interacts to send or receive business documents, such as orders and invoices, in electronic form.

Trading partner concepts are provided. See Trading Partners.

1. In the navigation pane, click **B2B**, then **Trading partners**.
   The Trading partners page is displayed. The **Usage** column shows the number of agreements associated with the trading partner. Click the entry to show the number of inbound and outbound agreements for that trading partner.



4 trading partners

Name	Usage	Last updated
Acme My first external trading partner	2 agreements	Jan 12th, 2023 03:19:26 PM GMT
Acme-May2021	10 agreements	Jan 12th, 2023 05:05:50 AM GMT
Midex Associates Supplier of paper goods	0 agreements	Jan 6th, 2023 11:02:20 PM GMT

2. Click **Create**.
3. Enter the trading partner name and an optional description. The **Identifier** field is automatically populated with the name you enter. The values for both must be unique.

> **Note:**
>
> The identifier cannot be changed after creation.

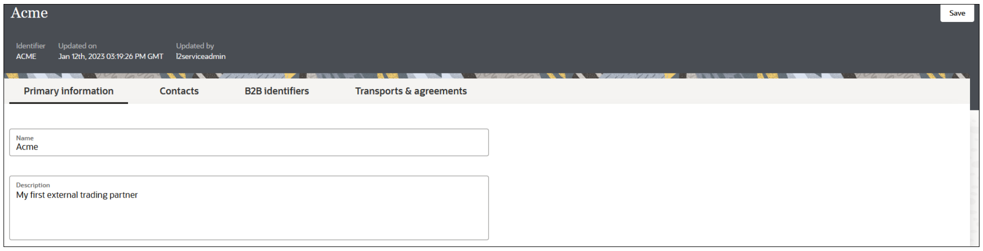
4. **Create**.
   A details panel is displayed that enables you to configure additional information about the newly created trading partner.

See the following sections:

- View Primary Information
- Select Contacts
- Define B2B Identifiers

View Primary Information

1. Click **Primary information** to view the same information you entered when you created the new trading partner. Both the name and identifier of a trading partner must be unique across all trading partners.



Acme

Identifier: ACME
Updated on: Jan 12th, 2023 03:19:26 PM GMT
Updated by: l2serviceadmin

Name: Acme

Description: My first external trading partner

2. If needed, change the name of the trading partner.
   The trading partner name can be changed at any time. There are valid scenarios for renaming an actively-used trading partner. In the real world, companies change their names, have mergers and acquisitions, sell business units, and so on. You need the ability to reflect the new name here.

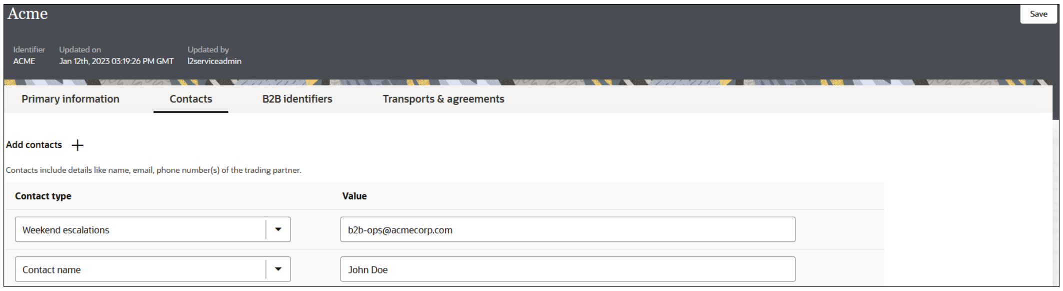
   Implications of renaming an active trading partner:

   The trading partner's name is displayed in a column of the Track B2B messages page for messages received from or sent to this trading partner. If you change the name of an actively-used trading partner with one or more runtime messages processed, then any existing messages on the Track B2B messages page still show the old name of the trading partner prior to the renaming. Any new runtime messages record the new name of the trading partner. While this is intentionally done for auditing purposes, when you filter messages by a trading partner, it shows all messages (both old and new) for that trading partner correctly. Old messages show the old name and new messages show the new name.

Select Contacts

You can add ways to contact the trading partner, such as their name, email, phone number, or short message service (SMS) number. The **Contact type** and **Value** fields are both free text fields. This enables you to enter custom text. Use this information to contact individuals offline, as needed. The **Contacts** field is currently provided only for reference and is not used in B2B for Oracle Integration.

1. Click **Contacts**.
2. From the **Contact type** list, select a method for contacting the trading partner, such as their name, email, phone number, or SMS number. Use this information to contact individuals offline as needed, and not from within B2B for Oracle Integration.
3. Add a corresponding value, then click **Save**.



Acme

Identifier ACME | Updated on Jan 12th, 2023 03:19:26 PM GMT | Updated by l2serviceadmin

Contacts include details like name, email, phone number(s) of the trading partner.

Contact type	Value
Weekend escalations	b2b-ops@acmecorp.com
Contact name	John Doe

Define B2B Identifiers

You collect identifying information from your external trading partner and enter it on their behalf in the **B2B identifiers** section. This is very similar, in concept, to the identifiers specified under **B2B**, then **Host profile**, and then **Identifiers**. In a message exchange, the host's identifiers and trading partner's identifiers are used in the role of a sender or receiver, depending on the message direction.

Direction	Sender	Receiver
Inbound message	Trading Partner (The trading partner's identifiers are used as **Sender ID** or **From Party**.)	Your company (that is, the host) (Host identifiers are used as **Receiver ID** or **To Party**.)
Outbound message	Your Company (that is, the host) (Host identifiers are used as **Sender ID** or **From Party**.)	Trading Partner (The trading partner's identifiers are used as **Receiver ID** or **To Party**.)

B2B identifiers defined here are used in two places:

- Transports
- Outbound agreements

Understand the following details about how the identifier is used.

- The following table lists where each type of identifier is used and its purpose.

Identifier Type	Where Used	Purpose	Value Restrictions
Application Partner ID	Implicitly used by all outbound agreements (** See the note at the bottom of the table.)	Optionally used as an alternate way to specify which trading partner to which to route an outbound message. For an outbound message, you can specify either a Trading Partner ID or an Application Partner ID for trading partner identification. See Create Backend Integrations for more details on this usage.	N/A

Identifier Type	Where Used	Purpose	Value Restrictions
AS2 Identifier	**Trading partners**, then **Transports & agreements**, then **AS2**, then **AS2 identifiers**, and then **Partner's identifier**	Mandatory only when the AS2 transport protocol is used. This identifier type is not used for the FTP transport protocol. For an outbound message, this value is inserted as the `AS2-To` HTTP header of the AS2 message. For an inbound message, this value is used for validation against the `AS2-From` header of the incoming message.	Up to 128 printable ASCII characters except double quotes or backslashes. It can be any value agreed upon with the trading partner as a value with which you can identify the trading partner. The value is case sensitive.
EDI Interchange ID	**Trading Partner**, then **Outbound agreements**, then **Select trading partner identifiers** Also used implicitly for all inbound agreements (** see the note)	Mandatory for all EDI data formats. This identifier is used as either the Interchange Sender or Receiver ID field of the interchange envelope (ISA segment for X12, and UNB segment for EDIFACT). For an outbound message, this value is inserted as the Interchange Receiver ID. For an inbound message, this identifier is used as the Interchange Sender ID to identity a trading partner as a sender of a message. If the EDI Interchange ID on its own does not uniquely identify a trading partner, it is used in combination with the EDI Group ID to locate a unique trading partner.	Up to 15 characters for X12. Up to 35 characters for EDIFACT. The value is case-sensitive and must not contain any of the delimiter characters.

Identifier Type	Where Used	Purpose	Value Restrictions
EDI Interchange ID Qualifier	Same as above.	Mandatory for X12 and optional for EDIFACT. This identifier is used as the Interchange Sender or Receiver ID Qualifier field of the interchange envelope of the EDI payload. It is a code to indicate the category of the value specified in the EDI Interchange ID (for example, DUNS number, IATA number, and so on). For an outbound message, this value is inserted as the Interchange Receiver ID Qualifier. For an inbound message, the value is not currently used (if it were used, it would be treated as the Interchange Sender ID Qualifier).	Must be exactly two characters for X12. The value must be from an X12 code list. See EDI Standards Reference. A generic sample value for X12 is `ZZ`. Up to four characters for EDIFACT. The value must be from an EDIFACT code list (https://www.gefeg.com/jswg/cl/v3/11b/cl3.htm). A generic sample value for EDIFACT is `ZZZ`. The value is case sensitive and must not contain any delimiter characters.
EDI Interchange Internal ID	Same as above.	Only used for EDIFACT, and in that too, it is optional. EDIFACT defines this field as an identification (for example, a division, branch or computer system/process) specified by the sender of the interchange to be included if agreed by the recipient in response interchanges to facilitate internal routing. For an outbound message, this value is inserted as the EDI Interchange Internal ID field in the EDIFACT UNB envelope. For an inbound message, this value is not used.	Up to 35 characters for EDIFACT. The value is case sensitive and must not contain any delimiter characters.

Identifier Type	Where Used	Purpose	Value Restrictions
EDI Interchange Internal Sub ID	Same as above.	Only used for EDIFACT syntax version 4. In that, it is also optional. EDIFACT defines this field as the sublevel of sender internal identification when further sublevel identification is required. For an outbound message, this value is inserted as the EDI Interchange Internal Sub ID field in the EDIFACT UNB envelope, only if the message follows the Syntax Version 4 (which is the default). For an inbound message, this value is not used.	Up to 35 characters for EDIFACT. The value is case sensitive and must not contain any of the delimiter characters.

Identifier Type	Where Used	Purpose	Value Restrictions
EDI Group ID	Same as above.	This is mandatory for X12 and optional for EDIFACT. EDIFACT defines this field as an identification of the sender's division, department, and so on from which a group of messages is sent. X12 has a similar definition. For an outbound X12 message, this value is inserted in the GS segment as the Application Receiver's Code. For an outbound EDIFACT message, this value is inserted in the UNG segment as the Receiver's Identification (a UNG is only generated when this identifier is selected in an outbound agreement; otherwise, not). For an inbound message, this value is used only in case the EDI Interchange ID, on its own, is not enough to uniquely identify a trading partner. In that case, a combination of the EDI Interchange ID and the EDI Group ID is used to locate a unique trading partner.	Minimum of two characters and up to 15 characters for X12. Up to 35 characters for EDIFACT. The value is case sensitive and must not contain any delimiter characters.
EDI Group ID Qualifier	Same as above.	Only used for EDIFACT. In that, it is optional. It is a code to indicate the category of the value specified in the EDI Group ID (for example, DUNS number, IATA number, and so on). Only used for an outbound message, to insert as EDI Group ID Qualifier, if specified.	Up to four characters for EDIFACT. The value must be from an EDIFACT code list (https://www.gefeg.com/jswg/cl/v3/11b/cl3.htm). A generic sample value is `ZZZ`. The value is case sensitive and must not contain any delimiter characters.
DUNS	Same as above.	Only used for RosettaNet.	A nine digit, numerical value.

Identifier Type	Where Used	Purpose	Value Restrictions
LocationID	Same as above.	This is a free text field and used to populate the Service Header if it is provided.	Free text field.

> **Note:**
>
> ** An implicit usage means you don't explicitly select the identifier in an inbound or outbound agreement. Instead, when you deploy an agreement, the identifier is automatically used in the runtime processing.

- Used B2B identifiers are protected:
  You cannot delete trading partner's B2B identifiers that are explicitly referenced in transports or agreements. The associations must be removed from the transports or agreements *before* you can delete an identifier from the trading partner's **B2B identifiers** section.
- Multiple identifiers of the same type:
  The concept is similar to defining multiple identifiers in the host profile.

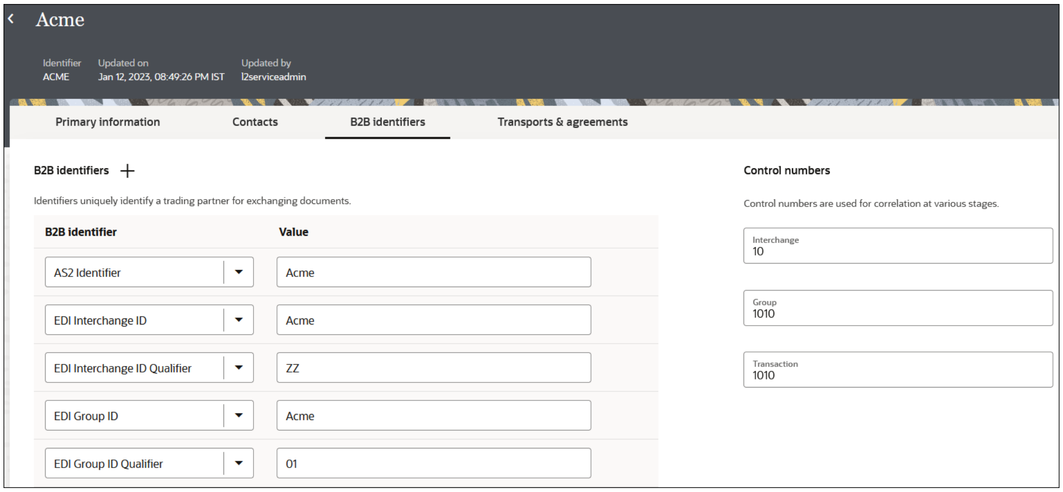
  You may add multiple B2B identifiers of the same identifier type. For example, you may want to add identifiers based on X12, X12 HIPAA, EDIFACT, or RosettaNet usage, based on business units within the trading partner, or based on an environment such as development, test, production, and so on. The combination of identifier type and value must be unique. This validation is enforced in the user interface.

  Even though you may define multiple identifiers of the same type, in outbound agreements, you must select specific unique types so that B2B knows exactly which identifiers to insert in an outbound message without ambiguity.
- Updating values and applying changes to runtime:
  You can update existing trading partner's B2B identifier values at any time. However, they are not used in runtime B2B processing, until:
  - Each transport where an identifier value was referenced is redeployed.
  - Each outbound agreement where an identifier value was referenced is redeployed.
  - For B2B identifiers that are implicitly used, any of the agreements is redeployed (it can be any one of the inbound or outbound agreements).

  Redeployment is a life cycle action available for transports and agreements. It applies changes to the runtime, on-the-fly, without disruption of message processing.

1. Click **B2B identifiers** to define the identifiers that uniquely identify a trading partner. This information is similar to what you defined for the host. See Define the Host Profile.
2. Select an identifier name and specify a value. You can specify multiple name and value pairs. If no identifiers are defined, click **Add** + to add a new identifier, selecting an identifier type and entering a value.
3. In the **Control numbers** section, enter the initial control numbers in the **Interchange**, **Group**, and **Transaction** fields. These numbers are automatically

incremented within the ISA and GS segments for each external trading partner. This feature is only applicable for outbound X12, X12 HIPAA, RosettaNet, and EDIFACT documents.
For example, assume the last control number used for trading partner A in a development environment was 100004. When migrating to a test environment, they want to start incrementing the initial control number for trading partner A with 200001. This eliminates the possibility of duplicate control numbers and enables them to identify the environment from which EDI messages are being sent.



Identifier: ACME — Updated on: Jan 12, 2023, 08:49:26 PM IST — Updated by: l2serviceadmin

B2B identifiers +

Identifiers uniquely identify a trading partner for exchanging documents.

B2B identifier	Value
AS2 Identifier	Acme
EDI Interchange ID	Acme
EDI Interchange ID Qualifier	ZZ
EDI Group ID	Acme
EDI Group ID Qualifier	01

Control numbers

Control numbers are used for correlation at various stages.

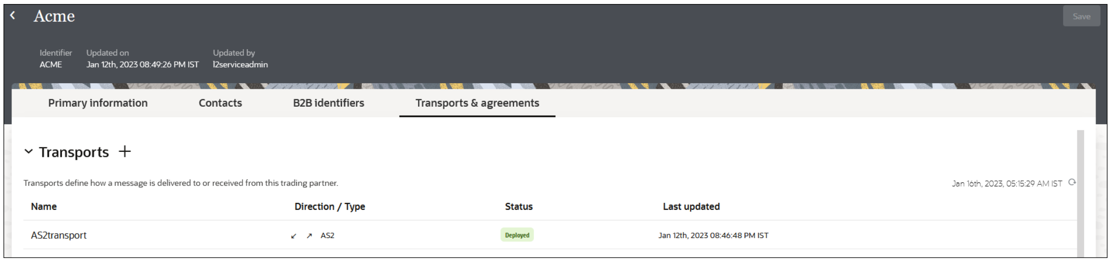
Interchange: 10
Group: 1010
Transaction: 1010

4. Click **Save**.

Define Transports

A transport maps to a technical communication protocol. In most cases, you add one transport per trading partner to receive or send messages from the partner. The AS2, RosettaNet, FTP, and REST transport protocols are currently supported.

The Transports & agreements page shows a list of transports for a trading partner.



Identifier: ACME — Updated on: Jan 12th, 2023 08:49:26 PM IST — Updated by: l2serviceadmin

Transports +

Transports define how a message is delivered to or received from this trading partner.

Jan 16th, 2023, 05:15:29 AM IST

Name	Direction / Type	Status	Last updated
AS2transport	↙ ↗ AS2	Deployed	Jan 12th, 2023 08:46:48 PM IST

Each transport is listed with its name, direction and type, status, and last updated time. The direction is an indicator of whether it is configured to receive (down arrow), send (up arrow), or both. Status can be:

- Not deployed
- Deploying
- Deployed

- Failed

Hover the cursor over a row to see additional icon buttons to view configuration, edit configuration, and open an action menu. When you define the transport protocol (AS2, FTP, RosettaNet, or REST) and transport parameters to use, you automatically create two integrations (to receive and send messages from or to the trading partner).

A transport includes the following configuration settings:

- A mandatory connection that you must separately create and select in the transport
- Protocol-specific settings (for example, AS2 settings, FTP settings)

Transport concepts are provided. See Transports for Communication.

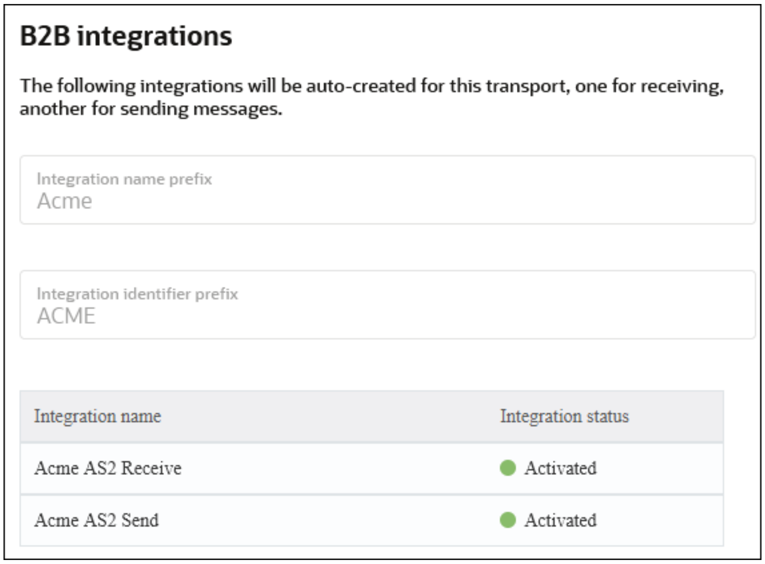
B2B Integrations

Two integrations are created automatically under-the-covers when a transport is created. These integrations are the heart of the transport for its runtime functioning. The two integrations actually process the runtime messages that pass through the transport.

- B2B integration for receiving messages
- B2B integration for sending messages

See Integrations Used for B2B Message Processing.

The **B2B integrations** section of the Edit transport panel shows the status of the two integrations.



B2B integrations

The following integrations will be auto-created for this transport, one for receiving, another for sending messages.

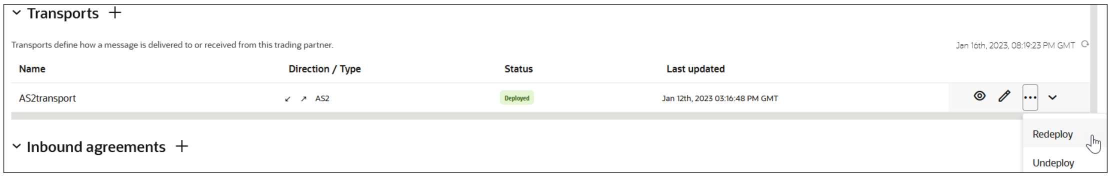
Integration name prefix
Acme

Integration identifier prefix
ACME

Integration name	Integration status
Acme AS2 Receive	● Activated
Acme AS2 Send	● Activated

Life Cycle Actions for Transports

Click the action menu on a row to view available actions.



Transports +

Transports define how a message is delivered to or received from this trading partner.

Jan 16th, 2023, 08:19:23 PM GMT

Name	Direction / Type	Status	Last updated
AS2transport	↙ ↗ AS2	Deployed	Jan 12th, 2023 03:16:48 PM GMT

Life cycle actions for transports are:

- Create a transport: Adds a definition for the transport. The pair of B2B integrations are automatically created and permanently linked to this transport.
- Receive schedule: Only available for the FTP transport. This action navigates you to the schedule configuration page where you can define the polling schedule for the B2B integration for receiving messages. You should add a schedule prior to deploying the transport. This enables the deploy action to also automatically start your schedule.
- Deploy a transport: Makes the transport visible for runtime processing and also activates the B2B integrations. This transport can now receive and send messages.
- Export a transport: Exports a transport for import into a different transport.
- Redeploy a transport: Applies configuration changes to the runtime on-the-fly without disrupting message processing (some types of configuration changes require undeployment and redeployment). See Apply Configuration Changes To Runtime.
- Undeploy a transport: Hides the transport from runtime processing and also deactivates the B2B integrations. This transport can no longer receive and send messages.
- Delete a transport: Removes the definition and also deletes the B2B integrations.

After a transport is deployed, the status column displayed in the user interface is an overall view of the status of the two integrations.

A failed status for a transport usually means one of these integrations failed to activate or was (unexpectedly) manually deactivated.

Apply Configuration Changes To Runtime

Configuration changes to a transport are not applied to runtime processing unless you execute an action. The following table lists the type of change and the action needed.

Type of Configuration Change	Actions to Take to Apply the Changes at Runtime
Common configuration settings in a transport	Redeploy the transport.
Protocol-specific configuration changes in a transport (that is, AS2, RosettaNet, REST, or FTP settings), including changes to the values of identifiers	Redeploy the transport.
Changes to the connection properties (other than user name/password)	Undeploy and then deploy the transport.
Changes to the user name/password credentials in the connection	No actions are needed. The transport automatically uses the updated credentials if authentication failure occurs at runtime (older, cached credentials are discarded). If you want to force the change, undeploy and then deploy the transport.

Type of Configuration Change	Actions to Take to Apply the Changes at Runtime
Changes to certificates in the Certificate management page	Undeploy and then deploy the transport.

The following sections describe transport configuration details:

- Collect AS2 Transport Details
- Define an AS2 Transport
- Collect RosettaNet Transport Details
- Define a RosettaNet Transport
- Collect REST Transport Details
- Define a REST Transport
- Collect FTP Transport Details
- Define an FTP Transport
- FTP Receive Integration Postprocessing Behavior

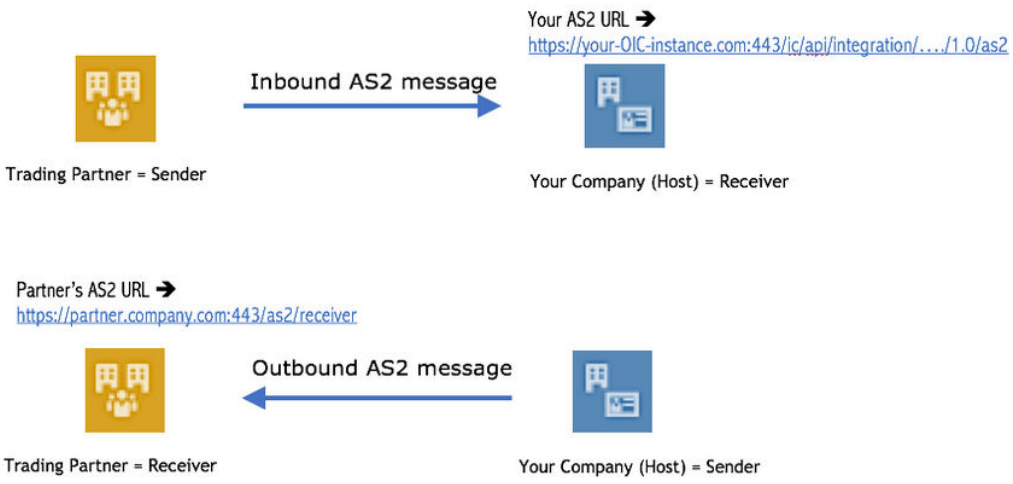
Collect AS2 Transport Details

You must collect AS2 transport details before you can define the AS2 transport in Oracle Integration.

- Collect AS2 Transport Details
- Credentials
- Certificates
- AS2 URL for Receiving

Collect AS2 Transport Details

AS2 is an HTTP-based, point-to-point protocol typically used for real-time transactions. A bidirectional AS2 message exchange involves two AS2 endpoints, as shown below:


Your AS2 URL ➔
https://your-OIC-instance.com:443/ic/api/integration/..../1.0/as2
Inbound AS2 message
Trading Partner = Sender
Your Company (Host) = Receiver
Partner's AS2 URL ➔
https://partner.company.com:443/as2/receiver
Outbound AS2 message
Trading Partner = Receiver
Your Company (Host) = Sender

While creating this transport, you must collect information from your trading partner about their AS2 endpoint. You also may need to provide information about your AS2 endpoint to your trading partner. The following table describes the information you need to collect:

For...	What You Need From Your Trading Partner	What You Need to Provide to Your Trading Partner
Basic connectivity	- The partner's AS2 URL. - An SSL certificate with a public key, if a self-signed certificate was used. - Username/password credentials for HTTP basic authentication, if enabled.	- Your AS2 URL (see AS2 URL for Receiving) - Username/password credentials (see Credentials)
Two-way SSL for outbound connections (an optional feature)	If you select the **Invoke** or **Trigger and invoke** role, you can create a certificate alias to use for establishing client identity during two-way SSL communication.	See Prerequisites for Creating a Connection in *Using the AS2 Adapter with Oracle Integration 3* .
Signed or encrypted AS2 messages (an optional feature)	- The partner's public certificate for signing and encryption. (Typically the same certificate is used for both signing and encryption, but if the partner prefers to use different ones, you should get two separate public certificates from them.)	- Your public certificate for signing and encryption (see Certificates)

Signing and encryption are optional features in AS2. You can start with only the basic connectivity first and add signing/encryption later. Signing/encryption provide nonrepudiation, message integrity, and security features and are recommended for production environments. However, there is a bit more complexity in setting those up.

The following table shows which PKI key is used in each scenario:

Message Configuration	Inbound Message	Outbound Message
Signed AS2 message	Partner's *public key for signing* is used to verify a signed message.	Your company's *private key for signing* is used to digitally sign the message.
Encrypted AS2 message	Your company's *private key for encryption* is used to decrypt the message.	Partner's *public key for encryption* is used to encrypt the message.

If you already have the AS2 endpoint information from your trading partner, follow these steps:

Step	Description
1	Upload each of the partner's certificates. Upload SSL certificates as X.509 Trust, whereas upload signing and encryption as X.509 Identity. For identity certificates, you decide and enter a unique alias. Note the aliases. In the navigation pane, click **Settings**, then **Certificates**. See Certificates.

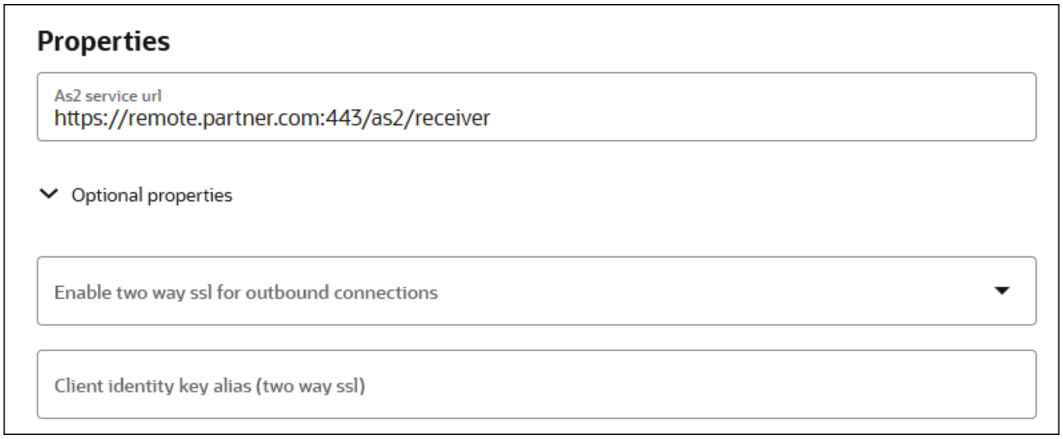
Step	Description
2	If signing/encryption is a requirement, acquire or generate a key-pair for signing and encryption (or two separate key-pairs, if you want to use separate keys for signing and encryption). Upload the private key as X.509 Identity and note the alias and password you enter. Share the public key with your trading partner. However, never share the private key. In the navigation pane, click **Settings**, then **Certificates**. See Certificates.
3	Create an AS2 connection with the **Trigger and invoke** role. In the Connections page, enter: • The partner's AS2 URL in the **AS2 service URL** field • The username/password in the corresponding fields If signing/encryption are a requirement, configure the AS2 connection further. If both your partner and your company use one certificate for signing and encryption, select **AS2 Basic Policy**. If either of you use different certificates, select **AS2 Advanced Policy**. • For AS2 Basic Policy, enter the partner's certificate alias, corresponding to the identity certificate from step 1. • For AS2 Basic Policy, enter the private key alias and key password, corresponding to the identity certificate from step 2. • For AS2 Advanced Policy, enter each of the certificate aliases into the fields. See Configure Connection Security in *Using the AS2 Adapter with Oracle Integration 3* .
4	Test the AS2 Adapter connection, to make sure it succeeds. If it fails, review the errors, verify that the AS2 URL entered is correct, and verify that the certificate aliases are correct. Save the AS2 Adapter connection.
5	Create an AS2 transport, selecting the AS2 Connection created in step 3. Complete the configuration. See Define an AS2 Transport.
6	Deploy the AS2 transport. After the state changes to deployed, the transport is ready for use.

If you do not yet have the AS2 endpoint information from your trading partner, but want to get your side ready for receiving AS2 messages, follow these steps:

Step	Description
1	Same as Step 1 in the previous table. Skip this step for now, but you can perform it when the information becomes available from the trading partner
2	Same as Step 2 in the previous table.
3	Same as Step 3 in the previous table, but given that the partner's AS2 URL is not yet available, enter a temporary placeholder URL in the **AS2 service URL** field. This can be the URL of your Oracle Integration instance, copy and pasted from the browser URL address or any other valid URL. This placeholder is only needed to pass the connection test (which fails if the URL is invalid). Outbound AS2 messages do not work with this placeholder, but inbound messages can be received (since the AS2 service URL is not used when receiving inbound messages).
4	Same as Step 4 in the previous table.
5	Same as Step 5 in the previous table.
6	Same as Step 6 in the previous table.

An example of an AS2 Adapter connection is shown below.

- Connection properties:



Properties

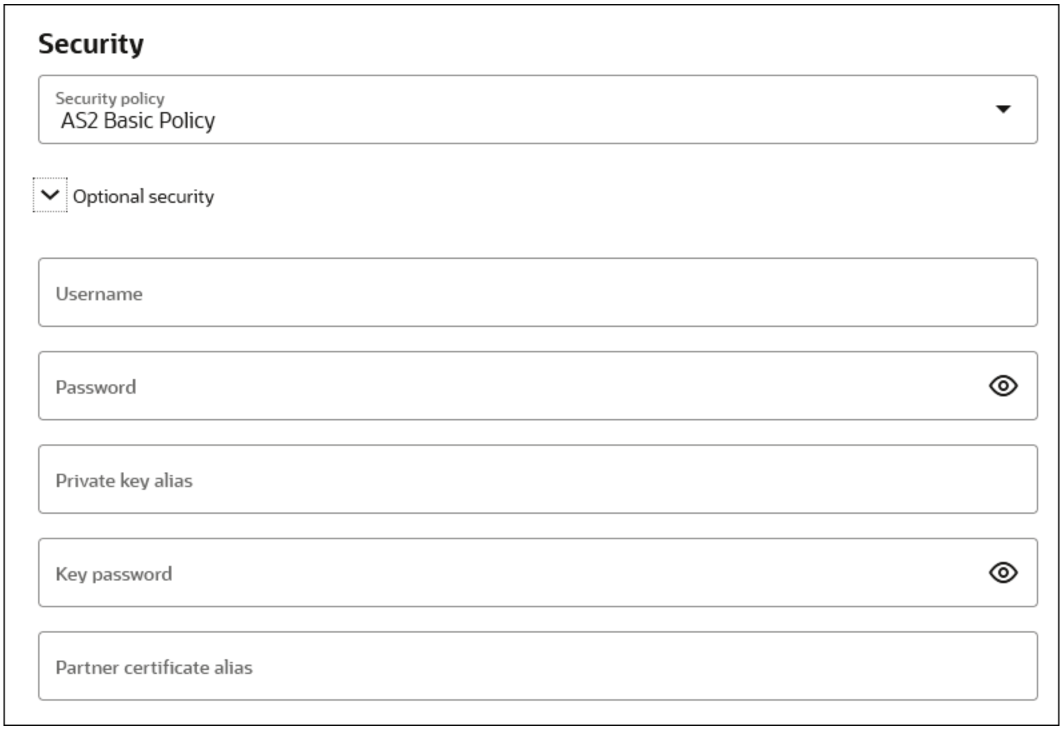
As2 service url
https://remote.partner.com:443/as2/receiver

Optional properties

Enable two way ssl for outbound connections

Client identity key alias (two way ssl)

- Connection security:



Security

Security policy
AS2 Basic Policy

Optional security

Username

Password

Private key alias

Key password

Partner certificate alias

Credentials

To receive messages over AS2 from an external trading partner, HTTP basic authentication is enforced. Your trading partner is required to send the Authorization HTTP header with username/password credentials you provide them in an AS2 message.

For internal testing you may use the same credentials that you use to log in to Oracle Integration to send test AS2 messages. However, it is not safe to share these credentials with an external trading partner because they can also log in to Oracle Integration with these credentials.

Instead, create a new user account in the Oracle Integration Identity Management application. Grant the Service Invoker role to this user account. This account is enough to send messages, but does not grant permissions to access any user interface pages in Oracle Integration. Share the username and password of this new user with the trading partner.

Certificates

If you want to enable encryption or signing for the AS2 communication, you must create a key pair and certificate following your company's process and generate a CA-signed certificate that you use for AS2 decryption and signing.

For testing with a self-signed certificate, here are simple steps to generate a key-pair using the Java `keytool`:

1. Generate public/private key pair using `keytool`.
   a. Specify any alias and a keystore file name, replacing `b2b-private-key-alias` and `b2b.jks` with your values.
   b. Enter a keystore password when prompted and note it.
   c. Enter your organization's information when prompted.

   This generates a key pair (a public key and associated private key) and self-signed digital certificate in a keystore. If the keystore does not exist, it is created.

   ```
   keytool -genkey -keyalg RSA -alias b2b-private-key-alias -validity
   1095 -keystore b2b.jks
   ```

2. Upload the JKS into Oracle Integration as the X.509 type (SSL transport) and Identity category using the same alias and password as entered above (this is part of Step 3 from the table of steps above).
3. Export the public key from this keystore as follows.
   a. Replace `b2b.jks`, `b2b-private-key-alias`, and *`public.cer`* with your keystore file name, alias that was used previously, and a file name to store the public certificate.

      ```
      keytool -export -keystore <b2b.jks> -alias <b2b-private-key-
      alias> -file <public.cer>
      ```

4. Convert it to any other industry-standard format using `keytool` as per your preference, if necessary. Share only the public certificate *`public.cer`* with your trading partner (never share the private key with anyone). Your trading partner uses the public key certificate for signature verification and encryption.

AS2 URL for Receiving

You need the AS2 URL for your AS2 endpoint to share with your trading partner. Once the transport is deployed (indicating it is ready to receive and/or send messages), your AS2 endpoint URL is displayed in the **AS2 endpoint URL for receiving** transport field. Copy this AS2 URL to share with your trading partner. This AS2 URL is not common across all trading partners; it is specific to the current trading partner that you are viewing or editing. Only that specific trading partner may send AS2 messages to this URL.

The AS2 URL is the URL to invoke the AS2 integration for receiving messages for this transport. While you can also get the same from the Integrations page, this provides an easier way to access it.

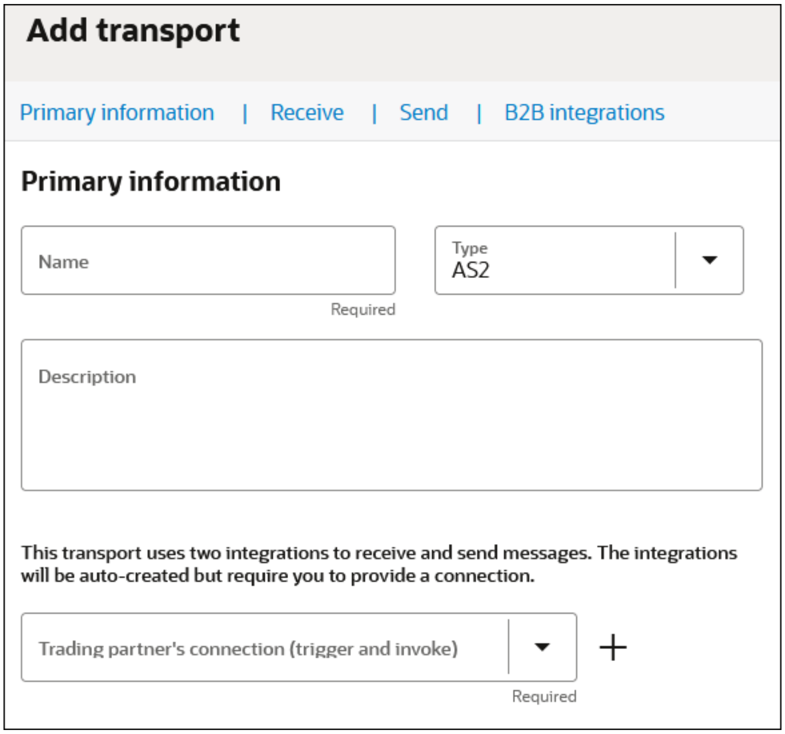
Define an AS2 Transport

Once you have collected AS2 transport details, you can define an AS2 transport in Oracle Integration.

1. In the navigation pane, click **B2B**, then **Trading partners**.
   The Trading partners page is displayed.
2. In the row of the trading partner for which to define transports, click **Edit** ✎.
3. Click **Transports & agreements**.

Define Transports to Create Integrations

1. In the **Transports** section, click **Add** + to define how a message is delivered to or received from this trading partner. The configuration panel is opened:



Add transport

Primary information | Receive | Send | B2B integrations

Primary information

Name — Required

Type: AS2

Description

This transport uses two integrations to receive and send messages. The integrations will be auto-created but require you to provide a connection.

Trading partner's connection (trigger and invoke) — Required

2. Define the following details.

Section	Description
Primary information section • **Name**	Enter a name for the transport. The name is used for display purposes only.
• **Type**	Select **AS2** from the dropdown. This represents the communication protocol you use to exchange messages with your trading partner. Based on selecting AS2, the appropriate configuration settings for the AS2 transport are displayed.

Section	Description
• **Description**	Enter an optional description of the transport. The description is used for display purposes only.
• **Trading partner's connection (Trigger and invoke)**	Select an existing AS2 Adapter connection configured for connectivity to your trading partner or click **Add** + to create a new AS2 Adapter connection on the Connections page. It must be a **Trigger and invoke** connection. Trigger Only and Invoke Only connections cannot be used for transports. See Create a Connection in *Using the AS2 Adapter with Oracle Integration 3* . If you want to select another connection, you can do so when this transport is not deployed. Once you deploy the transport, the connection selection cannot be changed. You can modify the configuration properties inside the connection at any time. However, if you modify the connection settings after this transport is deployed, you must undeploy and then redeploy the transport for the changes to take effect.
• **Partner's identifier**	Used as an `AS2-To` header for outbound messages and as the expected `AS2-From` header for inbound messages. See Define B2B Identifiers.
• **Host identifier**	Used as the `AS2-From` header for outbound messages and as the expected `AS2-To` header for inbound messages. See Host Profile.
• **Character encoding**	Select the character encoding to apply to all payloads processed through this transport. The character encoding is used at the EDI parsing (inbound) or EDI generation (outbound) step.
Receive section • **Allow any AS2 identifier defined for this trading partner**	Enable this checkbox if you want to accept an `AS2-From` header value from within a set of possible values (as opposed to accepting just one value). A typical case for this is when different business units within the trading partner's organization use different AS2 identifiers, and you want to accept messages from all of them. For this to work, you must also add all acceptable `AS2-From` values as B2B identifiers to the trading partner. If this is disabled, only one specific AS2 identifier selected for **Partner's identifier** is accepted at runtime for inbound processing.
• **Do not validate the AS2-to header**	Enable this if you want to allow any value for the `AS2-To` header, effectively disabling the strict validation otherwise done against the host's AS2 identifier.
• **AS2 endpoint URL for receiving**	This is a display-only field. After the transport is deployed, indicating it is ready to receive and send messages, your AS2 endpoint URL is displayed. You may share this URL with your trading partner. This AS2 URL is not common across all trading partners. It is specific to the current trading partner that you are viewing or editing.
Send section • **AS2 subject**	An optional field that is inserted into all outbound messages that use this transport, as the `Subject` HTTP header.
• **Content-type**	The payload's type for outbound messages. Typically for EDI payloads, use **application/EDI-Consent** as a generic way to specify that the payload can be either X12 or EDIFACT.
• **User-defined content-type**	To enter another value not already available in the **Content-type** drop-down list, select **User-defined Type** and enter your value.

Section	Description
• **Signature**	If you want to enable message signing for outbound messages, select the appropriate signing algorithm from the drop-down list. For signing, you must configure the AS2 Adapter connection appropriately with certificates, as described in Step 3 of the above table. This field does not apply for inbound message processing. Signature verification of signed inbound messages is done automatically and there is no configurable option to enable or disable it. You must configure the AS2 connection with the right certificates for that to work also.
• **Encryption**	If you want to enable message encryption for outbound messages, select the appropriate encryption algorithm from the drop-down list. For encryption, you must configure the AS2 Adapter connection appropriately with certificates as described in Step 3 of the above table. This field does not apply for inbound message processing. Decryption of encrypted inbound messages is done automatically and there is no configurable option to enable or disable it. You need to configure the AS2 Adapter connection with the correct certificates for that to work also.
• **Compression**	Optionally select to compress the outbound message. • **None** • **Digitally sign first, then compress**: Sign the outbound message before compressing it. • **Compress first, then digitally sign**: Compress the outbound message before signing it.
• **Request MDN**	Select a value in the drop-down list to request a synchronous or asynchronous MDN (or **None** for no MDN) when sending outbound messages. This field does not apply to inbound message processing. MDN is automatically generated and sent back to the trading partner based on whether your trading partner has requested an MDN as part of an inbound message. There is no configuration required. The AS2 HTTP headers `Disposition-Notification-To`, `Disposition-Notification-Options`, and `Receipt-Delivery-Option` convey whether the partner wants to receive an MDN back, whether it should be synchronous or asynchronous, and whether it needs to be signed. The AS2 transport automatically handles the MDN processing. See the AS2 specification (RFC 4130) to understand more technical details. • **None**: Request that no MDN be sent back. • **Async MDN**: Request that the MDN be sent separately from the outbound message. • **Sync MDN**: Request that the MDN be sent immediately in the response.
• **Request Signed MDN**	If you choose to request an MDN for an outbound message, select the checkbox to ask the trading partner to send signed MDNs. You must configure the AS2 Adapter connection appropriately with certificates as described in Step 2 of the table of steps above, so that the signed MDNs can be validated.
B2B Integrations section • **Integration name prefix**	Enter a short prefix that is used to form the complete integration names for receiving messages and sending messages. For the AS2 transport, it forms the integration names: ***your_prefix* AS2 Receive** and ***your_prefix* AS2 Send**. Details about these integrations are provided. See Create B2B Integrations for Receiving and Sending.

Section	Description
• **Integration identifier prefix**	Enter a short prefix that is used to form complete integration identifiers for receiving messages and sending messages. For the AS2 transport, it forms the integration identifiers: ***your_prefix*****_AS2_RECEIVE** and ***your_prefix*****_AS2_SEND**. The final integration identifier must be unique across all integrations. Therefore, ensure that you enter a prefix that is unique. If the uniqueness check fails, you get the opportunity to try with a different prefix.

3. Click **Add**.
   The new transport is displayed.
4. Select **Actions** ⋯, then select **Deploy**.
5. Select **Deploy** again when prompted.
   If successful, the following message is displayed.

   ```
   Transport transport_name was deployed successfully.
   ```

   The transport status is changed to **Active**.
6. Go to the Integrations page and note that both integrations are created and activated.
7. If you need to undeploy the transport, select **Actions** ⋯, then select **Undeploy**.
   Undeploying the transport also undeploys the integrations.

Collect RosettaNet Transport Details

You must collect RosettaNet transport details before you can define the RosettaNet transport in Oracle Integration.

- Collect RosettaNet Transport Details
- Credentials
- Certificates
- RosettaNet URL for Receiving

Collect RosettaNet Transport Details

RosettaNet is an HTTP-based, point-to-point protocol typically used for real-time transactions. A bidirectional RosettaNet message exchange involves two RosettaNet endpoints, as shown below:

While creating this transport, you must collect information from your trading partner about their RosettaNet endpoint. You also may need to provide information about your RosettaNet endpoint to your trading partner. The following table describes the information you need to collect:

For...	What You Need From Your Trading Partner	What You Need to Provide to Your Trading Partner
Basic connectivity	• The partner's RosettaNet service URL. • An SSL certificate with a public key, if a self-signed certificate was used. • Username/password credentials for HTTP basic authentication, if enabled.	• Your RosettaNet service URL • Username/password credentials
Two-way SSL for outbound connections (an optional feature)	If you select the **Invoke** role, you can create a certificate alias to use for establishing client identity during two-way SSL communication.	See Prerequisites for Creating a Connection in *Using the RosettaNet Adapter with Oracle Integration 3*.
Signed or encrypted RosettaNet messages (an optional feature)	• The partner's public certificate for signing and encryption. (Typically the same certificate is used for both signing and encryption, but if the partner prefers to use different ones, you should get two separate public certificates from them.)	• Your public certificate for signing and encryption (see Certificates)

Signing and encryption are optional features in RosettaNet. You can start with only the basic connectivity first and add signing/encryption later. Signing/encryption provide nonrepudiation, message integrity, and security features and are recommended for production environments. However, there is a bit more complexity in setting those up.

The following table shows which PKI key is used in each scenario:

Message Configuration	Inbound Message	Outbound Message
Signed RosettaNet message	Partner's *public key for signing* is used to verify a signed message.	Your company's *private key for signing* is used to digitally sign the message.
Encrypted RosettaNet message	Your company's *private key for encryption* is used to decrypt the message.	Partner's *public key for encryption* is used to encrypt the message.

If you already have the RosettaNet endpoint information from your trading partner, follow these steps:

Step	Description
1	Upload each of the partner's certificates. Upload SSL certificates as X.509 Trust, whereas upload signing and encryption as X.509 Identity. For identity certificates, you decide and enter a unique alias. Note the aliases. In the navigation pane, click **Settings**, then **Certificates**. See Certificates.

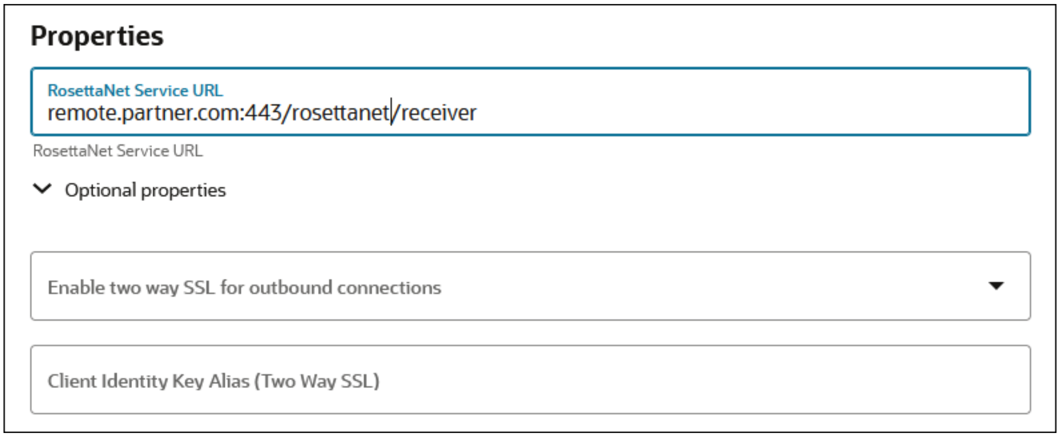
Step	Description
2	If signing/encryption is a requirement, acquire or generate a key-pair for signing and encryption (or two separate key-pairs, if you want to use separate keys for signing and encryption). Upload the private key as X.509 Identity and note the alias and password you enter. Share the public key with your trading partner. However, never share the private key. In the navigation pane, click **Settings**, then **Certificates**. See Certificates.
3	Create a RosettaNet connection with the **Trigger** or **Invoke**role. In the Connections page, enter: • The partner's RosettaNet URL in the **RosettaNet service URL** field • The username/password in the corresponding fields If signing/encryption are a requirement, configure the RosettaNet connection further. If both your partner and your company use one certificate for signing and encryption, select **RosettaNet Basic Policy**. If either of you use different certificates, select **RosettaNet Advanced Policy**. • For RosettaNet Basic Policy, enter the partner's certificate alias, corresponding to the identity certificate from step 1. • For RosettaNet Basic Policy, enter the private key alias and key password, corresponding to the identity certificate from step 2. • For RosettaNet Advanced Policy, enter each of the certificate aliases into the fields. See Configure Connection Security in *Using the RosettaNet Adapter with Oracle Integration 3*.
4	Test the RosettaNet Adapter connection, to make sure it succeeds. If it fails, review the errors, verify that the RosettaNet URL entered is correct, and verify that the certificate aliases are correct. Save the RosettaNet Adapter connection.
5	Create a RosettaNet transport, selecting the RosettaNet connection created in step 3. Complete the configuration. See Define a RosettaNet Transport.
6	Deploy the RosettaNet transport. After the state changes to deployed, the transport is ready for use.

If you do not yet have the RosettaNet endpoint information from your trading partner, but want to get your side ready for receiving RosettaNet messages, follow these steps:

Step	Description
1	Same as Step 1 in the previous table. Skip this step for now, but you can perform it when the information becomes available from the trading partner
2	Same as Step 2 in the previous table.
3	Same as Step 3 in the previous table, but given that the partner's RosettaNet URL is not yet available, enter a temporary placeholder URL in the **RosettaNet service URL** field. This can be the URL of your Oracle Integration instance, copy and pasted from the browser URL address or any other valid URL. This placeholder is only needed to pass the connection test (which fails if the URL is invalid). Outbound RosettaNet messages do not work with this placeholder, but inbound messages can be received (since the RosettaNet service URL is not used when receiving inbound messages).
4	Same as Step 4 in the previous table.
5	Same as Step 5 in the previous table.
6	Same as Step 6 in the previous table.

An example of a RosettaNet Adapter connection is shown below.

- Connection properties:



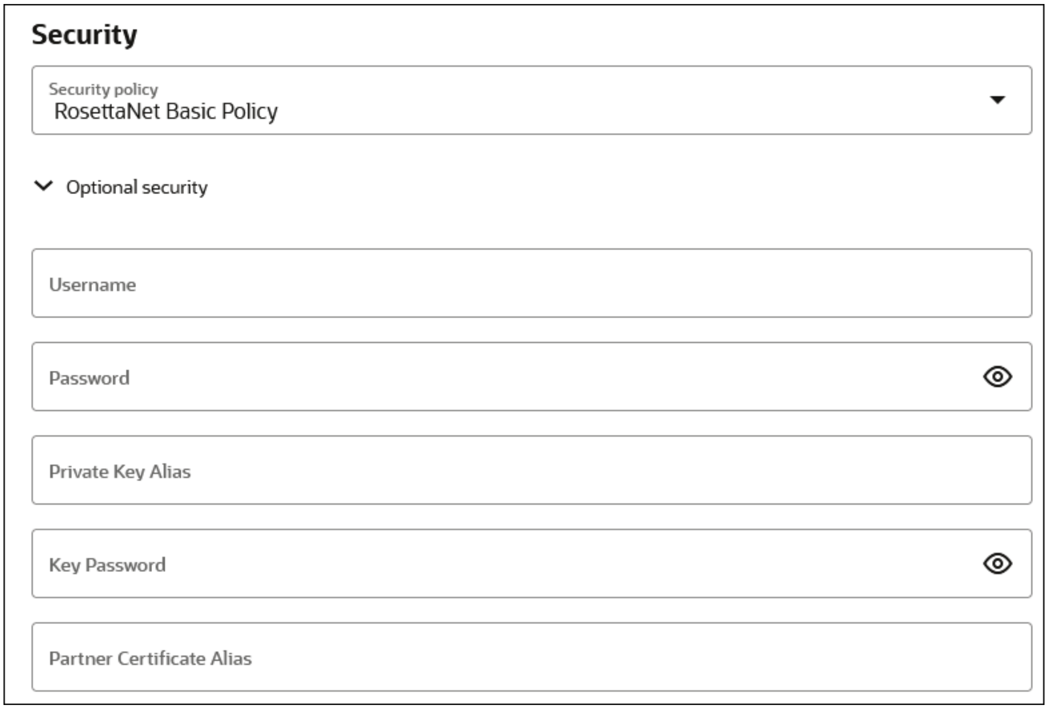
Properties

RosettaNet Service URL: remote.partner.com:443/rosettanet/receiver

RosettaNet Service URL

Optional properties
Enable two way SSL for outbound connections
Client Identity Key Alias (Two Way SSL)

- Connection security:



Security

Security policy: RosettaNet Basic Policy

Credentials

To receive messages over RosettaNet from an external trading partner, HTTP basic authentication is enforced. Your trading partner is required to send the Authorization HTTP header with username/password credentials you provide them in a RosettaNet message.

For internal testing you may use the same credentials that you use to log in to Oracle Integration to send test RosettaNet messages. However, it is not safe to share these credentials with an external trading partner because they can also log in to Oracle Integration with these credentials.

Instead, create a new user account in the Oracle Integration Identity Management application. Grant the Service Invoker role to this user account. This account is enough to send messages, but does not grant permissions to access any user interface pages in Oracle Integration. Share the username and password of this new user with the trading partner.

Certificates

If you want to enable encryption or signing for the RosettaNet communication, you must create a key pair and certificate following your company's process and generate a CA-signed certificate that you use for RosettaNet decryption and signing.

For testing with a self-signed certificate, here are simple steps to generate a key-pair using the Java `keytool`:

1. Generate public/private key pair using `keytool`.
   a. Specify any alias and a keystore file name, replacing `b2b-private-key-alias` and `b2b.jks` with your values.
   b. Enter a keystore password when prompted and note it.
   c. Enter your organization's information when prompted.

   This generates a key pair (a public key and associated private key) and self-signed digital certificate in a keystore. If the keystore does not exist, it is created.

   ```
   keytool -genkey -keyalg RSA -alias b2b-private-key-alias -validity
   1095 -keystore b2b.jks
   ```

2. Upload the JKS into Oracle Integration as the X.509 type (SSL transport) and Identity category using the same alias and password as entered above (this is part of Step 3 from the table of steps above).
3. Export the public key from this keystore as follows.
   a. Replace `b2b.jks`, `b2b-private-key-alias`, and *`public.cer`* with your keystore file name, alias that was used previously, and a file name to store the public certificate.

      ```
      keytool -export -keystore <b2b.jks> -alias <b2b-private-key-
      alias> -file <public.cer>
      ```

4. Convert it to any other industry-standard format using `keytool` as per your preference, if necessary. Share only the public certificate *`public.cer`* with your trading partner (never share the private key with anyone). Your trading partner uses the public key certificate for signature verification and encryption.

RosettaNet URL for Receiving

You need the RosettaNet URL for your RosettaNet endpoint to share with your trading partner. Once the transport is deployed (indicating it is ready to receive and/or send messages), your RosettaNet endpoint URL is displayed in the **RosettaNet endpoint URL for receiving** transport field. Copy this RosettaNet URL to share with your trading partner. This RosettaNet URL is not common across all trading partners; it is specific to the current trading partner that you are viewing or editing. Only that specific trading partner may send RosettaNet messages to this URL.

The RosettaNet URL is the URL to invoke the RosettaNet integration for receiving messages for this transport. While you can also get the same from the Integrations page, this provides an easier way to access it.

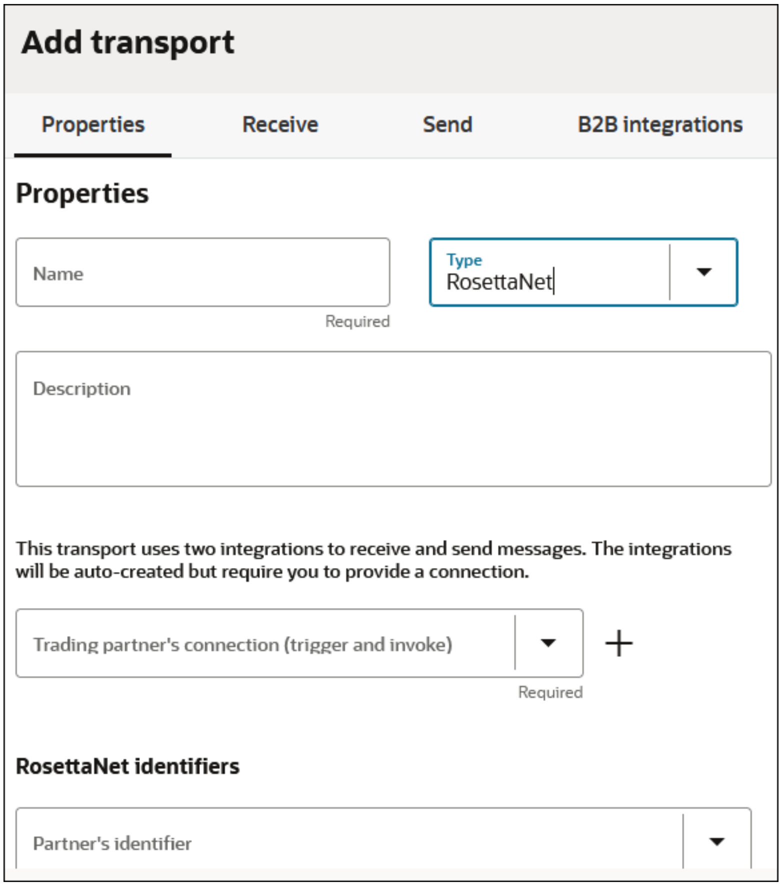
Define a RosettaNet Transport

Once you have collected RosettaNet transport details, you can define a RosettaNet transport in Oracle Integration.

1. In the navigation pane, click **B2B**, then **Trading partners**.
   The Trading partners page is displayed.
2. In the row of the trading partner for which to define transports, click **Edit** ✎.
3. Click **Transports & agreements**.

Define Transports to Create Integrations

1. In the **Transports** section, click **Add** + to define how a message is delivered to or received from this trading partner. The configuration panel is opened:



Add transport

Properties

Type: RosettaNet

This transport uses two integrations to receive and send messages. The integrations will be auto-created but require you to provide a connection.

RosettaNet identifiers

2. Define the following details.

Section	Description
Properties section • **Name**	Enter a name for the transport. The name is used for display purposes only.

Section	Description
• **Type**	Select **RosettaNet** from the drop-down list. This represents the communication protocol you use to exchange messages with your trading partner. Based on selecting RosettaNet, the appropriate configuration settings for the RosettaNet transport are displayed.
• **Description**	Enter an optional description of the transport. The description is used for display purposes only.
• **Trading partner's connection (trigger and invoke)**	Select an existing RosettaNet Adapter connection configured for connectivity to your trading partner or click **Add** + to create a new RosettaNet Adapter trigger or invoke connection on the Connections page. See Create a RosettaNet Connection in *Using the RosettaNet Adapter with Oracle Integration 3*. If you want to select another connection, you can do so when this transport is not deployed. Once you deploy the transport, the connection selection cannot be changed. You can modify the configuration properties inside the connection at any time. However, if you modify the connection settings after this transport is deployed, you must undeploy and then redeploy the transport for the changes to take effect.
• **Partner's identifier**	Specify a DUNS value that is used in the delivery header of the outbound payload. See Define B2B Identifiers.
• **Host identifier**	Specify a DUNS value that is used in the delivery header for inbound messages. See Host Profile.
• **Character encoding**	Select the character encoding to apply to all payloads processed through this transport. The character encoding is used at the EDI parsing (inbound) or EDI generation (outbound) step.
• **RosettaNet endpoint URL for receiving**	This is a display-only field. After the transport is deployed, indicating it is ready to receive and send messages, your RosettaNet endpoint URL is displayed. You may share this URL with your trading partner. This RosettaNet URL is not common across all trading partners. It is specific to the current trading partner that you are viewing or editing.
Send section • **Additional transport headers**	Manually enter any additional transport headers separated by a comma and a single blank space. For example: `header1`, `header2`, `header3`.
• **Signature**	If you want to enable message signing for outbound messages, select the appropriate signing algorithm from the drop-down list. For signing, you must configure the RosettaNet Adapter connection appropriately with certificates, as described in Step 3 of the above table. This field does not apply for inbound message processing. Signature verification of signed inbound messages is done automatically and there is no configurable option to enable or disable it. You must configure the RosettaNet connection with the right certificates for that to work also.

Section	Description
- **Encryption**	If you want to enable message encryption for outbound messages, select the appropriate encryption algorithm from the drop-down list. For encryption, you must configure the RosettaNet Adapter connection appropriately with certificates as described in Step 3 of the above table. This field does not apply for inbound message processing. Decryption of encrypted inbound messages is done automatically and there is no configurable option to enable or disable it. You need to configure the RosettaNet Adapter connection with the correct certificates for that to work also.
- **Compression**	Optionally select to compress the outbound message.
B2B Integrations section - **Integration name prefix**	Enter a short prefix that is used to form the complete integration names for receiving messages and sending messages. For the RosettaNet transport, it forms the integration names: ***your_prefix*** **RosettaNet Receive** and ***your_prefix*** **RosettaNet Send**. Details about these integrations are provided. See Create B2B Integrations for Receiving and Sending.
- **Integration identifier prefix**	Enter a short prefix that is used to form complete integration identifiers for receiving messages and sending messages. For the RosettaNet transport, it forms the integration identifiers: ***your_prefix*_RosettaNet_Receive** and ***your_prefix*_RosettaNet_Send**. The final integration identifier must be unique across all integrations. Therefore, ensure that you enter a prefix that is unique. If the uniqueness check fails, you get the opportunity to try with a different prefix.

3. Click **Add**.
   The new transport is displayed.
4. Select **Actions** •••, then select **Deploy**.
5. Select **Deploy** again when prompted.
   If successful, the following message is displayed.

   ```
   Transport transport_name was deployed successfully.
   ```

   The transport status is changed to **Active**.
6. Go to the Integrations page and note that both integrations are created and activated.
7. If you need to undeploy the transport, select **Actions** •••, then select **Undeploy**.
   Undeploying the transport also undeploys the integrations.

Collect REST Transport Details

You must configure REST Adapter connections to exchange inbound and outbound messages with a trading partner. You can create the connection either before or during configuration of the REST transport.

> **Note:**
>
> Only XML documents are currently tested and certified.

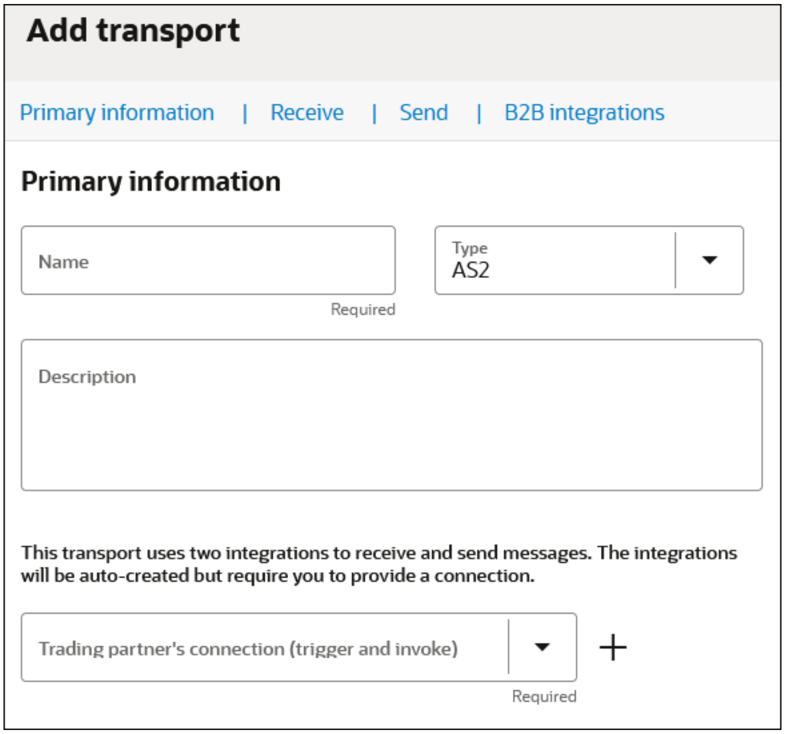
Define a REST Transport

Once you have collected REST transport details, you can completely define a REST transport in Oracle Integration.

1. In the navigation pane, click **B2B**, then **Trading partners**.
   The Trading partners page is displayed.
2. In the row of the trading partner for which to define transports, click **Edit** ✎.
3. Click **Transports & agreements**.

Define Transports to Create Integrations

1. In the **Transports** section, click **Add** + to define how a message is delivered to or received from this trading partner. The configuration panel is opened:



Add transport

Primary information | Receive | Send | B2B integrations

Primary information

Name (Required)

Type: AS2

Description

This transport uses two integrations to receive and send messages. The integrations will be auto-created but require you to provide a connection.

Trading partner's connection (trigger and invoke) (Required) +

2. Define the following details.

Section	Description
Properties section • **Name**	Enter a name for the transport. The name is used for display purposes only.
• **Type**	Select **REST** from the drop-down list. This represents the communication protocol you use to exchange messages with your trading partner. Based on selecting REST, the appropriate configuration settings for the REST transport are displayed.

Section	Description
• **Description**	Enter an optional description of the transport. The description is used for display purposes only.
• **Trading partner's connection (trigger and invoke)**	Select an existing REST Adapter connection configured for connectivity to your trading partner or click **Add** + to create a new REST Adapter trigger or invoke connection on the Connections page. **Note**: Only XML documents are currently tested and certified. See Create a REST Connection in *Using the REST Adapter with Oracle Integration 3*. If you want to select another connection, you can do so when this transport is not deployed. Once you deploy the transport, the connection selection cannot be changed. You can modify the configuration properties inside the connection at any time. However, if you modify the connection settings after this transport is deployed, you must undeploy and then redeploy the transport for the changes to take effect.
• **Character encoding**	Select the character encoding to apply to all payloads processed through this transport. The character encoding is used at the EDI parsing (inbound) or EDI generation (outbound) step.
Receive section • **REST transport URL for receiving**	This is a display-only field. After the transport is deployed, indicating it is ready to receive and send messages, your REST endpoint URL is displayed. You may share this URL with your trading partner. This REST URL is not common across all trading partners. It is specific to the current trading partner that you are viewing or editing.
Send section • **Enter Send URI**	Enter the send URI. This value is appended to the REST Adapter connection URL.
B2B Integrations section • **Integration name prefix**	Enter a short prefix that is used to form the complete integration names for receiving messages and sending messages. For the REST transport, it forms the integration names: ***your_prefix* REST Receive** and ***your_prefix* REST Send**. Details about these integrations are provided. See Create B2B Integrations for Receiving and Sending.
• **Integration identifier prefix**	Enter a short prefix that is used to form complete integration identifiers for receiving messages and sending messages. For the REST transport, it forms the integration identifiers: ***your_prefix*_REST_Receive** and ***your_prefix*_REST_Send**. The final integration identifier must be unique across all integrations. Therefore, ensure that you enter a prefix that is unique. If the uniqueness check fails, you get the opportunity to try with a different prefix.

3. Click **Add**.
   The new transport is displayed.
4. Select **Actions** •••, then select **Deploy**.
5. Select **Deploy** again when prompted.
   If successful, the following message is displayed.

```
Transport transport_name was deployed successfully.
```

The transport status is changed to **Active**.

6. Go to the Integrations page and note that both integrations are created and activated.
7. If you need to undeploy the transport, select **Actions** •••, then select **Undeploy**. Undeploying the transport also undeploys the integrations.

Collect FTP Transport Details

FTP is a file-based transfer protocol that requires an external FTP server. The FTP transport only acts as an FTP client that connects directly to your trading partner's FTP server or an FTP server hosted by your company (which the trading partner also connects to as a client). In either case, the FTP transport pulls files from (or pushes files to) an FTP server.

An FTP transport works on a time-based schedule when receiving files (as file polling), and in real-time when sending files. FTP is typically used in a batch mode (for example, processing a batch of purchase orders or a sales catalog on a nightly schedule).

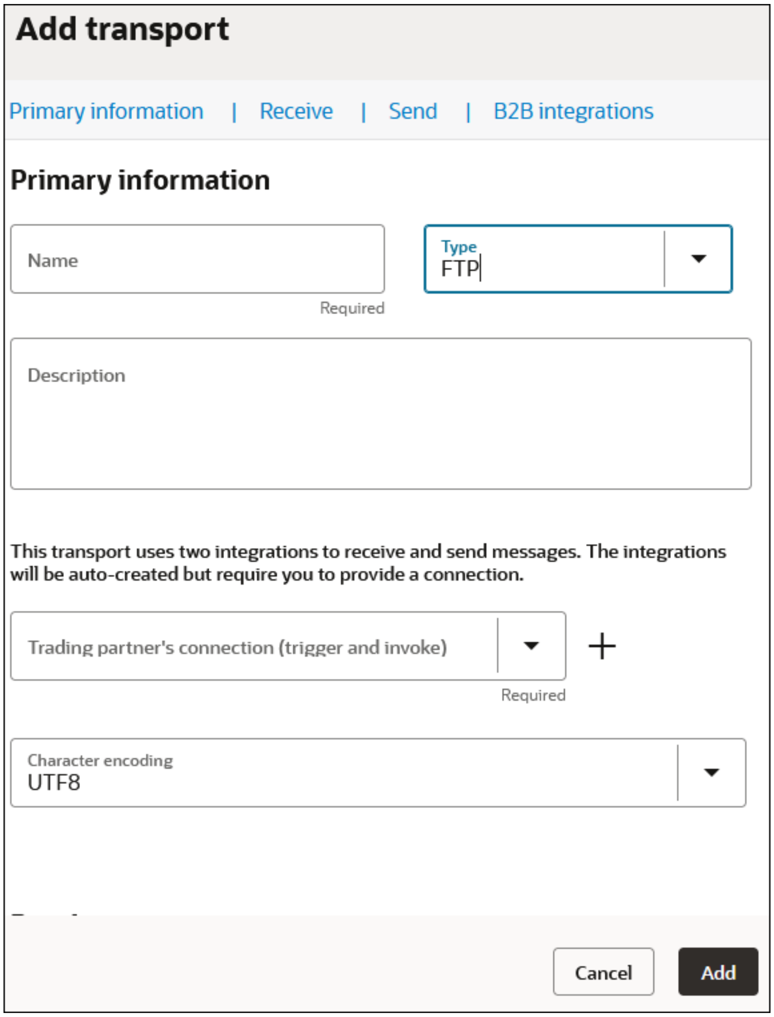
Before creating this transport, you need to create an FTP Adapter connection with the **Trigger and invoke** role. Review all prerequisites and details to create an FTP Adapter connection. See Create an FTP Adapter Connection in *Using the FTP Adapter with Oracle Integration 3*.

> **Note:**
>
> Note these FTP transport restrictions (although these features are provided in the FTP Adapter in standalone mode):
>
> - Connectivity to an on-premises FTP server through the connectivity agent is not currently supported.
> - PGP encryption and decryption are not currently supported.
> - Signing and signature verification for files is not currently supported.
> - Processing of `.zip` files is not currently supported.

Define an FTP Transport

Once you have collected FTP transport details, you can define an FTP transport in Oracle Integration.

1. In the navigation pane, click **B2B**, then **Trading partners**.
   The Trading partners page is displayed.
2. In the row of the trading partner for which to define transports, click **Edit** ✎.
3. Click **Transports & agreements**.
4. In the **Transports** section, click **Add** + to define how a message is delivered to or received from this trading partner. The configuration panel is opened:



Add transport

Primary information

Name
Required

Type
FTP

Description

This transport uses two integrations to receive and send messages. The integrations will be auto-created but require you to provide a connection.

Trading partner's connection (trigger and invoke)
Required

Character encoding
UTF8

5. Define the following details.

Section	Description
Primary Information section - **Name**	Enter a name for the transport. The name is used for display purposes only.
- **Type**	Select **FTP** from the drop-down list. This represents the communication protocol you use to exchange messages with your trading partner. Based on selecting FTP, the appropriate configuration settings for the FTP transport are displayed.
- **Description**	Enter a description of the transport. The description is used for display purposes only.

Section	Description
• **Trading partner's connection (trigger and invoke)**	Select an existing FTP Adapter connection to use or click **Add** + to create a new FTP Adapter connection on the Connections page. It must be a **Trigger and invoke** connection. **Trigger only** and **Invoke only** connections cannot be used for transports. See Create a Connection in *Using the FTP Adapter with Oracle Integration 3*. If you want to select another connection, you can do so when this transport is not deployed. Once you deploy the transport, the connection selection cannot be changed. You can modify the configuration properties inside the connection at any time. However, if you modify the connection settings after this transport is deployed, you must undeploy and then redeploy the transport for the changes to take effect.
• **Character encoding**	Select the character encoding to apply to all payloads processed through this transport. The character encoding is used at the EDI parsing (inbound) or EDI generation (outbound) step.
Receive section • **Input directory**	Specify a directory path on the FTP server to poll for files (inbound). For example: `/b2b/inbound` If the input directory is specified, the FTP transport is considered capable of receiving, and the direction indicates a darker up-arrow in the transports listing.
• **Scan recursively**	Select to scan all subdirectories to look for files. Deselect to only scan the immediate input directory, ignoring any subdirectories.
File name filter	Enter a wildcard expression that the FTP server understands to match files. For example, `*.edi`.
• **Minimum age (seconds)**	Specify a delay in seconds. The value specified indicates how long to ignore newly created files, relative to their creation time. For example, if a file is created at `11:02:00` and a minimum age of `60` seconds is specified, that file is ignored and not picked up for processing until `11:03:00`, when it becomes `60` seconds old. This delay allows the writer of a file to complete its transfer of bytes and avoid situations where not enough time was given to complete the file transfer. As a result, a half-written file was picked up for processing. If you specify a nonzero minimum age, make sure to also select an appropriate value for the FTP server time zone drop-down in the FTP Connections page. The time zone calculates the current age of a file, based on the current time and the creation time stamp of the file. If you do not select a time zone, then it defaults to the time zone of the Oracle Integration server. The mismatch can delay the processing of files for up to 12 hours. See Configure Connection Properties in *Using the FTP Adapter with Oracle Integration 3*.

Section	Description
• **Max count of files**	Specify the maximum number of files to be processed in one scheduled call. If the schedule is every hour, then every hour up to the maximum number of files are processed and any remaining files are picked up in a future run. See Create Scheduled Orchestration Integrations in *Using Integrations in Oracle Integration 3*. Limit this to a reasonable number so that the processing integration does not run for a very long time and consume precious resources just for one trading partner. The maximum value is 1000. The default is 100. **Note**: The file list is returned in a sorted order according to the last modified time. If you selected 10 as the maximum number of files and the last modified time of the eleventh file is the same as the tenth file, then the eleventh file is also added. This continues until you get a file with a different time stamp.
Send section • **Output directory**	Specify the directory path on the FTP server in which to put outbound files. For example: `/b2b/outbound` If the output directory is specified, the FTP transport is considered capable of sending, and the direction indicates a darker down-arrow in the transports listing. You can override the value you set for the outbound directory name during runtime. If not specified in the integration payload at runtime, the value falls back to that specified on this page. See Specify File Name and Directory Name Values at Runtime for Outbound FTP Transport Integrations.
• **Output file name**	Specify an output file name. The file name can include substitution patterns, for example, (`Out%SEQ%.edi` creates files with names like: `Out1.edi`, `Out2.edi`, and so on). The following patterns are supported (surround them inside `%` chars): • `SEQ` • `yyyyMMdd` • `MMddyyyy` • `yyMMddHHmmss` • `yyMMddHHmmssSS` • `yyMMddHHmmssz` • `yyMMddHHmmssSSz` **Note**: Use `%SEQ%` with caution because concurrent processing of messages may generate duplicate sequence numbers in some cases. This causes files to be overwritten. You can override the value you set for the output file name during runtime. If not specified in the integration payload at runtime, the value falls back to that specified on this page. See Specify File Name and Directory Name Values at Runtime for Outbound FTP Transport Integrations.
B2B integrations section • **Integration name prefix**	Enter a short prefix that is used to form the complete integration names for receiving messages and sending messages. For the FTP transport, it forms the integration names: ***your_prefix*** **FTP Receive** and ***your_prefix*** **FTP Send**. Details about these integrations are provided. See Create B2B Integrations for Receiving and Sending.

Section	Description
• **Integration identifier prefix**	Enter a short prefix that is used to form complete integration identifiers for receiving messages and sending messages. For the AS2 transport, it forms the integration identifiers: ***your_prefix*_FTP_RECEIVE** and ***your_prefix*_FTP_SEND**. The final integration identifier must be unique across all integrations. Therefore, ensure that you enter a prefix that is unique. If the uniqueness check fails, you get the opportunity to try with a different prefix.

> **Note:**
>
> There is one behavior not exposed in the transport configuration for FTP. It controls what should happen to an inbound file after processing is complete. You can alter this behavior by changing a property for the FTP receive integration. See FTP Receive Integration Postprocessing Behavior.

6. Click **Add**.
   The new transport is displayed.
7. If you selected FTP as the transport protocol, select **Actions** •••, then select **Receive schedule** to define a schedule.
8. Return to the **Transports & agreements** section. See Define the Integration Schedule in *Using Integrations in Oracle Integration 3*.
9. Select **Actions** •••, then select **Deploy**.
10. Select **Deploy** again when prompted.
    If successful, the following message is displayed.

    ```
    Transport transport_name was deployed successfully.
    ```

    The transport status is changed to **Active**.
11. Go to the Integrations page and note that both integrations are created and activated.
12. If you need to undeploy the transport, select **Actions** •••, then select **Undeploy**.
    Undeploying the transport also undeploys the integrations.

FTP Receive Integration Postprocessing Behavior

By default, files that are processed by the FTP Receive integration are moved to a backup directory. The backup directory name is assumed to be `original-input-directory_backup` and is created automatically provided the correct permissions are granted to create it.

You can customize this behavior as follows. To locate these properties, use the **Update property values** menu option as described below.

Permission	Description	Values
`Delete-Or-Keep-Processed-Files`	Controls whether processed files are retained or deleted.	• `deleteAlways`: Deletes a processed file regardless of whether processing succeeded or failed. • `keepOnError`: Deletes a processed file only on success. If the processed file failed, the file is retained and moved to a different directory. • `keepAlways`: (default) Retains a processed file, regardless of whether processing succeeded or failed by moving it to a different directory.
`Error-File-Extension`	If a processing error occurs, rename the file by appending the *`Error-File-Extension`* value (for example, `_error`), depending on the `Delete-Or-Keep-Processed-Files` setting. For example, if an input file name was `850_po.edi`, then the default value after processing the file is renamed to `850_po.edi_error`.	• `_error` (default). • You can set the value for `Error-File-Extension` to any string. However, the combination of *`original-input-filenameError-File-Extension`* must result in a valid directory path on the FTP server.
`Move-To-Directory`	Processed files are moved to another directory depending on the `Delete-Or-Keep-Processed-Files` value. The backup directory is assumed to be *`original-input-directoryMove-To-Directory`*. For example, if the input directory was `/b2b/inbound`, then with the default value, processed files are moved to the `/b2b/inbound_backup` directory.	• `_backup` (default). • You can set the value for `Move-To-Directory` to any string. However, the combination of *`original-input-directoryMove-To-Directory`* must result in a valid directory path on the FTP server.

Permission	Description	Values
`Processed-File-Extension`	If processing is successful, rename the file by appending the *`Processed-File-Extension`* value (for example, `_processed`), depending on the `Delete-Or-Keep-Processed-Files` setting. For example, if an input file name was `850_po.edi`, then with the default value, after processing, the file is renamed to `850_po.edi_processed`.	• `_processed` (default). • You can set the value for `Error-File-Extension` to any string. However, the combination of *`original-input-filenameError-File-Extension`* must result in a valid directory path on the FTP server.

To change the behavior for one specific FTP transport, locate its FTP Receive Integration, and navigate to the Update property values panel.

1. Hover over the row of the integration to update.
2. Select **Actions** ⋯, then select **Update property values**.
3. In the Update property values panel, select the property to change and enter a value in the **New value** field.
4. Select **Submit** to make the new values effective for future runs.
5. To change the behavior for all transports created in the future, edit the integration (for this example, named **B2B Integration Template FTP Receive**).
6. For the schedule action in the integration canvas, select **Actions** ⋯, then select **Edit integration properties** and change the default values.

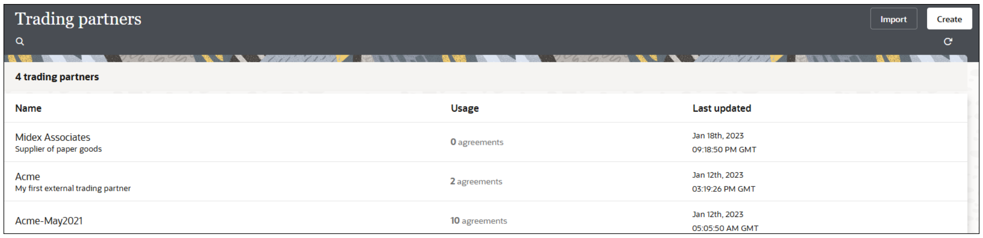
Create Agreements

This chapter providing details about creating and managing agreements. You define one or more agreements for a B2B trading partner with an intent to send or receive only certain types of business documents to or from that trading partner.

Detailed agreement concepts are provided. See Agreements.

You can view a list of inbound agreements and outbound agreements from the trading partner's **Transports & agreements** tab.

1. In the navigation pane, click **B2B**, then **Trading partners**.
   The Trading partners page is displayed.



4 trading partners

Name	Usage	Last updated
Midex Associates Supplier of paper goods	0 agreements	Jan 18th, 2023 09:18:50 PM GMT
Acme My first external trading partner	2 agreements	Jan 12th, 2023 03:19:26 PM GMT
Acme-May2021	10 agreements	Jan 12th, 2023 05:05:50 AM GMT

Note the following details:

- You cannot delete a trading partner with active agreements.
- Click **View** 👁 to view specific details about the trading partner, including any associated trading partner agreements.

2. In the row of the trading partner for which to create agreements, click **Edit** ✎.

Define Inbound and Outbound Agreements

1. Click **Transports & agreements**.
2. In the **Inbound agreements** section, click **Add** + to add a new agreement.
   a. Define the following details.

Field	Description
Name	Enter a name. This is only used for your reference. Your trading partner does not see any of the agreement names or configuration details you define.
Description	Enter an optional description.
Select a document	Select the type of document to receive. You can select an existing B2B document or create a new B2B document.
Select a backend integration	Select a backend integration to which to route this document after B2B processing. Either select one from the list or, if you know the identifier and version of your backend integration, enter it in the following format: `INTEGRATION_IDENTIFIER|01.00.0000` Backend integration concepts are provided. See Integrations Used for B2B Message Processing. Ensure that you have created a backend integration. See Create Backend Integrations. The drop-down list of integrations is filtered and only shows integrations that use the B2B action. It also currently shows all B2B system integrations. Do *not* select a B2B system integration. You must select a backend integration that you created separately to handle this type of document. A backend integration interfaces with your backend application, such as Oracle ERP Cloud. **Note**: Integrations specific to B2B for Oracle Integration can be displayed by entering `ediadapter` in the **Search** field on the Integrations page.

Field	Description
Enable validations	Select to perform syntactical validations on the received EDI payload and reject it if validation errors are found. If **Generate functional acknowledgment** is also enabled, the functional acknowledgment sent back to the trading partner conveys the acceptance or rejection, including any validation errors found. Deselect to skip syntactical validation checks and always accept the EDI payload.
Generate functional acknowledgment	Select to generate and send a functional acknowledgment back to your trading partner. • For EDI X12, a 997 document is generated. • For X12 HIPAA, a 999 acknowledgment document is generated. • For EDIFACT, a CONTRL document is generated to relay the outcome of the EDI translation. • For RosettaNet, a receipt acknowledgment is generated. The generated functional acknowledgment is automatically routed to the B2B Integration for sending messages that is linked to the transport from which the incoming document was received. Deselect to not generate or send functional acknowledgments. This setting requires your company and trading partner to mutually decide up front whether or not to use functional acknowledgments. A problem scenario is when one party expects these acknowledgments, but the other party does not send them.

Field	Description

Enable checks for duplicate control numbers	Select to check for duplicate control numbers in B2B transactions in inbound agreements. This prevents processing of transactions with duplicate control numbers. For example, if duplication exists between the control numbers in two transactions, the first message is successfully processed because the number is unique, but the second transaction is caught and not processed. Transaction failure is visible in the **Message logs** section of the transaction on the Track B2B messages page. For example, here is part of the message for a duplicate control number that was caught and not processed:
	`This EDI transaction with`
	`document type [850], version`
	`[4010]`
	`was not translated because it`
	`was`
	`determined to be a duplicate of`
	`a`
	`transaction.`

b. Click **Add**.

c. From the **Actions** ⋯ menu, select **Deploy**.

d. Select **Deploy** again when prompted.
The inbound agreement status is changed to **Active**.

e. If you need to undeploy the agreement, select **Undeploy** from the **Actions** ⋯ menu.

f. Click **Edit** ✎ to make any updates.

3. In the **Outbound agreements** section, click **Add** + to add a new agreement.

a. Define the following details.

Field	Description
Name	Enter a name. This is only used for your reference. Your trading partner does not see any of the agreement names or configuration details you define.
Description	Enter an optional description.
Select a document	Select the type of document to send. You can select an existing B2B document or create a new B2B document.

Field	Description
Select identifiers	Certain mandatory and optional identifiers, such as **EDI Interchange ID** or **DUNS** (for RosettaNet), are inserted into EDI envelope segments during the outbound translation. Select the identifiers you want inserted into the envelopes. See Define B2B Identifiers and Define Identifiers in the Host Profile. • **Select trading partner identifiers**: Identifies the trading partner receiving the document. Select trading partner identifiers that you want to insert as the receiver-side values (for example, Interchange Receiver ID). • **Select host identifiers**: Identifies the host sending the document. Select host identifiers that you want to insert as the sender-side values (for example, Interchange Sender ID).
Select a transport	Select a transport to route messages processed through the current outbound agreement to that transport for final delivery to the external trading partner. Selecting a transport defines a routing rule that controls how a message gets routed from a backend integration to a B2B integration for sending messages. Details about how outbound messages are routed are provided. See Message Routing Between Integrations.
Enable validations	• Select to perform syntactical validations on the canonical XML payload during the outbound EDI translation. If validation errors are detected at this step, it means that the output EDI is syntactically invalid and must not be sent to the trading partner. Therefore, it is rejected during the translation phase and not routed to the B2B integration for sending messages. • Deselect to skip syntactical validation checks and send the EDI payload to the trading partner even if it has errors.

Field	Description
Functional acknowledgment required	• Select to require (and expect) that your trading partner always send back functional acknowledgments. When an acknowledgment is required, an outbound business message goes into a **Pending functional acknowledgment** state after transmission and is marked as **Successful** or **Failed** only when an acknowledgments is received. • Deselect to not require (or expect) functional acknowledgments. Outbound business messages are immediately marked as **Successful**. This setting requires your company and your trading partner to mutually decide up front whether or not to use functional acknowledgments. A problem scenario is when one party expects these acknowledgments, but the other party does not send them.

b. Click **Add**.

c. From the **Actions** ••• menu, select **Deploy**.

d. Select **Deploy** again when prompted. The outbound agreement status is changed to **Active**.

e. If you need to undeploy the agreement, select **Undeploy** from the **Actions** ••• menu.

f. Click **Edit** ✎ to make any updates.

Life Cycle Actions for Agreements

Click **Actions** ••• on a row to view available actions.

Life cycle actions for agreements are:

- Create an agreement: Adds the definition in design-time only (but unless deployed, the new agreement is not enforced at runtime).
- Deploy an agreement: Makes the agreement visible for runtime processing and is immediately enforced.
- Redeploy an agreement: Applies configuration changes to the runtime on-the-fly without disrupting message processing.
- Undeploy an agreement: Hides the agreement from runtime processing, making it no longer effective, starting immediately.
- Delete an agreement: Removes it from design-time.

Updating Values and Applying Changes To Runtime

You can update existing agreement settings at any time. However, the changes are not applied to the runtime processing until the agreement is redeployed.

Redeployment is a life cycle action available for agreements. It applies changes to the runtime, on-the-fly, without disruption to message processing.

Changes to Identifier Values

Even if you don't directly change the agreement's settings, you cause an indirect change to an agreement if you modify values for an identifier. Select **Trading partner**, then **B2B identifiers** or **Host profile**, and then **Identifiers**.

To apply indirect changes, redeploy the agreement.

There is one special case for inbound agreements, because B2B identifiers are implicitly used by inbound agreements:

- If you add, update, or delete trading partner's identifiers, redeploy any of the agreements to apply the identifier changes to runtime.

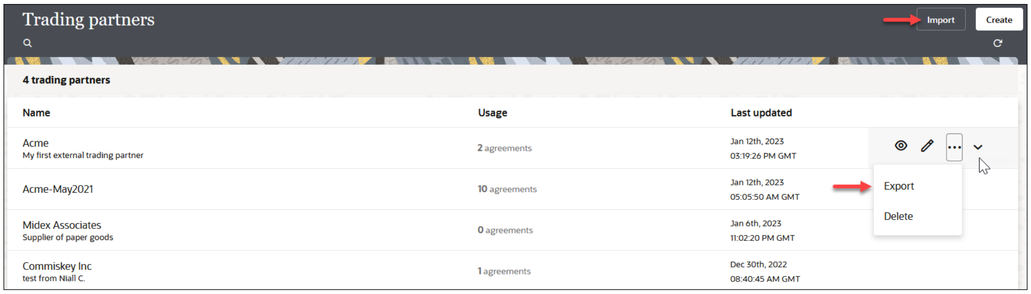
Export and Import a Trading Partner

You can export an individual trading partner and import it into another instance of Oracle Integration. Importing a trading partner also imports the associated agreement references (for example, documents, schemas, and identifier references).

> **Note:**
>
> Ensure that you also separately import any backend integrations referenced in the trading partner agreement.

Export a Trading Partner

You can export or import a trading partner definition from the trading partners listing page.



4 trading partners

Name	Usage	Last updated
Acme My first external trading partner	2 agreements	Jan 12th, 2023 03:19:26 PM GMT
Acme-May2021	10 agreements	Jan 12th, 2023 05:05:50 AM GMT
Midex Associates Supplier of paper goods	0 agreements	Jan 6th, 2023 11:02:20 PM GMT
Commiskey Inc test from Niall C.	1 agreements	Dec 30th, 2022 08:40:45 AM GMT

1. In the navigation pane, click **B2B**, then **Trading partners**.
2. From **Actions** •••, select **Export**.
3. Download the ZIP file.

Import a Trading Partner

1. In the navigation pane, click **B2B**, then **Trading partners**.
2. Click **Import**.
3. If you want to overwrite an existing trading partner agreement of the same name, click **Overwrite the trading partner definition, if it exists**.

This selection overwrites the trading partner and all associated artifacts (identifiers and agreements) when there are no active agreements. Associated documents and schemas are also overwritten if they do not have any active agreement references for other trading partners.

If you select to overwrite, you must click the checkbox explicitly for the import to overwrite the existing trading partner, associated documents, and schemas. If the checkbox is not selected, the artifact import is skipped (that is, the document and schemas are skipped during the import if they already exist in the system).

If these documents and schemas have other references (that is, if another agreement for a different trading partner refers to the same document or schema), then regardless of whether you selected to overwrite, the import action is skipped for the document and schemas. Artifacts can be overwritten only if they do not have any other dependencies. For example, a schema cannot be overwritten because it is referenced by another B2B document used by another trading partner.

4. If import is successful, a message is displayed:

```
B2B Trading Partner import successfully for trading_partner_name. More
Information.
```

5. Click **More information** to view the Trading partner import report, which provides details about what was overwritten and not overwritten.
6. Click **Close**.
7. Click **Edit** ✎, then click **Transports & agreements**.
8. Verify that all references are intact (for example, if any backend integrations require importing and reactivating) and make any necessary updates.
9. Redeploy the agreements.

Create B2B Integrations for Receiving and Sending

When you add a transport to a trading partner, two integrations are automatically created for you: the B2B integration for receiving and the B2B integration for sending (these are not their exact names, but rather indicative of the concept).

Both integrations are front-end integrations because they directly interface with the external trading partner. These integrations are at the heart of the transport for its runtime functioning.

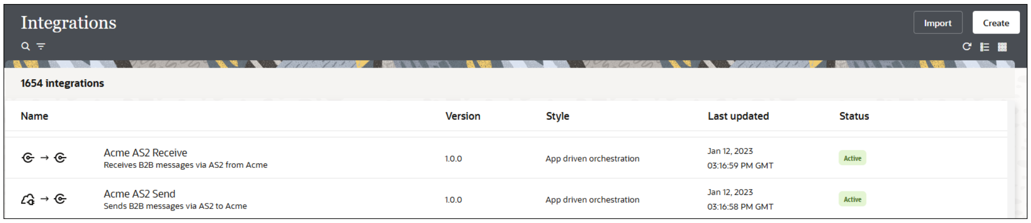
These integrations are automatically created when you define a transport. Two master template integrations are cloned internally to create these two integrations for the transport. The names of these master template integrations are as follows:

- B2B Integration template FTP Receive and B2B Integration template FTP Send: Used as the master template for the FTP transport.
- B2B Integration template AS2 Receive and B2B Integration template AS2 Send: Used as the master template for the AS2 transport.
- B2B Integration template RosettaNet Receive and B2B Integration template RosettaNet Send: Used as the master template for the RosettaNet transport.
- B2B Integration template REST Receive and B2B Integration template REST Send: Used as the master template for the REST transport.

The master template integrations are included with B2B for Oracle Integration. They are not initially present in your Oracle Integration instance, but created on-demand when you create your first transport.

When you create a transport, you have some control over the actual names of these B2B integrations. In the transport settings, you are prompted for an integration name prefix and an integration identifier prefix. These prefixes are used to generate the actual names and identifiers for the B2B integrations.

For example, the automatically created B2B integrations look as follows when the name prefix entered is `Acme`.



1654 integrations

Name	Version	Style	Last updated	Status
Acme AS2 Receive Receives B2B messages via AS2 from Acme	1.0.0	App driven orchestration	Jan 12, 2023 03:16:59 PM GMT	Active
Acme AS2 Send Sends B2B messages via AS2 to Acme	1.0.0	App driven orchestration	Jan 12, 2023 03:16:58 PM GMT	Active

In general, you don't need to edit or manage the B2B integrations directly because most functionality is driven through lifecycle actions on the transport.

> **Note:**
>
> You may customize the B2B integrations. However, in future versions, Oracle may provide enhancements to the master template integrations and a way to upgrade a transport, which overwrite the B2B integrations so as to leverage new features. You may have to add any customizations again into the integrations.

There are a few cases where you may need to do some management directly on the B2B integrations, specifically for the FTP transport.

1. View Past and Future Scheduled Runs for an FTP Receive Integration - refer to various other functions you can perform. See Create Scheduled Orchestration Integrations in *Using Integrations in Oracle Integration 3*.
2. Use submit now to trigger ad-hoc runs and make the FTP Receive integration execute immediately to poll for files.
3. Change the default behavior for what should happen to files, in case of FTP Receive, after the inbound processing is completed. See FTP Receive Integration Postprocessing Behavior.

Create Backend Integrations

You create one backend integration for each unique document definition that you want to receive or send. There are two types of backend integrations, depending on the direction:

- Inbound backend integration: Handles documents received from a partner.
- Outbound backend integration: Handles documents sent to a partner.

This section explains how many of these integrations you need to create. Learn the concepts behind backend integrations. See Integrations Used for B2B Message Processing.

Let's take an example scenario:

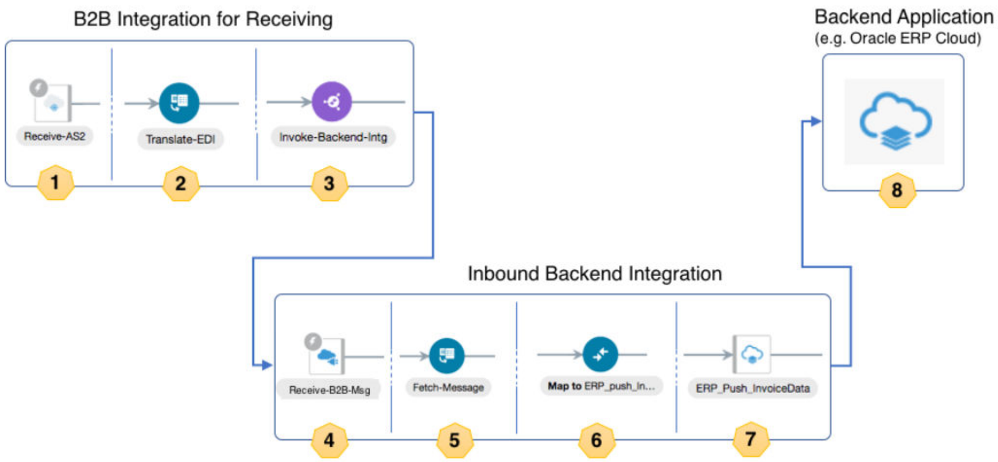
Document Types	How Many Backend Integrations You Need
You are expecting to receive the following types of inbound documents from various trading partners: • Purchase order acknowledgments • Invoices • Advanced shipment notices	You need three inbound backend integrations. (You can collapse the three into a single backend integration, if that is your design preference.)
You are expecting to send the following types of outbound documents to various trading partners: • Purchase orders • Inventory inquiry	You need two outbound backend integrations. (You can collapse the two into a single backend integration, if that is your design preference.)

Let's take another variant of the first scenario:

Document Types	How Many Backend Integrations Do You Need
You are expecting to receive the following types of inbound documents from various trading partners: • Purchase order acknowledgments (uses the standard X12 schema) • Invoices (uses the standard X12 schema) • Invoices (uses a customized X12 schema specifically for two trading partners, X and Y) • Advanced shipment notices (uses the standard X12 schema)	You need four inbound backend integrations because you actually get the invoices in two different formats, one that is X12 standard and another that is custom. (You can collapse the four into a single backend integration, if that is your design preference.)
You are expecting to send the following types of outbound documents to various trading partners: • Purchase orders (uses the standard X12 schema) • Purchase orders (uses a customized X12 schema specifically for two trading partners, X and Y) • Inventory inquiry (uses the standard X12 schema)	You need three outbound backend integrations because you are generating purchase orders in two different formats, one that is X12 standard and another that is custom. (You can collapse the three into a single backend integration, if that is your design preference.)

Inbound Message Processing

This section describes the high-level steps involved in processing an inbound B2B message.



Step	Description
1	A message sent or a file dropped in an FTP location by your trading partner arrives at the adapter endpoint (AS2 or FTP). It receives and unpacks the message.
2	The message is translated from EDI to a canonical XML format and persisted in the Oracle Integration persistence store. A unique ID is assigned to it.
3	Based on the current message type and the inbound agreements defined for the trading partner, an appropriate inbound backend integration is triggered. The message ID is handed to it.
4	The backend integration instance starts and receives the message ID at its REST trigger endpoint.
5	The backend integration, given the message ID, retrieves the translated canonical XML message from the Oracle Integration persistence store. It uses the B2B fetch message operation to retrieve it.
6	The canonical XML is further transformed to a backend application message.
7	Using an application adapter, the backend application message is sent to your backend application.
8	Your backend application now consumes the business transaction sent by your trading partner. Further processing is performed.

Design an Inbound Backend Integration

An inbound backend integration is triggered automatically by the B2B integration for receiving messages using a local integration invoke (that is, call an integration) action. For it to be triggered correctly, the backend integration must adhere to an API contract that requires:

- The integration must have a REST Adapter trigger configured for OAuth 2.0.
- The REST Adapter trigger must use / as the resource URI (that is, a root resource URI).
- The REST Adapter trigger must use a specific request payload schema, given in step 1(e).
- The REST Adapter trigger must not return a response; it must be asynchronous. This allows the B2B integration for receiving messages to be unblocked quickly and continue processing, even though the backend integration may take time to process.

This backend integration is given a collection of message IDs, in the form of repeating **b2b-message-reference** elements. A collection is necessary when your trading partner sends a batched message (that is, in one message or file, there may be multiple business documents). The B2B integration for receiving messages automatically splits the message into multiple documents and returns one **b2b-message-reference** element for each one. A maximum chunk of 200 records is provided at a time to the backend integration. If the inbound message has more documents, the backend integration may be called multiple times with chunks of 200.

1. Create the integration and configure the REST Adapter trigger.
    a. Select an application pattern. Give it any name. For this example, `Backend - Inbound Purchase Orders` is used).
    b. Add a REST Adapter trigger that uses the OAuth 2.0 Or Basic Authentication security policy.
    c. Enter `Receive-B2B-Msg` (or any other name) as the name of the trigger connection.
    d. On the Resource Configuration page, enter `/` as the resource URI, select the **POST** action, and enable the **Configure a request payload for this endpoint** check box.
    e. On the Request Parameters page, select **JSON Sample** and paste the JSON provided below as an inline sample.

```
{
  "type": "MSG",
  "id": "12345",
  "direction": "INBOUND",
  "trading-partner": "ACME",
  "document-definition": "PO_850",
  "message": [
    {
      "b2b-message-reference": "biz:0AC400D117503A8246000000347849EB"
    },
    {
      "b2b-message-reference": "biz:0AC400D117503A8246000000347849EA"
    }
  ]
}
```

> **Note:**
>
> In the above JSON sample, the structure is important. The values are just placeholders or representative values.

    f. Click **Continue** to access the Summary page to review your selections, then click **Finish**.
2. Place a for-each action after the REST Adapter trigger.
3. Select **request-wrapper** > **message** as the **Repeating Element** and enter `Current-Msg` as the **Current Element Name** value.
4. Add a scope inside the for-each action and name it (for this example, named `Handle-One-Message`).

5. Add a B2B action configured with a Fetch Message operation inside the scope. The B2B action appears in the palette under **Actions**.
   1. Enter a name (for this example, named `Fetch-Message`), select **B2B Trading Partner mode**, and click **Continue**.
   2. Select **Inbound** as the direction and **Fetch Message** as the operation. Click **Continue**.
   3. On the Select Data Formats page, select a document definition from the drop-down list that this backend integration is handling.
      Existing B2B documents are displayed in the drop-down list for selection. Alternately click **Search** to select a B2B document by document standard, version, and type.
   4. Click **Continue** once you make the selection.
   5. In the Summary Page, review your selections, and click **Finish**.
   6. Configure the mapper to **Fetch-Message**.
      Expand **Current-Msg** and map its **B2b Message Reference** to the **B2B Message Reference** of **FetchMessageInput**.
   7. Close and apply changes to the mapper.
6. Add a scope level fault handler.
   This handler is added so that if the message fails to be processed by the backend application, the corresponding transaction is marked as **Failed** in the **B2B Tracking** > **Business Messages** view (instead of **Success**).
   1. Click **Fault Handler** > **Default Handler** for the scope. Initially it is empty.
   2. Place a B2B action inside the fault handler and name it (for this example, `Mark-As-Error`). Click **Continue**.
   3. Select **Inbound** as the direction and **Mark As Error** as the operation. Click **Continue**.
   4. On the Summary page, review your selections, and click **Finish**.
   5. Configure the mapper to **Mark-As-Error** by mapping the following elements:
      - Source **Current-Msg > Message** > **B 2b Message Reference** to target **MarkAsErrorInput** > **B2B Message Reference**.
      - Source **Handle-One-MessageFaultObject** > **errorCode** to target **MarkAsErrorInput** > **Error Code**.
      - Source **Handle-One-MessageFaultObject** > **reason** to target **MarkAsErrorInput** > **Reason of the error**, and in the Expression Builder, enter:

        ```
        concat('Failed to send the message to the backend
        application, cause: ', $Handle-One-MessageFaultObject/
        nsmpr0:fault/nsmpr0:reason)
        ```

      - Source **Handle-One-MessageFaultObject** > details to target **MarkAsErrorInput** > **Error Details**.
   6. Save and apply the mapping.
   7. Exit the fault handler.
7. Add actions to call a backend application.

Add one or more actions inside the scope to send the business message to a backend application.

In the example above, a REST Adapter invoke connection sends the message to a backend application. To use with your specific backend application, you have many application adapters provided in Oracle Integration that can interface with several popular backend applications. You can also use technology adapters, including REST, SOAP, JMS, AQ, File, and so on.

The **Fetch-Message Response** (EDI Translation Adapter) provides you with a B2B canonical XML from which to map. Its **edi-xml-document** is the key element that contains the canonical form of the inbound EDI document. See Schema Elements for Inbound EDI.

You must design the data mapping to prepare the message prior to the backend invoke. This can be a complex task. On the left-hand side is the B2B canonical XML format, represented by the **edi-xml-document**. On the right-hand side (not shown below) is your backend application schema for the business document. You must create mappings for the elements on the left-hand side (B2B canonical XML) to the right-hand side (backend application schema). The mappings cannot be generalized since the right-hand side is specific to a target backend application.

8. Add identifiers for integration tracking.
   Select fields from the input schema for integration tracking to complete the integration.
9. Save and then activate the backend integration.

Design a Backend Integration to Handle Multiple Types of Documents

The detailed steps assume you create one backend integration for each document definition that you want to handle. You can clone the integration for other document types because the basic pattern is identical.

If you want to design a single integration that handles multiple document types, add a switch action in your integration and specific routes for each document definition or trading partner.

Each route has its own B2B Fetch Message operation, a data mapping, and an invocation call to the backend application. This design pattern is a trade-off between ease of development and operations. For example, if you need to fix a mapping for your purchase order document, you must:

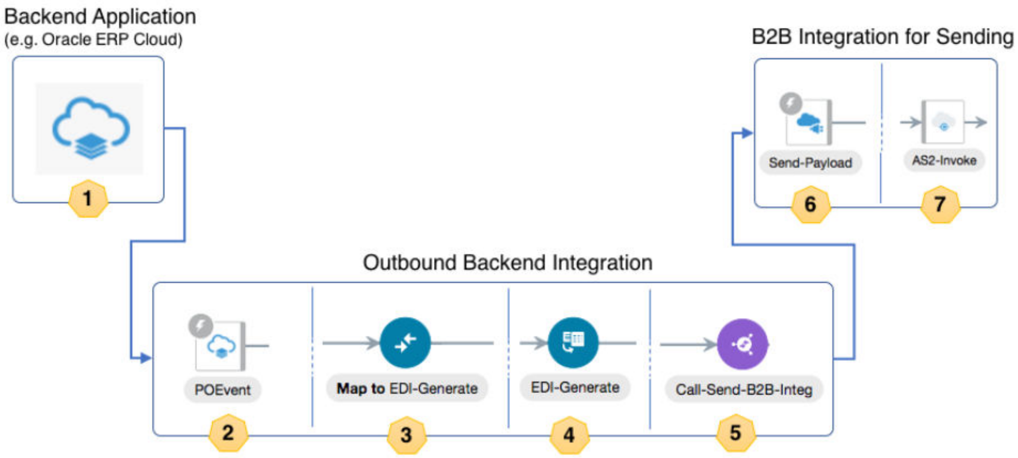
- Deactivate the backend integration that handles the purchase order document.
- Fix the mapping.
- Reactivate the integration.

If you had separate backend integrations for each document type, you are isolating the impact to one specific integration. Whereas, if everything is built into a single integration, you have a wider impact in a production deployment, for example.

Outbound Message Processing

This section describes the high-level steps involved in processing an outbound B2B message.

The high-level steps involved in processing an outbound B2B message are as follows.



Step	Description
1	Your backend application has a requirement to send a business transaction to an external trading partner. It triggers your outbound backend integration by sending it a notification message.
2	Your backend integration instance receives the notification that includes the application message in the backend application message format.
3	Using a mapper, the backend application message is transformed into a B2B canonical XML format.
4	The canonical XML message is provided to a B2B action for outbound translation (the action named **EDI-Generate** above). A trading partner is specified as an input to the B2B action. The B2B action translates the canonical XML message to a native EDI format (X12 or EDIFACT) and persists it in the Oracle Integration persistence store. A unique ID is assigned to it.
5	Based on the target trading partner, the current document type, and the outbound agreements defined for the trading partner, an appropriate B2B integration for sending messages is triggered. The message ID is handed to it.
6	The B2B integration for sending messages instance starts and receives the message ID at its REST Adapter trigger endpoint.
7	The B2B integration for sending messages instance uses an adapter (AS2 or FTP) to pack the message and then transmit it to the external trading partner through the AS2 or FTP protocol.

Design an Outbound Backend Integration

An outbound backend integration is triggered by a backend application. For sending a message to an external trading partner, this integration must know exactly which trading partner the message must go to. This is specified either directly or indirectly by the backend application in either of two ways:

1. The trading partner's identifier is specified. It is the same identifier you entered when you created a new trading partner in B2B, displayed in the Primary Information page.
   or
2. An application partner ID is specified. This is a type of B2B identifier you can add to a trading partner.

The idea behind the Application Partner ID is the following: the backend application and the B2B system both model the concept of a trading partner. A backend application may treat such a business entity in a specific role, such as a supplier, vendor, purchaser, and so on. The business entity is assigned a unique identification in the backend application. However, in B2B, you again register this entity as a B2B trading partner for message exchange purposes. Your suppliers, vendors, purchases are all treated the same: as B2B trading partners. At the hand-off point between the backend application and B2B, one of these systems must map between the backend application's entity ID and the B2B system's trading partner ID. The application partner ID is exactly for this purpose; it allows you to add the backend application's entity ID into the B2B system.

For this to work, add the Application Partner ID as a B2B identifier type to all of your trading partners. The value is the unique identification defined in your backend application. Once you define these, the B2B system knows how to look up a trading partner, given the value for Application Partner ID.

Step 3 specifies how you map the application partner ID element in the TranslateInput schema.

1. Create the integration and configure the trigger connection.
   Depending on how you want a notification from the backend application to arrive, configure a trigger connection using an appropriate application adapter or a technology adapter.

   In our example, a simple REST Adapter trigger is used. Replace it with Oracle ERP Cloud Adapter, Oracle NetSuite Adapter, or another one.
2. Place a B2B action with an outbound translate operation.
   a. Add a B2B action into the integration. The B2B action can be seen in the palette under **Actions**.
   b. Enter a name (for this example, `EDI-Generate` is added). Click **Continue**.
   c. Select **Outbound** as the direction and **Translate** as the operation. Click **Continue**.
   d. In the Select Data Formats page, select a document definition from the drop-down list for this backend Integration to handle. Click **Continue** after you make the selection. Existing B2B documents are displayed in the drop-down list for selection. Alternately, click **Search** to select a B2B document by document standard, version, and type.
   e. In the Summary Page, review your selections and then click **Finish**.
3. Configure the mapper to convert the application message to B2B canonical format.
   This mapping can be complex. Let us break it down into two parts.
   - In the first part, map an element to either **B2B Trading Partner ID** or **Application Partner ID**. These elements are used to specify the trading partner to which to send. This concept was explained earlier. In the example shown, only the **B2B Trading Partner ID** has been mapped.
     Also, optionally map **Application Message ID**. This is a message ID assigned by the backend application, if any. If this is mapped, the value is displayed in **Monitoring** > **B2B Tracking** > **Business Messages** in the details panel for any outbound message. Its value is not used during processing, but is stored and is displayed for your reference only.
   - The second part of the map is for **edi-xml-document**, which is the B2B canonical XML format for an EDI document, shown on the right-hand side of the map. On the left-hand side is your backend application schema for the business document. Because that schema is very specific to your backend application, the mappings from the backend application schema to the B2B canonical XML cannot be generalized.
4. Add a check for successful EDI translation.

   a. Place a switch action after the B2B translate (that is, **EDI-Generate**) action. This action is for checking translation status to find if the EDI translation succeeded or failed.

   b. Add a route, called **Translation Succeeded**, and configure the following expression:

      ```
      translation-status = "Success" or translation-status = "Warning"
      ```

5. If translation is successful, prepare to deliver the document to the trading partner.

   a. Add a local integration invoke (that is, call an integration) action in the **Translation Succeeded** route.

   b. On the Select Integration page, select any available B2B Integration for sending messages (the exact name of such an integration varies, but the name always ends with **AS2 Send** or **FTP Send**). In the following example, the selected integration, **Ext TP FTP Send (1.0)** is a B2B integration for sending messages. While the selection must be a B2B integration for sending messages because of a specific API contract, it does not matter which specific trading partner it is, since this selection is overridden from the mapper as explained in Step c .

   c. Click **Continue** several times, and then click **Finish**.

   d. Configure the mapper before the local integration invoke (that is, map to **Send-To-Partner**) as follows:

      - Map source **TranslateOutput** > **B2B Message Reference** to target **components.schemas.request-wrapper** > **messages** > **b2b-message-reference**.
      - Map source **TranslateOutput** > **Trading Partner Name** to target **components.schemas.request-wrapper** > **trading-partner**.
      - Map source **TranslateOutput** > **connectivity-properties-code** to target **ConnectivityProperties** > **LocalIntegration** > **code**.
      - Map source **TranslateOutput** > connectivity-properties-version to target **ConnectivityProperties** > **LocalIntegration** > **version**.

      The mappings for **ConnectivityProperties** > **LocalIntegration** > **code** and **version** override the selection of an integration in Step 5(b)).

      Here's how the routing works:

      - The B2B action checks the trading partner's outbound agreements and finds one that matches the B2B document selected in step 2(d).
      - The transport linked to the outbound agreement is found.
      - Once the transport is known, the B2B integration for sending messages linked to the transport is also known. That's the integration to call for delivering the document to the trading partner. The B2B action populates the **connectivity-properties-code** and **connectivity-properties-version** with that integration's identifier and version.
      - If you have mapped the fields correctly, the local integration invoke (that is, calls an integration) action honors the mapper override and calls the integration specified in the **connectivity-properties-code** and **connectivity-properties-version**, instead of the one selected in Step 5(b).

6. Add a return action for success and a fault for an error.
    a. Add a return action for the integration to return a successful response for the **Translation Succeeded** route and a fault return for the **Otherwise** route.
    This step is specific to the example selected with the REST Adapter trigger. Your case may vary. Depending on your backend application, there may or not be a mechanism to relay the outcome of a successful or a failed translation back into the backend application.
7. Add B2B business identifiers for integration tracking.
    a. Select fields from the input schema for integration tracking to complete the integration. This is largely specific to your backend application schema.
    b. Save, and then activate the backend integration.

Design a Backend Integration to Handle Multiple Types of Documents

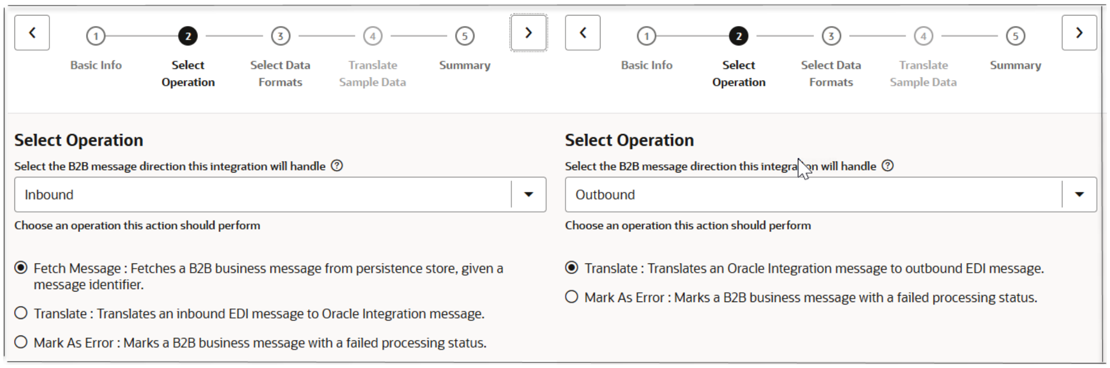
The detailed steps assume you create one backend integration for each document definition you want to handle. You may clone the integration for other document types because the basic pattern is identical.

Similar to an inbound backend integration, you may design a single integration that handles multiple document types by adding a switch action in your integration and specific routes for each document definition or trading partner and their own map and B2B translate action.

Use the B2B Action In Trading Partner Mode

This section describes the operations provided by the B2B action during trading partner mode configuration.

The following operations are available in the inbound and outbound directions.



Select Operation

Select the B2B message direction this integration will handle ⓘ

Inbound

Choose an operation this action should perform

● Fetch Message : Fetches a B2B business message from persistence store, given a message identifier.

○ Translate : Translates an inbound EDI message to Oracle Integration message.

○ Mark As Error : Marks a B2B business message with a failed processing status.

Select Operation

Select the B2B message direction this integration will handle ⓘ

Outbound

Choose an operation this action should perform

● Translate : Translates an Oracle Integration message to outbound EDI message.

○ Mark As Error : Marks a B2B business message with a failed processing status.

Of these five operations, you must use the three highlighted below in your backend integrations. Those are the only ones you need to more fully understand. The remaining operations are used in the B2B transport integrations (that is, B2B integrations for receiving messages and sending messages). Because those integrations are automatically created, you don't need to understand their usage in as much detail.

Inbound Direction

Operation	Used By	Purpose
Fetch Message	Inbound backend integration	This operation retrieves an already processed B2B business message from the Oracle Integration persistence store. It outputs the B2B canonical XML format for a business message, given **b2b-message-reference** as input. The canonical XML format is represented by the **edi-xml-document** element. It is accessible inside an inbound backend integration. You use the mapper to transform it into a backend application format. You must select a specific B2B document during design time when you configure this operation. At runtime, it only retrieves a compatible document. If this operation is given a **b2b-message-reference** for a different B2B document, an error occurs (for example, if the fetch message is configured for a purchase order and at runtime it was asked to retrieve an invoice).
Translate	B2B integration for receiving messages	The B2B integration for receiving messages uses this operation for parsing and debatching an inbound EDI message into B2B canonical XML format, represented by the **edi-xml-document** element. One inbound EDI message may produce multiple B2B business messages (each one having a separate canonical XML document). The action outputs a collection of repeating **b2b-message-reference** elements, each containing an internal message ID of one business message. The canonical XML format is accessible inside the integration with the fetch message operation.
Mark As Error	Inbound backend integration	This operation provides for more robust error handling, in case of failures. This operation updates a B2B business message and reflects the failure to process this message by the backend integration, if an error occurs.

Outbound Direction

Operation	Used By	Purpose
Translate	Outbound backend integration	You must use this action directly within your outbound backend integrations. An outbound backend integration uses this operation to translate from a B2B canonical XML format to an EDI format. The EDI format cannot be accessed inside the integration directly. Instead, an internal message ID is assigned that is returned in the element **b2b-message-reference**. You can view or download the EDI-formatted payload. 1. In the navigation pane, click **Observability**, then **B2B tracking**. 2. Select **Business message**.
Mark As Error	B2B integrations for receiving and sending messages	This operation provides for more robust error handling, in case of failures. This operation updates a B2B wire message and reflects the failure to process this message by the B2B integration for sending messages. For example, for the FTP sending messages integration, if the file write operation fails, this operation updates the wire message as failed. There is also a similar error condition that can occur while sending back a functional acknowledgment in the B2B integration for receiving.

3
Use B2B for Oracle Integration in Standalone Mode

You can create and manage B2B standalone mode integrations in B2B for Oracle Integration.

Topics:

- Standalone Mode Concepts
- Use the B2B Action in Standalone Mode
- Override Validation and Functional Acknowledgment Settings in the Mapper

Standalone Mode Concepts

B2B for Oracle Integration provides several components that enable you to design integrations to handle B2B messages. This includes the AS2 Adapter and the B2B action for EDI translation. Use these components to create integrations to receive and send messages from (or to) your external trading partners. In this mode, the external trading partner is not defined explicitly as you do in B2B trading partner mode. Standalone mode ignores any trading partner definitions.

Standalone mode supports the following:

- Interfacing with external trading partners using any adapter provided in Oracle Integration: AS2, FTP, REST, SOAP, AQ, and more.
- Handling any data formats that Oracle Integration supports: EDI, CSV, XML, JSON, and more.

In contrast, B2B trading partner mode currently supports a limited set of adapters (AS2 and FTP) and only EDI data formats.

B2B Action

The B2B action was previously called EDI Translate. This action was renamed because it now provides additional operations besides EDI translation. However, those operations are only available in B2B trading partner mode. When using the B2B action in standalone mode, it provides the two key operations, inbound and outbound EDI translation, just as it did before. Any existing integration you created using the EDI Translate action works the same as before.

Support for the EDIFACT, X12, X12 HIPAA, OAG, custom, fixed length, and delimiter document standards is available in standalone mode.

Details about how to use the B2B action in your integrations are provided. See Design a B2B Standalone Mode Integration.

Adapters for Communication

You can design your B2B integrations with any adapter as a trigger or invoke connection for receiving or sending messages from (or to) a trading partner. This section provides pointers for using the popular communication protocols in conjunction with B2B processing below:

- AS2
  You can design an integration to receive or send EDI or another type of payload over the AS2 protocol.
- FTP
  You can design an integration to poll for files or send files with EDI or another type of payload. A stage file action is typically used with the FTP Adapter.
- REST, plain HTTP
  You can design an integration to receive an EDI or another type of payload over REST or plain HTTP using the REST Adapter. For EDI processing, you can configure the REST Adapter trigger to accept a binary payload and process it using the B2B action.
- SOAP
  You can design an integration to receive or send a SOAP message from (or to) a trading partner, typically with an XML payload to cover any of the XML-based B2B standards. In this pattern, you bring the XSD schemas and upload them into the SOAP Adapter trigger or invoke adapter configuration.

Use the B2B Action in Standalone Mode

You can translate a message to or from the Electronic Data Interchange (EDI) format in an integration with the B2B action.

The B2B action translates an incoming EDI document into an Oracle Integration XML message and an outgoing Oracle Integration XML message into an EDI X12 document.

> **Note:**
>
> This section provides an overview of how to add a B2B action to an integration. Complete integration design details are provided. See Create Application Integrations in *Using Integrations in Oracle Integration 3*.

1. In the navigation pane, click **Design**, then **Integrations**.
2. Click **Create**, then select **Application** or **Schedule**.
3. Design your integration. For example, add a trigger connection if you selected **Application**.
4. Add a B2B action to the integration in either of the following ways:
   - On the right side of the canvas, click **Actions** ⚡ and drag the **B2B** action to the appropriate location.

- Click + at the location where you want to add the for-each action, then select **B2B**.

The Configure B2B Action Wizard is displayed.

5. Enter a name and optional description.
6. Select how you want to use the B2B action, then click **Continue**.
   - **B2B Trading Partner mode**: Select to include B2B trading partner profiles and message persistence in your integration.
   - **Standalone mode**: Select to use the B2B action independently in your integration *without* a B2B trading partner profile or message persistence.
7. If you select **B2B Trading Partner mode**, follow these wizard steps:
   a. Select the B2B message direction for this B2B action:
      - **Inbound**: The B2B message is sent from the trading partner to the host trading partner (partner where Oracle Integration is installed).
      - **Outbound**: The B2B message is sent from the host trading partner (partner where Oracle Integration is installed) to the trading partner.

   b. Select the operation for the B2B action to perform based on the direction.

If You Selected...	Then Select an Operation...
Inbound	• **Fetch Message**: Fetches a B2B business message from the persistence store, given a message identifier. • **Translate**: Translates an inbound EDI message to an Oracle Integration message. • **Mark As Error**: Marks a B2B business message with a failed processing status.
Outbound	• **Translate**: Translates an Oracle Integration message to an outbound EDI message. • **Mark As Error**: Marks a B2B business message with a failed processing status. • **Fetch raw messages**: Fetches a B2B business message payload from persistence stored for processing.

   c. If you select either of the following, the Select Data Format page is displayed.
      - **Inbound** message direction and **Fetch Message** operation
      - **Outbound** message direction and **Translate** operation

      i. Select the document definition for the B2B action to handle, and click **Search** to refresh the page.

      ii. Select the document standard, version, and type.

      iii. Click **>**.

   d. Review your selections on the Summary page, then click **Done**.

8. If you select **Standalone mode**, follow these wizard steps:
   a. Specify the message translation and document format details.

Element	Description
Select the direction in which to translate the message	• **Inbound EDI message to Oracle Integration message**: When an integration receives an EDI document from a business partner, it is considered an inbound document (an EDI document is translated to XML). • **Oracle Integration message to outbound EDI message**: When an integration sends an EDI document to a business partner, it is considered an outbound document (an EDI document is generated from XML).
Document Standard	Select the document standard to follow when exchanging business documents between partners. • **Custom** • **Delimited** • **EDIFACT** • **FixedLength** • **OAG** • **X12** • **X12HIPAA**
Document Version	Select the version of the document standard to use.
Document Type	Select the document type (for example, purchase order, invoice, shipping notice, or others). The document types available for selection are based on the document version you selected.
Document Definition	Select the document definition that you created on the Documents page. For **EDIFACT**, **X12**, or **X12HIPAA**, you can also select **Standard**. See Create a Custom B2B Document Definition.
EDI Character Encoding	Select the character encoding that the inbound EDI document is expected to use.
Perform validations on input data	• **Yes**: Validates the structure and data of an inbound EDI message. Enabling message validation has an impact on performance. If errors are found, translation does not succeed. • **No**: Errors are ignored during translation and the message is passed through in its current format. You can override this setting at runtime with the **Validate** request element in the mapper. See Override Validation and Functional Acknowledgment Settings in the Mapper.

b. Click **Continue**.

c. Specify to optionally upload sample data to test that translation is successful.

Element	Description
Run a test translation with sample data (optional)	Select or drag and drop a sample file. To test inbound EDI message translation, upload an EDI document. To test outbound EDI message generation, upload an XML document.

Element	Description
Translate	Click to translate your sample data. Output is displayed in the **Output of translation** field. Any errors are displayed in the **Error in translation** field.

d. Click **Continue**.

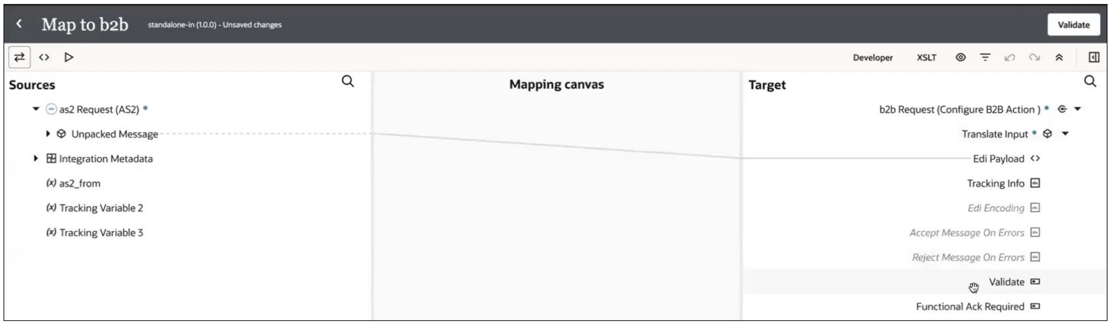
e. Review your selections on the Summary page, then click **Finish**.

Override Validation and Functional Acknowledgment Settings in the Mapper

You can override the design-time settings for validation and functional acknowledgment at runtime in the mapper that precedes the B2B action in the integration canvas.

Override the design-time settings at runtime with the following response elements:

- **Validate**: Set to **Yes** or **No** to override the design-time setting you made in the Configure B2B Action Wizard.
- **Functional Ack Required**: Set to **Yes** or **No** to override the document definition setting.



4

B2B Schemas and B2B Documents

You create a B2B schema using a standard schema definition as a starting point. You can also customize the schema according to your needs. You then create a document definition based on the schema. This document definition represents the message to use in your integrations. You then select this document to convert when configuring the B2B action in your integration.

Topics:

- Work with B2B Schemas
- Work with B2B Documents

Work with B2B Schemas

B2B for Oracle Integration allows you to create a new schema using a standard schema or by importing a Standard Exchange Format (SEF) file. In addition, you can perform a host of customizations on the schemas and generate implementation guides for custom or standard schemas.

Topics:

- Create a New B2B Schema
- Create a Schema from an Imported SEF File
- Customize or Edit a B2B Schema
- Generate an Implementation Guide

Create a New B2B Schema

You can create a new B2B schema based on a standard document type and customize it to match your requirements.

1. In the navigation pane, click **B2B**, then **Schemas**.
2. On the Schemas page, click **Create**.
3. Enter the following details.

Element	Description
Name	Enter a name.
Identifier	This field is automatically populated with the name. You can manually change this value.
Description	Enter an optional description of the customization details for this schema.

Element	Description
Document standard	Select the document standard. The document standard identifies the business protocol to follow when exchanging business documents between partners. • **EDIFACT** • **X12** • **X12HIPAA** • **OAG** • **Custom** • **RosettaNet** (version 2.0) • **FixedLength** • **Delimited**

4. If you selected **EDIFACT**, **X12**, or **X12HIPAA**, enter the following details:

Element	Description
Document version	Select the document version from the drop-down list. You cannot manually define a document version.
Document type	Select the document type from the drop-down list. You cannot manually define a document type.

5. If you selected **OAG**, **Custom**, or **RosettaNet**, enter the following details:

Element	Description

Schema location	Click **Browse** to select the schema to use.
	If you select **RosettaNet**, note that RosettaNet PIPs are not preseeded in B2B for Oracle Integration. You can download the RosettaNet schema to use from this site. Expand the **Specification** list to find the schema you want to download and use in B2B for Oracle Integration.
	Prepare the schema for upload into B2B for Oracle Integration:
	a. Download the schema from RosettaNet and place it in a new folder.
	b. Unzip the downloaded schema.
	c. Navigate to the `XML` directory inside the unzipped folder.
	d. Find and delete all non-XSD files in this directory.
	`$ find . -type f  | grep -v xsd | xargs rm`
	Note: If you are using a UI-based explorer, some hidden files/ folders may exist that also need to be removed. In which case, the above command must be modified. For example, the `__MACOX__` folder on macOS or the `Thumbs.db` file on Windows.
	e. Create a new `zip` file from current XML directory onwards.
	`zip -r pip4b2-v11-01-00-notifyofshipmentreceipt.zip ./`
	f. Upload this new `zip` file to define a new schema.
	After you upload a RosettaNet schema to use, the name of the PIP is displayed in the **Root element** field (for example, **Pip3A4PurchaseOrderRequest**.)

Document version	• For **Custom** or **RosettaNet**, manually enter a new value or select an existing value from the drop-down list. • For **OAG**, select an existing value from the drop-down list. You cannot manually define a document version.
Document type	• For **Custom** or **RosettaNet**, manually enter a new value or select an existing value from the drop-down list. • For **OAG**, select an existing value from the drop-down list. You cannot manually define a document type.

6. If you selected **FixedLength**, enter the following details:

Element	Description
Scan for character	Enter an integer between `1` and `32` to identify the number of characters to scan to identify the segments in the payload. For example, enter `3` to identify the schema segments from the first three characters in each row. See Positional Flat or Fixed Length Payloads Support.
Drag and Drop	Drag a sample payload file with a fixed-length content. UTF-8 is the supported character encoding. After creating the schema, you can add child elements, as necessary. See Add Child Elements to a Fixed-Length Schema.

Element	Description
Character Encoding	Select the character encoding to use.
Document version	Manually enter a new value or select an existing value from the drop-down list.
Document type	Manually enter a new value or select an existing value from the drop-down list.

7. If you selected **Delimited**, enter the following details:

Element	Description
Drag and Drop	Drag and drop the delimited payload file.
Character Encoding	Select the character encoding to use.
Segment Delimiter	Select the delimiter used to separate segments in the payload. See Delimited Payload Support.
Element Delimiter	Select the delimiter used to separate elements in the payload.
Document version	Manually enter a new value or select an existing value from the drop-down list.
Document type	Manually enter a new value or select an existing value from the drop-down list.

8. Click **Create**.
9. Customize the schema to match your business requirements.

   For example customizations, see Customize or Edit a B2B Schema.

   After completing the customization, you can select the custom schema while creating a new B2B document as described in Create a Custom B2B Document Definition.

Create a Schema from an Imported SEF File

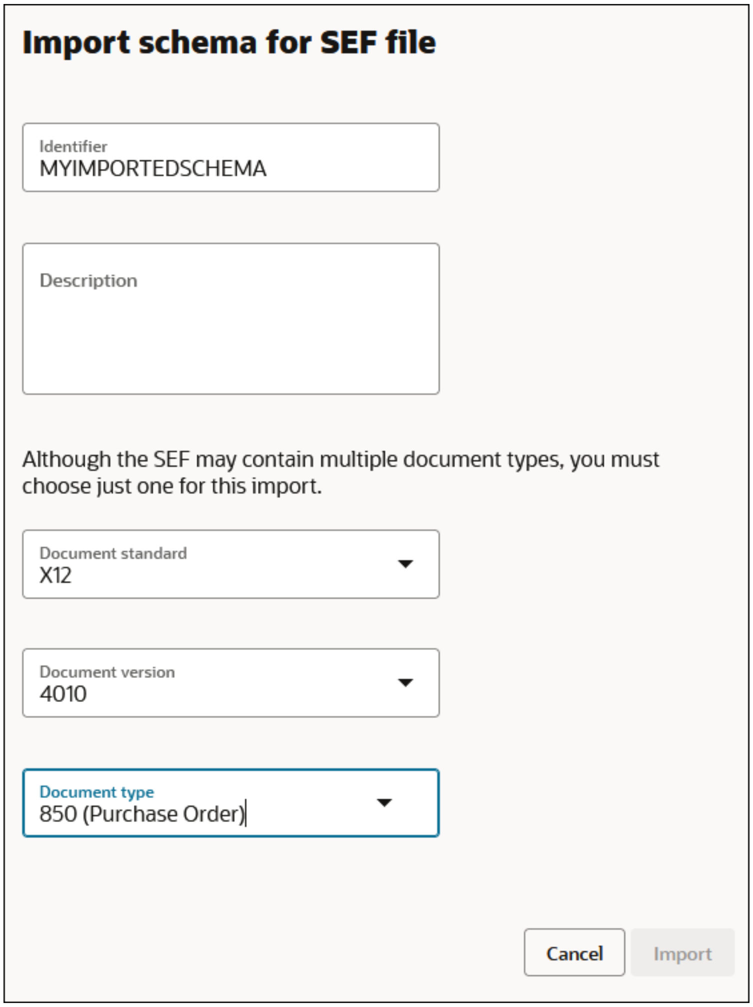
You can create a schema by importing a Standard Exchange Format (SEF) file. A SEF file exchanges EDI implementation guidelines in a machine-readable form.

1. In the navigation pane, click **B2B**, then **Schemas**.
2. Click **Import SEF**.

   The Import schema for SEF file dialog is displayed.
3. Enter the following details.

Element	Description
Select SEF File	Select the SEF file you want to import.
Name	Enter a name for the schema you want to create.
Identifier	This field is automatically populated with the document name. You can manually change this value.
Description	Enter an optional description of the customization details for this document.
Document Standard	Select the document standard. • **EDIFACT** • **X12** EDI X12 is automatically selected for use.

Element	Description
Document Version	Select the document version.
Document Type	Select the document type.



Import schema for SEF file

Identifier: MYIMPORTEDSCHEMA

Description

Although the SEF may contain multiple document types, you must choose just one for this import.

Document standard: X12

Document version: 4010

Document type: 850 (Purchase Order)

4. Click **Import**.

   An information message is displayed indicating that the SEF file has been imported successfully. The B2B schema is then created based on the SEF file.

Customize or Edit a B2B Schema

To create custom schemas, you can add new constructs (such as new segments and elements) to a standard B2B schema or edit existing constructs within it.

Topics:

- About B2B Schema Constructs
- Add New Segments and Other Schema Constructs
- Edit Properties of Segments and Other Schema Constructs

- Edit the Code List for an Element
- Edit Syntax Rules for Segments and Composites
- Add Child Elements to a Fixed-Length Schema

About B2B Schema Constructs

A B2B schema represents the message format of an EDI X12, EDIFACT, or X12 HIPAA document, which is a hierarchical structure based on four types of constructs:

- **Element:** The smallest unit that represents a single data field of a primitive type, such as, alphanumeric text, integer, decimal, date, time, or binary.
- **Composite:** A complex data type consisting of one or more elements.
- **Segment:** The next higher level construct, consisting of a sequence of elements and composites.
- **Loop or Loop Segment:** A container for a specific set of segments or child loops, which make its structure nested and hierarchical.

The schema editor in B2B for Oracle Integration displays each of these constructs in its own row. The rows for segments, composites, and loops are expandable, and display the elements they contain. The order of the rows in the schema defines the exact order in which each of the constructs must appear in an actual EDI X12, EDIFACT, or X12 HIPAA document.

- An EDI X12 document, which is the topmost construct, is defined as an ordered set of segments and loops. The schema for all EDI X12 documents starts with a segment named `ST` and ends with the segment `SE` across all documents types. These segments are called transaction set envelope segments, and they cannot be deleted in the B2B schema editor.
- An EDIFACT document, which is the topmost construct, is also defined as an ordered set of segments and loops. The schema for all EDIFACT documents starts with a segment named `UNH` and ends with the segment `UNT` across all documents types. These segments are called transaction set envelope segments, and they cannot be deleted in the B2B schema editor.
- An X12 HIPAA document, which is the topmost construct, is also defined as an ordered set of segments and loops. The schema for all X12 HIPAA documents starts with a segment named `UNH` and ends with the segment `UNT` across all documents types. These segments are called transaction set envelope segments, and they cannot be deleted in the B2B schema editor.

Add New Segments and Other Schema Constructs

According to your organization's requirements, you can add new constructs to a standard B2B schema to customize it.

To add new constructs to a schema:

1. In the navigation pane, click **B2B**, then **Schemas**.
2. Hover over a schema and click **Edit** ✎.
3. Hover over the row of a required hierarchy and click **Actions** •••.
4. Select an option to add a new construct. The Details panel appears on the right, displaying the **Properties** tab, where you can enter the necessary information.

> **Note:**
>
> The options displayed in the **Actions** ••• list vary according to the hierarchy of the rows. The **Actions** ••• list for a segment or loop row has options to create new child elements, child composites, segments and loops, whereas the **Actions** ••• list for a composite or element row has options to create only new composites and elements.

Edit Properties of Segments and Other Schema Constructs

B2B for Oracle Integration allows you to modify properties of existing constructs of a schema that's derived from a standard schema.

To edit the properties of constructs:

1. In the navigation pane, click **B2B**, then **Schemas**.
2. Hover over a schema and click **Edit** ✎.
3. In the schema editor, hover over a row you want to modify and click **Edit details** ✎.
   A panel open on the right, displaying the **Properties** tab. The property fields displayed in the tab vary for element, composite, segment, and loop rows. You can edit the field values to customize and deviate from the X12 standard according to your business requirements. The following sections describe the different property fields that are displayed for each construct type.

Element Properties

Property	Description
Name	A descriptive name for the element.
Element ID	The ID that identifies this element in the EDI X12, X12 HIPAA, or EDIFACT element dictionary. This field is not editable.
Position	The element's position number within the parent segment or composite. This field is not editable.
Purpose	A detailed description of the element's purpose.
Requirement	• **Mandatory:** The element must have a nonempty value in an EDI document. • **Optional:** The element may have a value or be empty. • **Conditional:** The element may have a value or not depending on syntax rules defined on the parent segment or composite. This setting is preselected by the X12, X12 HIPAA, or EDIFACT standard and typically not modified. To override this setting, modify the **Usage** field.

Property	Description
Usage	This field overrides the **Requirement** field. When this field is left blank (no value selected), it implies that the element is used and the selection made in the **Requirement** field is enforced. • **Must Use:** Overrides the selection in the **Requirement** field and treats the element as mandatory. • **Not Used:** Overrides the selection in the **Requirement** field and specifies that the element should remain empty (that is, not have a value). • **Recommended:** Follows the selection in the **Requirement** field, but suggests that the element is a preferred element. • **Not Recommended:** Follows the selection in the **Requirement** field, but suggests that the element is not a preferred element.
Type	The element's type as defined by the X12 standard in the EDI X12 element dictionary, X12 HIPAA standard, or EDIFACT standard in the EDIFACT element dictionary. The type is one of the following: ID (Identifier), AN (Alphanumeric Text String), DT (date), TM (time), N(0-9) (Integer with implied decimal), R (Decimal), or B (Binary). This field is not editable.
Length (Min / Max)	The minimum and maximum characters (or numeric digits) that the element's value allows.
Repeat	Specifies if it's a repeating element. • `1`: Indicates that the element has a single value (non-repeating). • A number greater than one or the `>1` symbol: Indicates that this is a repeating element; that is, one element accommodates multiple values. The EDI X12 version 4012 and older versions do not support repeating elements; therefore, this setting is ignored for those versions.

Composite Properties

A composite has property settings similar to an element construct, except for the **Type** and **Length** fields.

Segment Properties

A segment has several property settings similar to an element construct. However, it doesn't have the **Element ID**, **Type**, and **Length** fields. In addition, there are the following differences.

Property	Description
Repeat	For a segment, this property indicates how many consecutive instances of the segment may occur in an EDI document. Segment repetition is supported in all X12 versions, unlike elements.
Table Area	Indicates the section of the EDI document to which this segment belongs. Typically, an EDI document is defined with a heading, summary, and detail sections, also called table areas.

Loop Properties

A loop has four property settings, namely **Requirement**, **Usage**, **Repeat**, and **Table Area**, which have similar meanings as those of a segment.

Edit the Code List for an Element

In addition to properties, you can also edit code lists for element constructs while customizing standard B2B schemas.

Some EDI X12 elements are restricted to a set of enumerated values. A code list displays the allowed values for an element. Typically, elements of the type ID (identifier) have a code list associated with them; however, any element can have a code list.

To edit an element's code list,

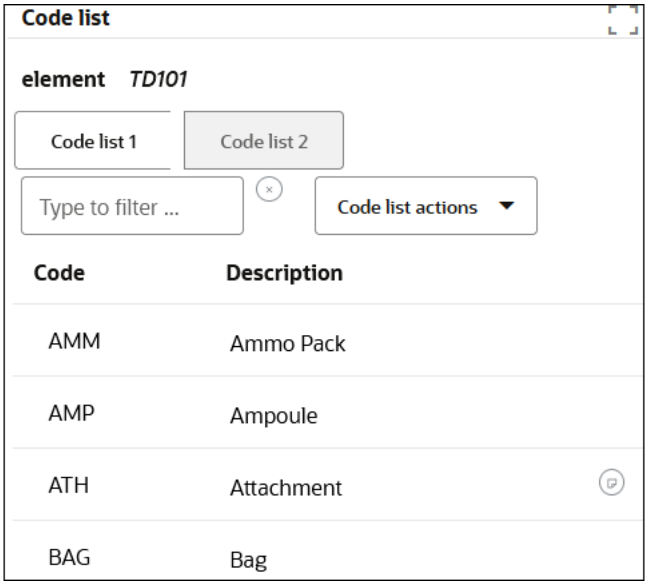
1. Hover over the element row and click **Edit Details** ✎.
   The Details panel appears on the right, displaying the **Properties** tab.
2. Click **Code list** ‹›.
   If there is a code list defined, all code values are displayed in the table.
3. Hover over a row and use the buttons that appear to edit or delete the row or to add a new row.
   While editing, you can add a description and a note to each code. While deleting, a row is first marked for deletion (which can be reversed with an **Undo**) and is actually deleted upon clicking **Save**.

   > **Note:**
   >
   > While adding a new row, you can insert it at any position. The validation of an element's data against the code list happens regardless of the order. After a **Save** action, the code list is sorted alphabetically.

4. If you want to edit several rows in a code list, use the **Export CSV** action under **Code list actions** to export the list to a comma-separated values file for editing in an external tool, such as a spreadsheet editor.
   The CSV file has a header row identical to the code list table. After the editing is complete, you can import it back using the **Import CSV** action. This overwrites the existing code list completely with the codes defined in the CSV file.

Multiple Code Lists for an Element

Occasionally, you may find elements with multiple code lists. One such example is the element TD101 (Packaging Code) for the document type EDI X12 850 (any version). This element has two code lists defined as follows:



Code list

element *TD101*

Code	Description
AMM	Ammo Pack
AMP	Ampoule
ATH	Attachment
BAG	Bag

The element's data value is a concatenation of one code from each of the code lists. For example, a value of `AMM01` is a valid data value because it uses `AMM` from **Code list 1** and `01` from **Code list 2**.

Customizing a Code List

An element is considered customized (with respect to code lists) if any of its associated code lists have codes that are different from the standard X12 codes. However, only changing the description or adding notes is not considered a customization because it does not affect the element's allowed data values or the validation.

Edit Syntax Rules for Segments and Composites

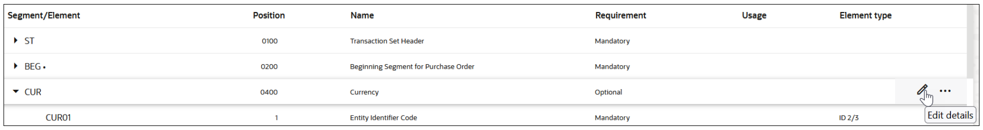
You can view and edit the syntax rules, if defined for a segment or a composite. These rules apply to data elements and are enforced during the validation of the input data.

> **Note:**
>
> You cannot edit a schema currently in use in an active document definition.

1. In the navigation pane, click **B2B**, then **Schemas**.

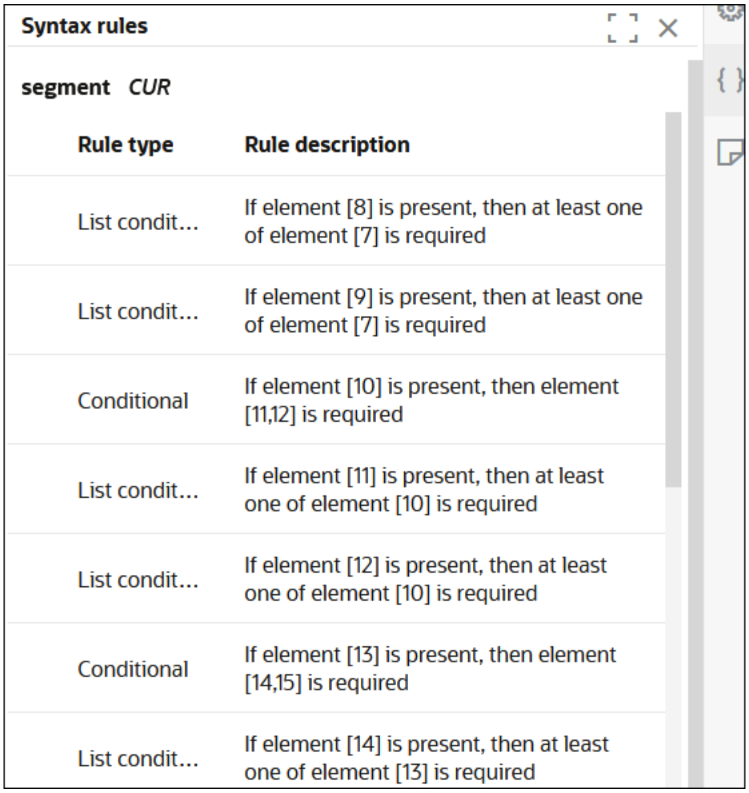
   The B2B schemas page shows all custom schemas you have created by cloning the standard schemas.

2. Open a custom schema, find the segment you want to view (for example, **CUR**), and select **Edit details** ✎.



Segment/Element	Position	Name	Requirement	Usage	Element type
▸ ST	0100	Transaction Set Header	Mandatory		
▸ BEG •	0200	Beginning Segment for Purchase Order	Mandatory		
▾ CUR	0400	Currency	Optional		
CUR01	1	Entity Identifier Code	Mandatory		ID 2/3

3. Click { } in the Details panel to open the syntax rules enforced for the selected segment.



Syntax rules

segment ***CUR***

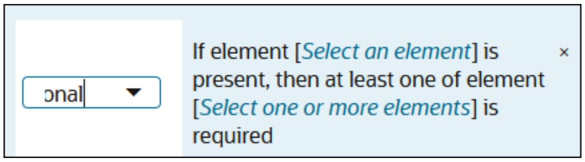
Rule type	Rule description
List condit...	If element [8] is present, then at least one of element [7] is required
List condit...	If element [9] is present, then at least one of element [7] is required
Conditional	If element [10] is present, then element [11,12] is required
List condit...	If element [11] is present, then at least one of element [10] is required
List condit...	If element [12] is present, then at least one of element [10] is required
Conditional	If element [13] is present, then element [14,15] is required
List condit...	If element [14] is present, then at least one of element [13] is required

4. Hover over a syntax rule row to see options to edit, add, or delete a rule.
5. If you select **Add +**, a **Select** list is displayed with the following syntax rule types:

X12 Syntax Rule	Description
Pairing	If any of the elements are present, all are required.
Required	At least one of the elements is required.
Exclusion	Only one of the elements may be present.
Conditional	If one element is present, other elements are required.
List Conditional	If one element is present, at least one of the other elements is required.

   For each rule, you are presented with either one or two element selections into which you select elements.

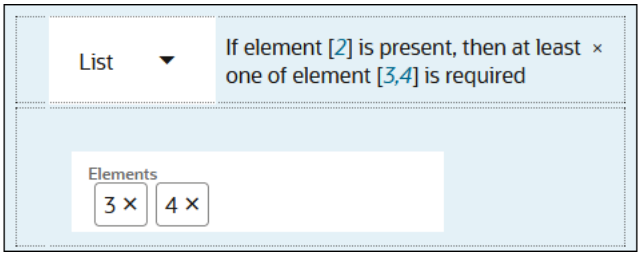
6. For example, if you add the **List Conditional** rule, click **select an element** for the first element selection and click **select one or more elements** for the second element selection.



If element [*Select an element*] is present, then at least one of element [*Select one or more elements*] is required

7. Select appropriate elements.

   The current selections are displayed in terms of the element position numbers.



If element [*2*] is present, then at least one of element [*3,4*] is required

   The element position numbers identify the corresponding child elements inside the selected segment or composite. For example, **2** means **CUR02** below and **3** and **4** mean **CUR03** and **CUR04**.

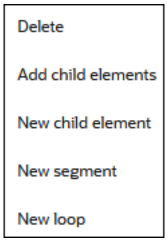
8. Click the element numbers in the rule description to add or remove elements.

> **Note:**
>
> Deleting a syntax rule first marks it for deletion (which can be undone) and then deletes it when the B2B schema is saved.

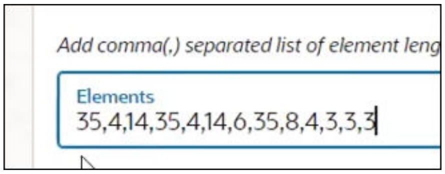
Add Child Elements to a Fixed-Length Schema

When editing a fixed-length schema, an additional option is provided for adding multiple child elements to a segment, each of a specified fixed length.

1. Hover over the fixed-length schema and click **Edit** ✎.
2. Hover over the row of a segment and click **Actions** •••.
3. Select **Add child elements**.



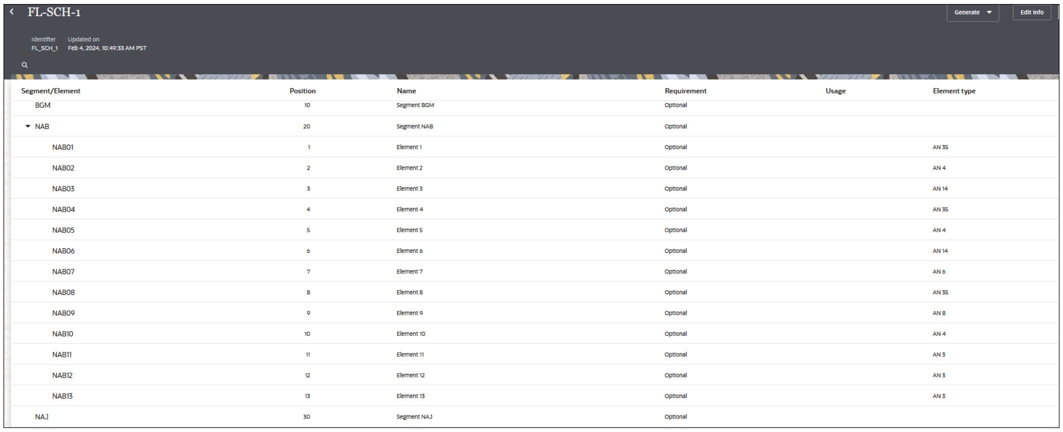
4. Enter a comma-separated list of element lengths. For example:



Add comma(,) separated list of element leng

Elements
35,4,14,35,4,14,6,35,8,4,3,3,3

This action adds alphanumeric child elements beneath the selected segment. The **Element type** column shows the length you entered for each child element.



FL-SCH-1

Identifier: FL_SCH_1
Updated on: Feb 4, 2024, 10:49:33 AM PST

Segment/Element	Position	Name	Requirement	Usage	Element type
BGM	10	Segment BGM	Optional		
NAB	20	Segment NAB	Optional		
NAB01	1	Element 1	Optional		AN 35
NAB02	2	Element 2	Optional		AN 4
NAB03	3	Element 3	Optional		AN 14
NAB04	4	Element 4	Optional		AN 35
NAB05	5	Element 5	Optional		AN 4
NAB06	6	Element 6	Optional		AN 14
NAB07	7	Element 7	Optional		AN 6
NAB08	8	Element 8	Optional		AN 35
NAB09	9	Element 9	Optional		AN 8
NAB10	10	Element 10	Optional		AN 4
NAB11	11	Element 11	Optional		AN 3
NAB12	12	Element 12	Optional		AN 3
NAB13	13	Element 13	Optional		AN 3
NAJ	30	Segment NAJ	Optional		

Generate an Implementation Guide

You can generate implementation guides for standard or customized schemas to share with your trading partners.

Topics:

- About the Implementation Guide
- Generate the Implementation Guide for a Schema

About the Implementation Guide

An implementation guide is a single, consolidated document with full details about a B2B schema.

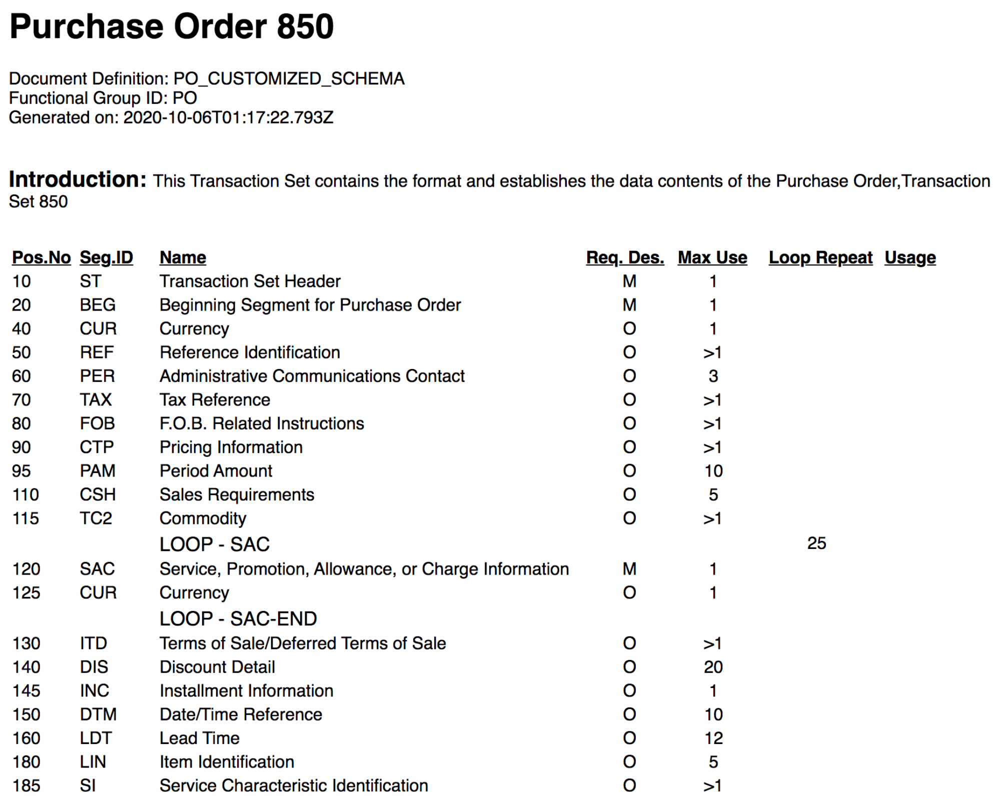
Typically, a host company shares the implementation guide for an EDI schema with their trading partners. This guide is shared especially when a schema has been customized to deviate from a standard X12 or X12 HIPAA document. It points out some of the customizations made to the standard schema (not all customizations are highlighted in the document).

In B2B for Oracle Integration, you can generate implementation guides for both standard and custom B2B schemas. The guide is generated as an HTML document, which you can edit to apply branding and print as a PDF file if necessary.

Note:

You cannot currently generate an implementation guide for EDIFACT documents.

The following image shows an example X12 implementation guide:



Purchase Order 850

Document Definition: PO_CUSTOMIZED_SCHEMA
Functional Group ID: PO
Generated on: 2020-10-06T01:17:22.793Z

Introduction: This Transaction Set contains the format and establishes the data contents of the Purchase Order,Transaction Set 850

Pos.No	Seg.ID	Name	Req. Des.	Max Use	Loop Repeat	Usage
10	ST	Transaction Set Header	M	1		
20	BEG	Beginning Segment for Purchase Order	M	1		
40	CUR	Currency	O	1		
50	REF	Reference Identification	O	>1		
60	PER	Administrative Communications Contact	O	3		
70	TAX	Tax Reference	O	>1		
80	FOB	F.O.B. Related Instructions	O	>1		
90	CTP	Pricing Information	O	>1		
95	PAM	Period Amount	O	10		
110	CSH	Sales Requirements	O	5		
115	TC2	Commodity	O	>1		
		LOOP - SAC			25	
120	SAC	Service, Promotion, Allowance, or Charge Information	M	1		
125	CUR	Currency	O	1		
		LOOP - SAC-END				
130	ITD	Terms of Sale/Deferred Terms of Sale	O	>1		
140	DIS	Discount Detail	O	20		
145	INC	Installment Information	O	1		
150	DTM	Date/Time Reference	O	10		
160	LDT	Lead Time	O	12		
180	LIN	Item Identification	O	5		
185	SI	Service Characteristic Identification	O	>1		

Structure of an Implementation Guide

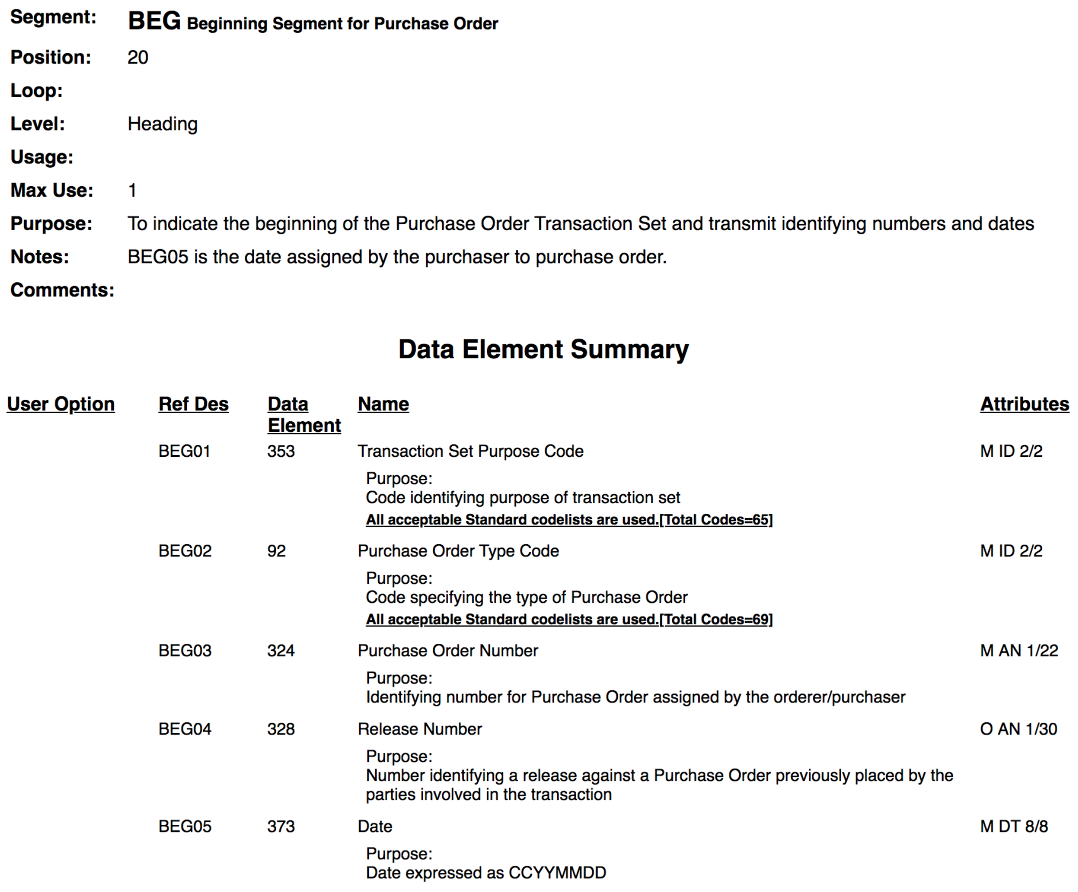
List of Segments and Loops

The guide begins with a listing of each segment and loop defined in the document schema. Each row defines a segment or loop with the following columns:

- Pos.No: The position number of the segment as defined in the X12 standard.
- Seg.ID: The segment identifier.
- Name: A descriptive name for the segment.
- Req.Des: The requirement designator (M = Mandatory, O = Optional, C = Conditional).
- Max Use: The Repeat property of the segment.
- Loop Repeat: The Repeat property of the loop, when the row represents a loop.
- Usage: The Usage property of the segment (Must Use, Not Used, Recommended, Not Recommended).

Details Section for Each Segment

Each segment has a details section that lists elements in the segment as shown in the following image:



Segment: **BEG** **Beginning Segment for Purchase Order**

Position: 20

Loop:

Level: Heading

Usage:

Max Use: 1

Purpose: To indicate the beginning of the Purchase Order Transaction Set and transmit identifying numbers and dates

Notes: BEG05 is the date assigned by the purchaser to purchase order.

Comments:

Data Element Summary

User Option	Ref Des	Data Element	Name	Attributes
	BEG01	353	Transaction Set Purpose Code Purpose: Code identifying purpose of transaction set All acceptable Standard codelists are used.[Total Codes=65]	M ID 2/2
	BEG02	92	Purchase Order Type Code Purpose: Code specifying the type of Purchase Order All acceptable Standard codelists are used.[Total Codes=69]	M ID 2/2
	BEG03	324	Purchase Order Number Purpose: Identifying number for Purchase Order assigned by the orderer/purchaser	M AN 1/22
	BEG04	328	Release Number Purpose: Number identifying a release against a Purchase Order previously placed by the parties involved in the transaction	O AN 1/30
	BEG05	373	Date Purpose: Date expressed as CCYYMMDD	M DT 8/8

The Data Element Summary table lists the elements declared in the segment and has the following columns:

- User Option: The Usage property of the element (Must Use, Not Used, Recommended, Not Recommended).
- Ref Des: The reference designator of the element, which is a convention to represent an element. For example, BEG01 is the first element in the segment BEG. If an element is a composite element, the child elements in the composite use the convention REF04-01, where 01 is the sub-element position, 04 is the composite element's position, and REF is the segment.
- Data Element: The identifier of the element's definition.
- Name: A descriptive name assigned to the element.
- Attributes: Includes more properties of the element's definition. A value like `M ID 2/2` is made of the following parts:
  - The requirement designator for the element (M = Mandatory, O = Optional, C = Conditional).
  - Element's data type (ID=Identifier, AN=Alphanumeric String, N0=integer number, etc.).
  - Element's minimum and maximum character length. A value `1/30` implies that the element's value must be between 1 and 30 characters to be valid.

Generate the Implementation Guide for a Schema

View or download the implementation guide for a schema in the HTML format. The implementation guide is supported only for X12 and X12 HIPAA.

1. In the navigation pane, click **B2B**, then **Schemas**.
2. **For a standard schema:** Click **Generate**, then click **Implementation Guide**.
3. **For a custom schema:** Hover over a schema row.
4. Click **Actions** ⋯, then select **Generate implementation guide**.
5. On the Generate Implementation Guide page, click **Download** or **View** to generate the document in HTML format.

Work with B2B Documents

After you create a schema, you must create a document based on that schema for use in your integration. The document represents a specific snapshot of a schema to use in an integration.

Topics:

- Create a Custom B2B Document Definition
- Edit or Clone a B2B Document

Create a Custom B2B Document Definition

You can create customized document definitions to use in your B2B integrations. Custom document definitions are useful for scenarios in which your trading partner requires specific customizations to meet certain business requirements.

> **Note:**
>
> You must first create an **OAG**, **Custom**, **RosettaNet**, **FixedLength**, or **Delimited** schema before you can create a document definition that uses one of these standards. See Create a New B2B Schema.

1. In the navigation pane, click **B2B**, then **Documents**.
2. On the Documents page, click **Create**.
   The Create B2B document panel is displayed.
3. Enter the following details to create a new B2B document definition.

Element	Description
Name	Enter a document name.
Identifier	This field is automatically populated with the document name. You can manually change this value.
Description	Enter an optional description of the customization details for this document.

Element	Description
Document standard	Select a document standard: • **EDIFACT** • **X12** • **X12HIPAA** • **OAG** • **Custom** • **RosettaNet** (version 2.0) • **FixedLength** • **Delimited**

4. If you selected **EDIFACT**, **X12**, or **X12HIPAA**, enter the following details:

Element	Description
Document version	Specify the document version. The versions shown are based on the document standard you selected.
Document type	Specify the document type. The types shown are based on the document version you selected.
Document schema	Select an existing schema or select **Standard**. If you select **Standard**, you can customize the document schema after creation.

5. If you selected **OAG**, **Custom**, or **RosettaNet**, enter the following details:

Element	Description
Document version	Select an existing document version.
Document type	Select an existing document type.
Document schema	Select an existing schema. If a schema does not exist, you cannot create a document definition. See Create a New B2B Schema.

6. If you selected **FixedLength**, enter the following details:

Element	Description
Document version	Manually enter a new value or select an existing value from the drop-down list.
Document type	Manually enter a new value or select an existing value from the drop-down list.
Document schema	Select an existing fixed-length schema. If a fixed-length schema does not exist, you cannot create a document definition. See Create a New B2B Schema.

7. If you selected **Delimited**, enter the following details:

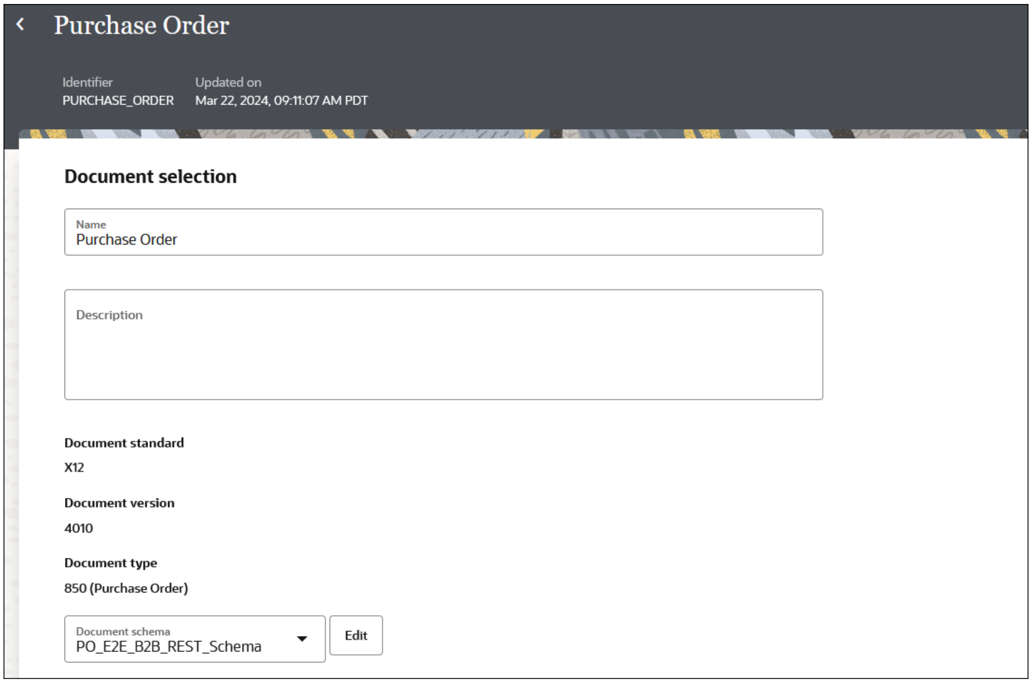
Element	Description
Document version	Manually enter a new value or select an existing value from the drop-down list.
Document type	Manually enter a new value or select an existing value from the drop-down list.

Element	Description
Document schema	Select an existing delimited schema. If a delimited schema does not exist, you cannot create a document definition. See Create a New B2B Schema.

8. Click **Create**.
   The details page for your new B2B document is displayed.
   - If You Selected EDIFACT, X12, or X12HIPAA
   - If You Selected RosettaNet
   - If You Selected OAG
   - If You Selected Custom
   - If You Selected FixedLength
   - If You Selected Delimited
   - Customize an EDIFACT, X12, or X12HIPAA Document Definition

If You Selected EDIFACT, X12, or X12HIPAA

The schema document details page appears (for this example, the **X12** document standard was selected).



Purchase Order

Identifier: PURCHASE_ORDER
Updated on: Mar 22, 2024, 09:11:07 AM PDT

Document selection

Name: Purchase Order

Description

Document standard: X12

Document version: 4010

Document type: 850 (Purchase Order)

Document schema: PO_E2E_B2B_REST_Schema

1. If you selected a previously-created schema as the document schema, the **Document schema** field shows the **Edit** button.
   a. If you want to edit the schema, click **Edit** to access the schema details page.
   b. If you want to change the schema, select a different schema from the drop-down list. If you had previously created custom schemas, they are also displayed for selection in the drop-down list. If you want, you can select those schemas to create further customizations.

c. If you want to customize the standard schema to satisfy your business requirements, click **Customize**. See Customize an EDIFACT, X12, or X12HIPAA Document Definition.

You can also define document properties for the schema.

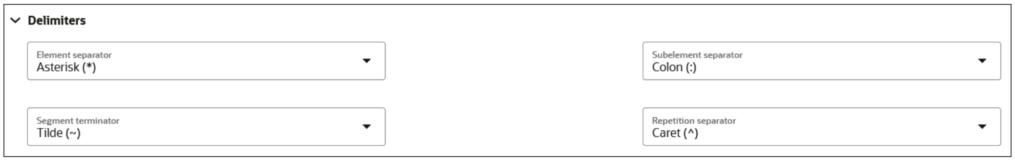
a. Scroll down to the **Document property** section.

b. In the **Business identifiers** section, add at least one identifier (primary). You can also add a second and third identifier to associate with this document. The business identifier names and expressions enable filtering by the identifiers on the Instances page when this document is sent or received.

c. In the **Expression** field, click to select a business identifier expression path.

d. In the **Delimiters** section, select the following delimiters to use for outbound X12 and EDIFACT documents:

   - Element
   - Subelement
   - Segment
   - Repetition (only for X12 versions 4020 or later)



Delimiters

Element separator: Asterisk (*)

Subelement separator: Colon (:)

Segment terminator: Tilde (~)

Repetition separator: Caret (^)

Note the following guidelines:

   - If the delimiter character you want to use is an ASCII printable character, assign a string value such as `|` to the corresponding attribute.
   - If you want to use a special character as a delimiter, specify it in hexadecimal format. For example, use `0xA`, where `0x` is a fixed prefix and `A` is the hexadecimal representation of the ASCII line feed character.
   - If your EDI payload uses a different encoding such as UTF-8, use the hexadecimal format to specify a Unicode character as a delimiter. For example, specify `0x03A60` to use the Unicode character `Φ` as a delimiter.
   - Prepare the receiving trading partner to receive a UTF-8 EDI payload for this process to work correctly. The delimiter string value you specify must equate to only a single character. Multiple characters are not allowed. The only exception is the `segment-terminator`, which allows one character plus an optional ASCII carriage return and line feed character at the end. For example, a `segment-terminator` value of `~0xd0xa` uses `~` followed by `<CR>` and `<LF>` as a `segment-terminator`. This value means each EDI segment is output on a separate line, making it more readable.

You can also define these delimiters for standalone mode outbound X12 and EDIFACT documents in the `TranslateOutput` XML.

   - `headers > interchange-ctrl > attribute 'element-separator'`
   - `headers > interchange-ctrl > attribute 'subelement-separator'`
   - `headers > interchange-ctrl > attribute 'segment-terminator'`

- `headers > interchange-ctrl > attribute 'repetition-separator'` (Only applies to X12 versions 4020 or later.)

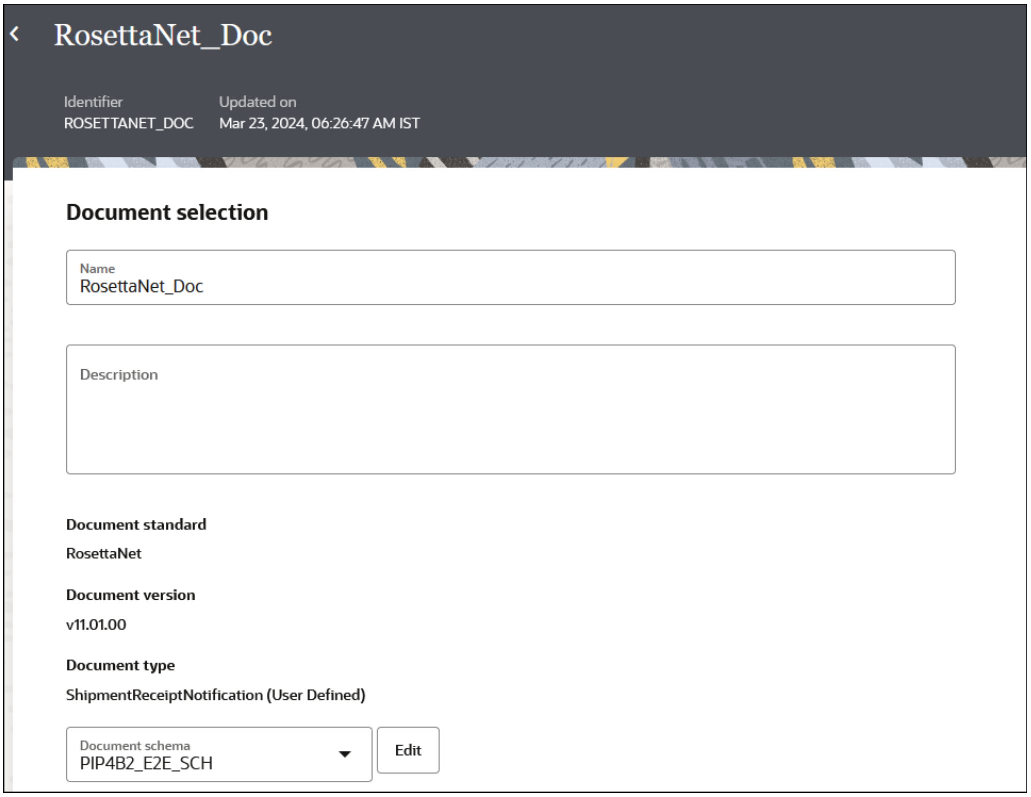
See Schema Elements for Outbound EDI.

e. In the **Correlation** section, select **Correlation from** or **Correlation to** from the drop-down list, enter an XPath name, and click the field to select an XPath expression. These values enable you to correlate related documents in business transactions. You can only select one option: **Correlation from** or **Correlation to**.
   - The **Correlation from** XPath retrieves the value from the payload to initiate the correlation.
   - The **Correlation to** XPath retrieves the value from the payload for correlation.

f. Click **Save**.

If You Selected RosettaNet

The schema document details page appears.



RosettaNet_Doc

Identifier: ROSETTANET_DOC
Updated on: Mar 23, 2024, 06:26:47 AM IST

Document selection

Name: RosettaNet_Doc

Description

Document standard: RosettaNet

Document version: v11.01.00

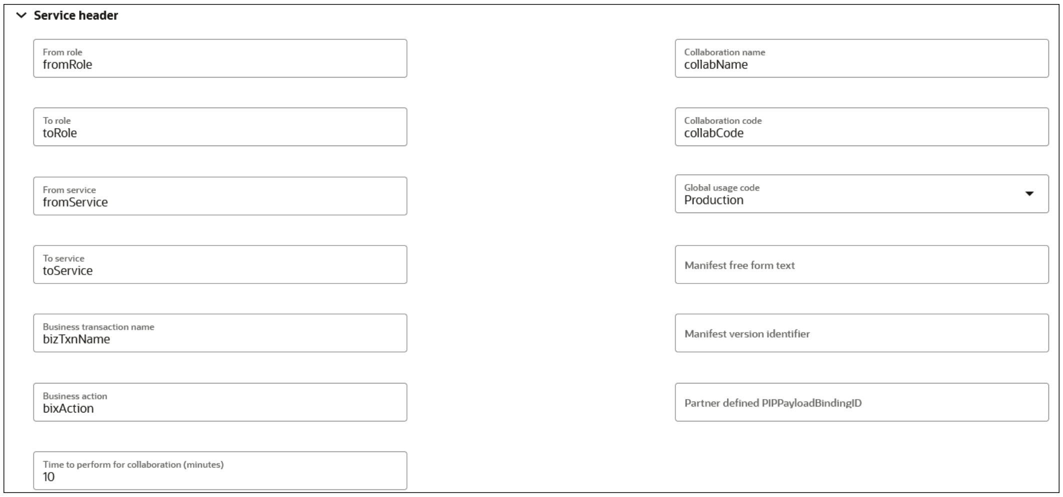
Document type: ShipmentReceiptNotification (User Defined)

Document schema: PIP4B2_E2E_SCH

1. If you want to use a different document schema, select an existing schema from the drop-down list or click **Edit** to upload a new schema file.
   You can also define document properties for the schema.
2. Scroll down to the **Document property** section.
3. In the **Business identifiers** section, add at least one identifier (primary). You can also add a second and third identifier to associate with this document. The

business identifier names and expressions enable filtering by the identifiers on the Instances page when this document is sent or received.

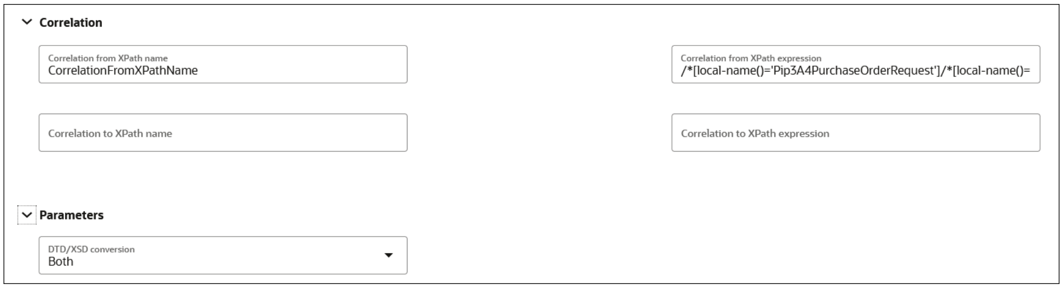
4. In the **Expression** field, click to select a business identifier expression path.
5. Expand **Service header** to view parameters used for validating inbound messages and creating RosettaNet outbound messages. For this example, the service header parameters provided are for a document configured to use PIP3A4 (Purchase Order Request). The **time to perform for collaboration (in minutes)** field shows the time window in which the buyer is expecting the purchase order acceptance message from the seller. If the acceptance message does not arrive within this time window, it times out.



Service header

From role: fromRole
To role: toRole
From service: fromService
To service: toService
Business transaction name: bizTxnName
Business action: bixAction
Time to perform for collaboration (minutes): 10
Collaboration name: collabName
Collaboration code: collabCode
Global usage code: Production
Manifest free form text
Manifest version identifier
Partner defined PIPPayloadBindingID

6. Expand **Correlation** and **Parameters** to show additional parameters. You must manually populate these fields. In particular, note the three fields for XPath expressions.



Correlation

Correlation from XPath name: CorrelationFromXPathName
Correlation from XPath expression: /*[local-name()='Pip3A4PurchaseOrderRequest']/*[local-name()=
Correlation to XPath name
Correlation to XPath expression

Parameters

DTD/XSD conversion: Both

The **Correlation from XPath expression** field shows the following expression:

```
/*[local-name()='Pip3A4PurchaseOrderRequest']/*[local-
name()='thisDocumentIdentifier']/*[local-
name()='ProprietaryDocumentIdentifier']/text()
```

The value for `ProprietaryDocumentIdentifier` in the request purchase order from the buyer integration must match with the value in the response purchase order of the seller integration. Whichever fields you configure for correlations to occur, you must supply the value correctly in the request/response payloads to get the relevant messages successfully correlated.

For example:

- If the seller's `ProprietaryDocumentIdentifier` parameter in the request purchase order consists of this value:

  ```
  <thisDocumentIdentifier>
     <ProprietaryDocumentIdentifier>420002933046</
  ProprietaryDocumentIdentifier>
  </thisDocumentIdentifier>
  ```

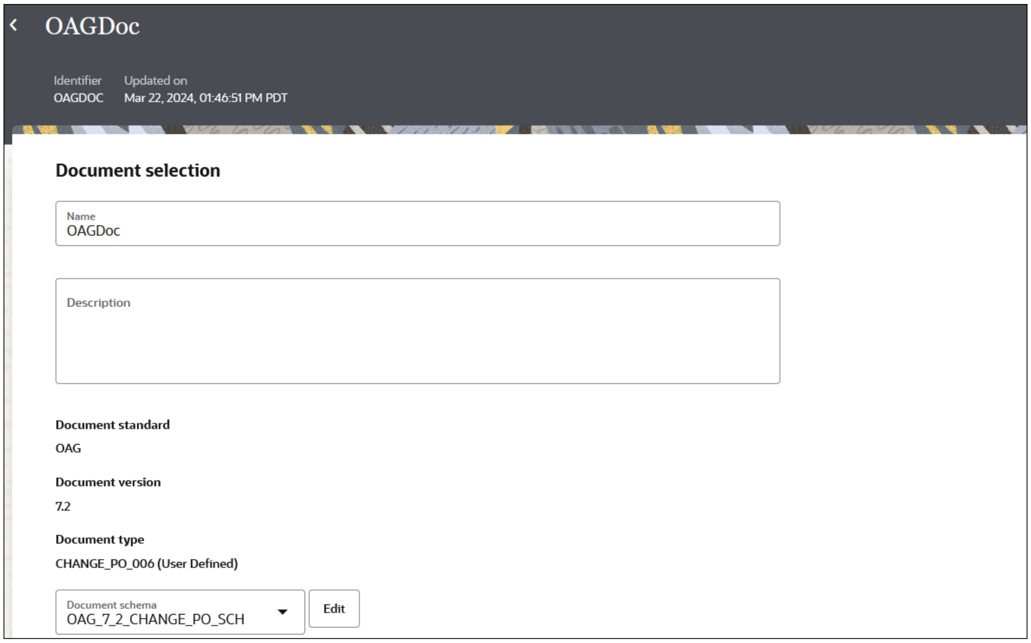
- Then the seller's the `ProprietaryDocumentIdentifier` parameter in the response purchase order must consist of the same value:

  ```
  <thisDocumentIdentifier>
     <ProprietaryDocumentIdentifier>420002933046</
  ProprietaryDocumentIdentifier>
  </thisDocumentIdentifier>
  ```

If your buyer and seller integrations are both REST Adapter-triggered, you can make these edits in the **Body** section of the Configure and run page. This page appears when you select **Run** from the **Actions** ••• menu for your integrations. This configuration ensures that message delivery occurs in the correct sequence at runtime.

If You Selected OAG

The schema document details page appears.



OAGDoc

Identifier: OAGDOC
Updated on: Mar 22, 2024, 01:46:51 PM PDT

Document selection

Name: OAGDoc

Description:

Document standard: OAG

Document version: 7.2

Document type: CHANGE_PO_006 (User Defined)

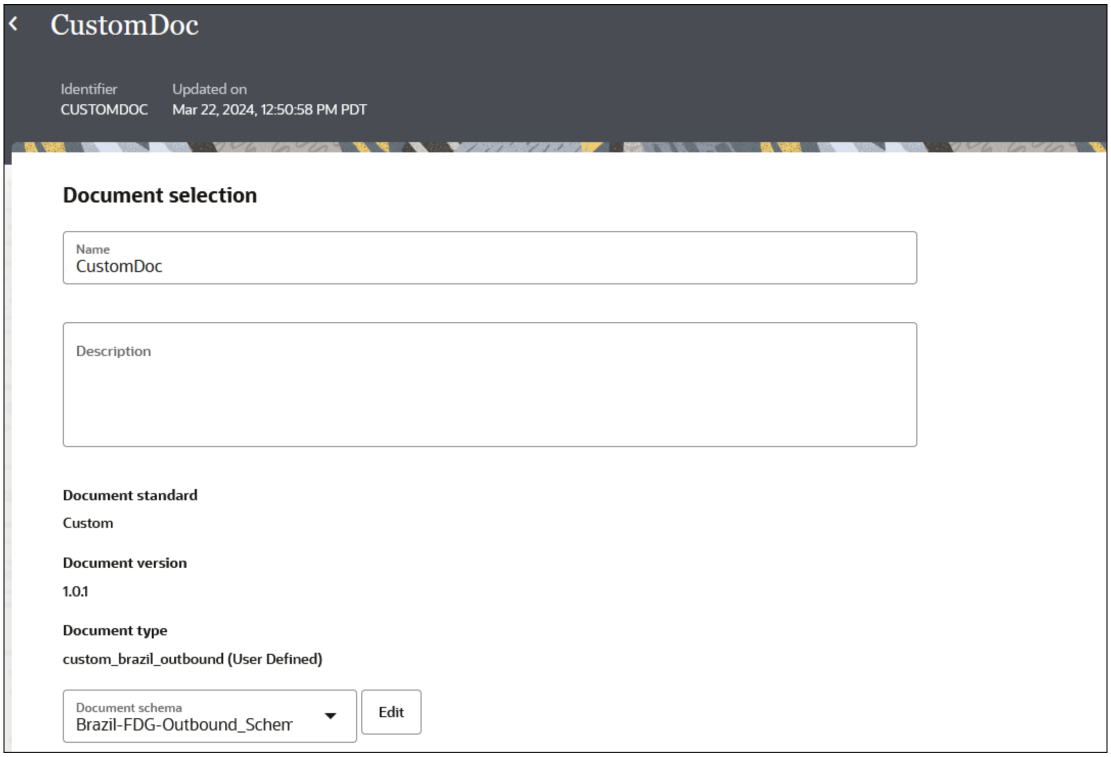
Document schema: OAG_7_2_CHANGE_PO_SCH

1. If you want to use a different document schema, select an existing schema from the drop-down list or click **Edit** to upload a new schema file.
   You can also define document properties for the schema.
2. Scroll down to the **Document property** section.

3. In the **Business identifiers** section, add at least one identifier (primary). You can also add a second and third identifier to associate with this document. The business identifier names and expressions enable filtering by the identifiers on the Instances page when this document is sent or received.
4. In the **Expression** field, click to select a business identifier expression path.
5. From the **DTD/XSD conversion** drop-down list, select a value.
   - **None**
   - **Outbound**
   - **Inbound**
   - **Both**
6. Optionally enter a logical ID and auth ID.
7. In the **Correlation** section, select **Correlation from** or **Correlation to** from the drop-down list, enter an XPath name, and click the field to select an XPath expression. These values enable you to correlate related documents in business transactions. You can only select one option: **Correlation from** or **Correlation to**.
   - The **Correlation from** XPath retrieves the value from the payload to initiate the correlation.
   - The **Correlation to** XPath retrieves the value from the payload for correlation.
8. Click **Save**.

If You Selected Custom

The schema document details page appears.



CustomDoc

Identifier: CUSTOMDOC
Updated on: Mar 22, 2024, 12:50:58 PM PDT

Document selection

Name: CustomDoc

Description

Document standard: Custom

Document version: 1.0.1

Document type: custom_brazil_outbound (User Defined)

Document schema: Brazil-FDG-Outbound_Scherr

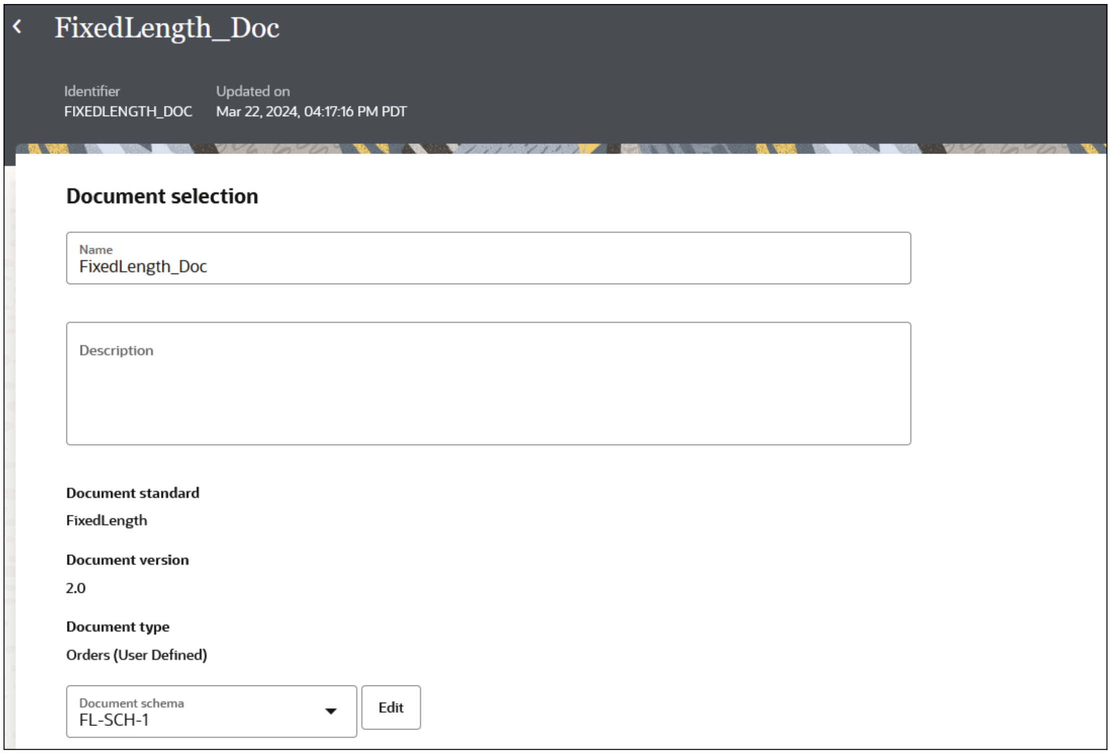
1. If you want to use a different document schema, select an existing schema from the drop-down list or click **Edit** to upload a new schema file.
   You can also define document properties for the schema.
2. Scroll down to the **Document property** section.
3. In the **Business identifiers** section, add at least one identifier (primary). You can also add a second and third identifier to associate with this document. The business identifier names and expressions enable filtering by the identifiers on the Instances page when this document is sent or received.
4. In the **Expression** field, click to select a business identifier expression path.
5. From the **Identification type** drop-down list, select a value.
   a. If you select **XML**, specify an XPath identification expression and value. For example, In the **Identification expression (XPath)** field, specify the identification type as an XPath expression.

      ```
      /*[local-name()='Brazil-FDG_OUTBOUND_Scherr']
      ```

      This expression uniquely identifies the document. For this example, the XPath expression identifies that the document must have a root element of `Brazil-FDG_OUTBOUND_Scherr`.
   b. If you select **Flat**, specify an identification value and identification start and end positions.
6. In the **Correlation** section, select **Correlation from** or **Correlation to** from the drop-down list, enter an XPath name, and click the field to select an XPath expression. These values enable you to correlate related documents in business transactions. You can only select one option: **Correlation from** or **Correlation to**.
   - The **Correlation from** XPath retrieves the value from the payload to initiate the correlation.
   - The **Correlation to** XPath retrieves the value from the payload for correlation.
7. Click **Save**.

If You Selected FixedLength

The schema document details page appears.



FixedLength_Doc

Identifier
FIXEDLENGTH_DOC

Updated on
Mar 22, 2024, 04:17:16 PM PDT

Document selection

Name
FixedLength_Doc

Description

Document standard
FixedLength

Document version
2.0

Document type
Orders (User Defined)

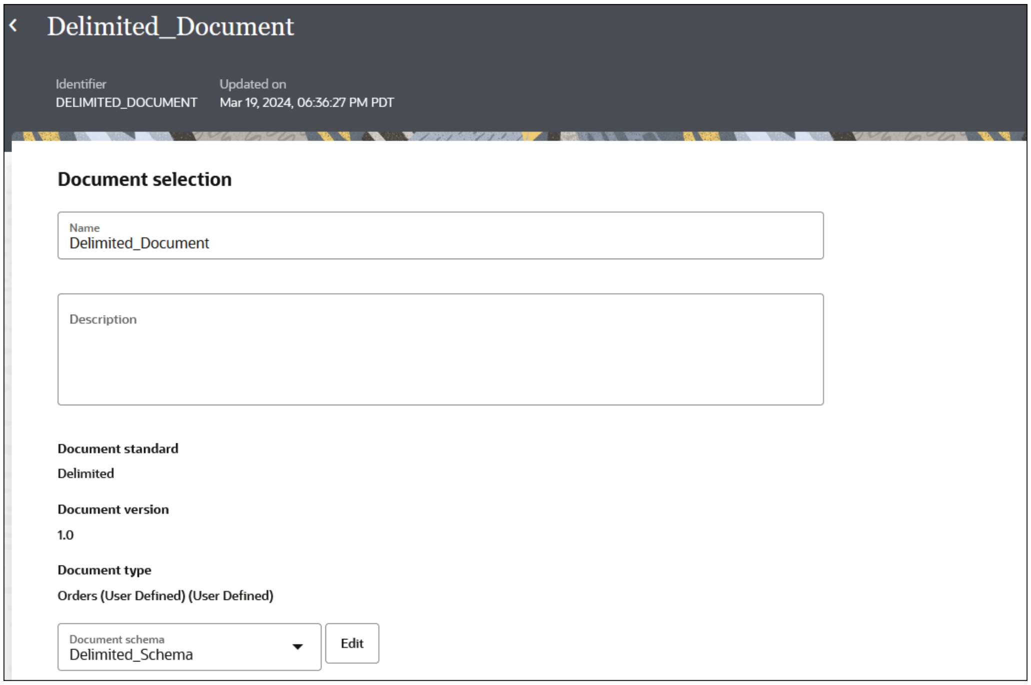
Document schema
FL-SCH-1

1. If you want to use a different document schema, select an existing schema from the drop-down list or click **Edit** to go to the details page for the schema.
   You can also define document properties for the schema.
2. Scroll down to the **Document property** section.
3. In the **Business identifiers** section, add at least one identifier (primary). You can also add a second and third identifier to associate with this document. The business identifier names and expressions enable filtering by the identifiers on the Instances page when this document is sent or received.
4. In the **Expression** field, click to select a business identifier expression path.
   The **Identification type** list shows a value of **Flat**. This value cannot be changed.
5. Specify values.
   - **Identification value**: Enter the segment key for identifying the payload (for example, `BGM`).
   - **Identification start position**: Enter the starting character position for the identifier (for example, `1`).
   - **Identification end position**: Enter the ending character position for the identifier (for example, `3`).
6. In the **Correlation** section, select **Correlation from** or **Correlation to** from the drop-down list, enter an XPath name, and click the field to select an XPath expression. These values enable you to correlate related documents in business transactions. You can only select one option: **Correlation from** or **Correlation to**.
   - The **Correlation from** XPath retrieves the value from the payload to initiate the correlation.
   - The **Correlation to** XPath retrieves the value from the payload for correlation.

7. Click **Save**.

If You Selected Delimited

The schema document details page appears.



Delimited_Document

Identifier
DELIMITED_DOCUMENT

Updated on
Mar 19, 2024, 06:36:27 PM PDT

Document selection

Name
Delimited_Document

Description

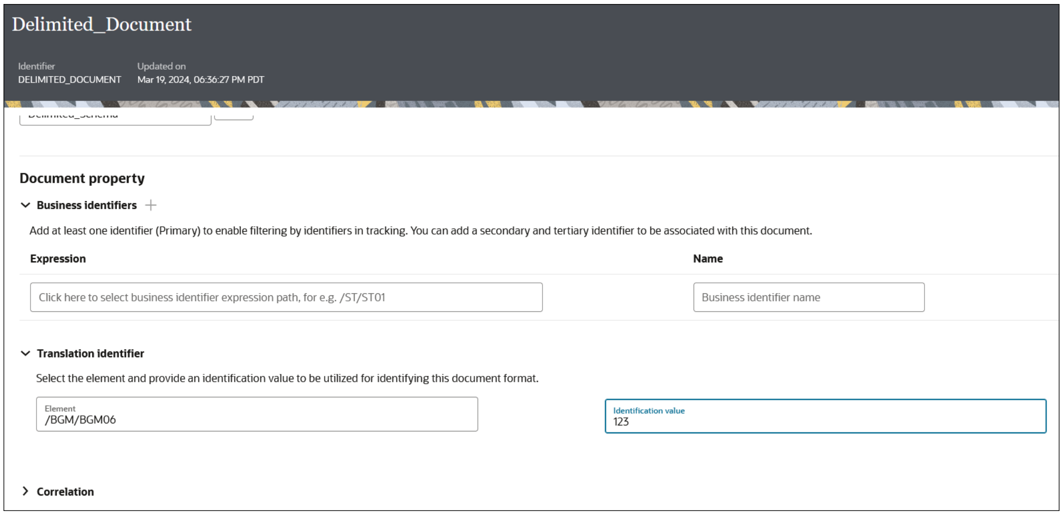
Document standard
Delimited

Document version
1.0

Document type
Orders (User Defined) (User Defined)

Document schema
Delimited_Schema

1. If you want to use a different document schema, select an existing schema from the drop-down list or click **Edit** to go to the details page for the schema.
   You can also define document properties for the schema.
2. Scroll down to the **Document property** section.
3. In the **Business identifiers** section, add at least one identifier (primary). You can also add a second and third identifier to associate with this document. The business identifier names and expressions enable filtering by the identifiers on the Instances page when this document is sent or received.
4. In the **Expression** field, click to select a business identifier expression path.
5. In the **Translation identifier** field, click to select a translation identifier. This identifier is used to identify the document at runtime. For example,



Delimited_Document

Identifier
DELIMITED_DOCUMENT

Updated on
Mar 19, 2024, 06:36:27 PM PDT

Document property

Business identifiers

Add at least one identifier (Primary) to enable filtering by identifiers in tracking. You can add a secondary and tertiary identifier to be associated with this document.

Expression	Name
Click here to select business identifier expression path, for e.g. /ST/ST01	Business identifier name

Translation identifier

Select the element and provide an identification value to be utilized for identifying this document format.

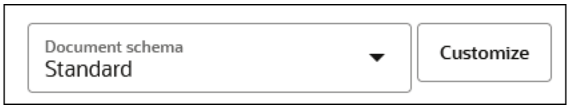
Element: /BGM/BGM06

Identification value: 123

6. In the **Correlation** section, select **Correlation from** or **Correlation to** from the drop-down list, enter an XPath name, and click the field to select an XPath expression. These values enable you to correlate related documents in business transactions. You can only select one option: **Correlation from** or **Correlation to**.
   - The **Correlation from** XPath retrieves the value from the payload to initiate the correlation.
   - The **Correlation to** XPath retrieves the value from the payload for correlation.
7. Click **Save**.

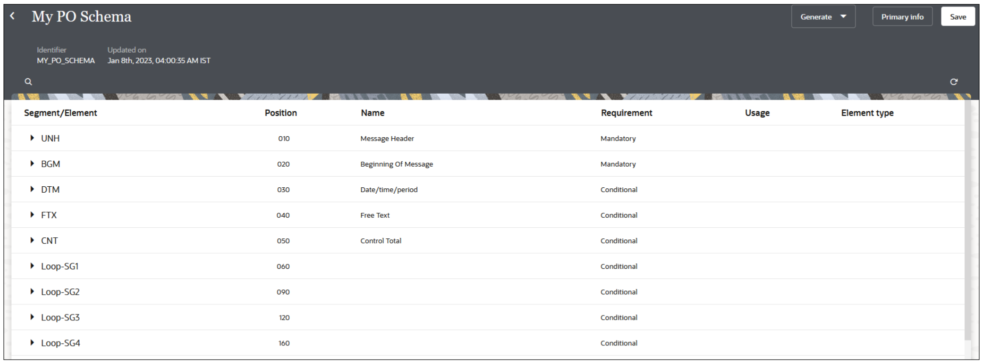
Customize an EDIFACT, X12, or X12HIPAA Document Definition

If you click **Customize** for EDIFACT, X12, or X12HIPAA, the Clone standard schema dialog is displayed. This option enables you to customize the standard schema to satisfy your business requirements.



Document schema: Standard

1. Enter the details to create a new B2B document definition.
2. Click **Save**.
   This action creates a copy of the standard EDIFACT, X12, or X12HIPAA schema for you to use as a baseline to customize. For this example, the EDIFACT schema is shown.

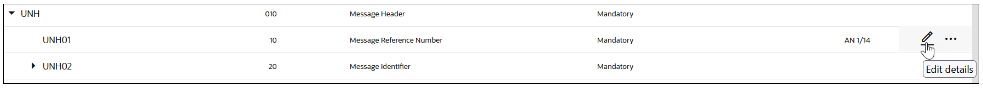


My PO Schema

Identifier MY_PO_SCHEMA
Updated on Jan 8th, 2023, 04:00:35 AM IST

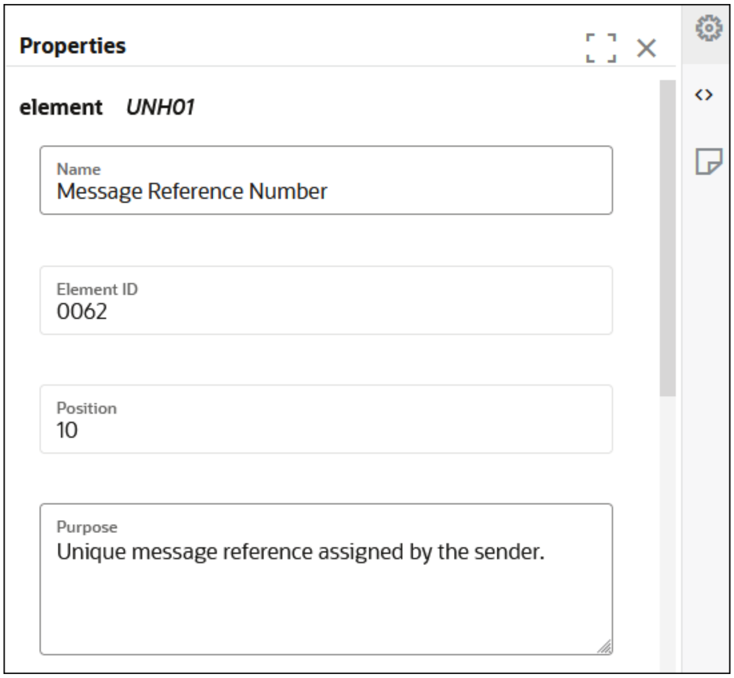
Segment/Element	Position	Name	Requirement	Usage	Element type
UNH	010	Message Header	Mandatory		
BGM	020	Beginning Of Message	Mandatory		
DTM	030	Date/time/period	Conditional		
FTX	040	Free Text	Conditional		
CNT	050	Control Total	Conditional		
Loop-SG1	060		Conditional		
Loop-SG2	090		Conditional		
Loop-SG3	120		Conditional		
Loop-SG4	160		Conditional		

3. On the schema page, find the element you want to customize, and select ✎. As an example, select the **UNH01** currency code element (part of the **UNH** segment) to edit it. You can also add new constructs to the schema. See Edit Properties of Segments and Other Schema Constructs.



UNH	010	Message Header	Mandatory		
UNH01	10	Message Reference Number	Mandatory		AN 1/14
UNH02	20	Message Identifier	Mandatory		

The Properties panel is displayed.



Properties

element *UNH01*

Name
Message Reference Number

Element ID
0062

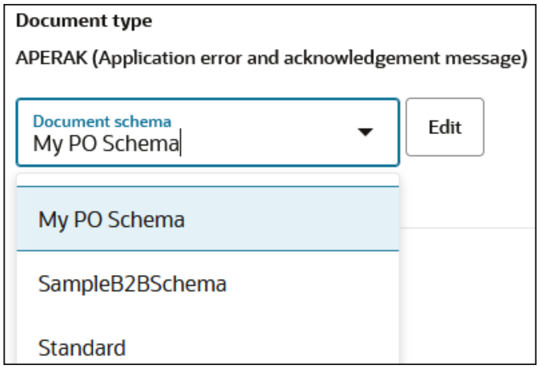
Position
10

Purpose
Unique message reference assigned by the sender.

The three tabs on the Details pane enable you to perform customizations.

Tab	Description
Properties ⚙	Displayed by default when you initially access the Details pane. You can modify the following standard EDI X12 properties for the selected segment or element: • Purpose of element • Requirement (mandatory, optional, or conditional) • Usage (must use, do not use, recommended, or not recommended) • Minimum and maximum number of characters • Number of times to repeat the element
Code List <>	Click to view the code list for the element. A code list defines an enumeration of allowed values for the element.
Notes	Click to add notes to the element.

4. Edit the element details as necessary for your business environment. For this example, select **Code List** <> to add new code.
5. Click **Add a new code list** (if the element does not already have a code list defined).
6. Define the syntax rules.
7. When complete, click **X** in the upper right corner to close the dialog.
8. Return to the B2B document's details page and click **Save** to associate the customized schema to the document.
9. Go to an orchestration integration either to add a new B2B action or edit an already configured B2B action that is not part of an activated integration. See Use the B2B Action in Standalone Mode.
10. Note that the customized document definition is now available to use from the **Document schema** list in the wizard.



Document type

APERAK (Application error and acknowledgement message)

Document schema: My PO Schema

Edit or Clone a B2B Document

You can edit an existing B2B document to associate a different schema or add business identifiers. You can also clone a document to create a copy of it.

1. In the navigation pane, click **B2B**, then **Documents**.

2. Hover over a document row to see the actions you can perform on a B2B document.

3. Click **Edit** ✎ to change the document's name, description, document type or document properties (business identifiers, correlations, and more).

   > **Note:**
   >
   > You cannot change the document standard or version.

5

B2B Tracking

On the Track B2B messages page, you can track the B2B-message interaction between trading partners and work with the messages.

Topics:

Overview

You can track the B2B-message interaction between trading partners on the Track B2B messages page. This page provides B2B-specific message tracking information that is not available on the Instances page for integrations.

To view the Track B2B messages page, click **Observability** in the navigation pane, then click **B2B tracking**.
The page contains the following two tabs:

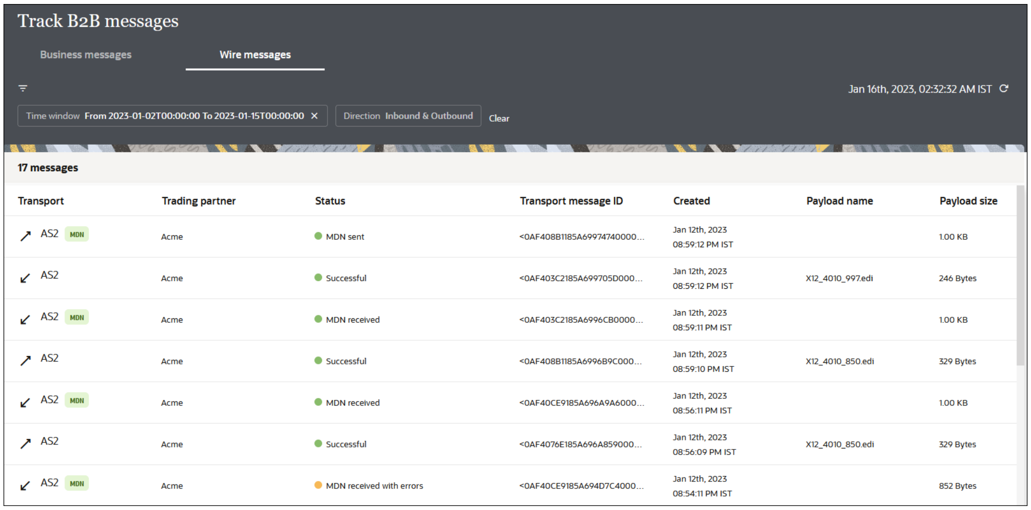
- Wire messages
- Business messages

To understand the type of transactions displayed on each of these tabs, let's consider an example in which an interaction is initiated by an inbound message sent from a trading partner to the host trading partner (site in which Oracle Integration is installed). This interaction contains the following four messages:

1. The host trading partner receives an invoice document (EDI 810) from a trading partner.
2. The host trading partner performs the following actions:
   a. Sends a message disposition notification (MDN) to the trading partner to acknowledge receipt of the invoice.
   b. Parses the EDI document to a format understood by Oracle Integration.
3. The host trading partner sends a functional acknowledgment (EDI 997) to the trading partner.
4. The trading partner sends a message to the host trading partner acknowledging receipt of the EDI 997.

Wire Messages

Wire messages are raw messages exchanged between trading partners. Wire messages flow back and forth between trading partners. All the messages in the document transaction example above are tracked on the **Wire messages** tab.



Track B2B messages

Jan 16th, 2023, 02:32:32 AM IST

Time window From 2023-01-02T00:00:00 To 2023-01-15T00:00:00 ✕ | Direction Inbound & Outbound | Clear

17 messages

Transport	Trading partner	Status	Transport message ID	Created	Payload name	Payload size
↗ AS2 MDN	Acme	● MDN sent	<0AF408B1185A69974740000...	Jan 12th, 2023 08:59:12 PM IST		1.00 KB
↙ AS2	Acme	● Successful	<0AF403C2185A699705D000...	Jan 12th, 2023 08:59:12 PM IST	X12_4010_997.edi	246 Bytes
↙ AS2 MDN	Acme	● MDN received	<0AF403C2185A6996CB0000...	Jan 12th, 2023 08:59:11 PM IST		1.00 KB
↗ AS2	Acme	● Successful	<0AF408B1185A6996B9C000...	Jan 12th, 2023 08:59:10 PM IST	X12_4010_850.edi	329 Bytes
↙ AS2 MDN	Acme	● MDN received	<0AF40CE9185A696A9A6000...	Jan 12th, 2023 08:56:11 PM IST		1.00 KB
↗ AS2	Acme	● Successful	<0AF4076E185A696A859000...	Jan 12th, 2023 08:56:09 PM IST	X12_4010_850.edi	329 Bytes
↙ AS2 MDN	Acme	● MDN received with errors	<0AF40CE9185A694D7C4000...	Jan 12th, 2023 08:54:11 PM IST		852 Bytes

The left column identifies the message direction. Inbound messages (those received by the host trading partner with Oracle Integration) are identified by **Inbound message** ↙ and outbound messages (those sent from the host trading partner) are identified by **Outbound message** ↗.

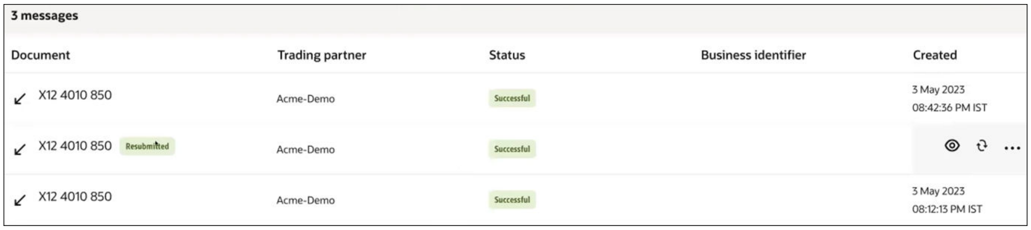
The **Payload name** column shows the payload file that was transmitted.

Business Messages

The **Business messages** tab displays the transactions or messages at the business level as they are being processed by the integrations. For the current example, only the business messages in the interaction are shown on this page. Depending upon the message interaction, multiple business messages can be displayed. When the payload is translated with the EDI parser, an XML message is created that corresponds to this business message.

If the business message includes an invoice, the **Business identifier** column displays the invoice number extracted from the business message. In addition, all the important details, such as processing logs including any errors, payload information, or functional acknowledgments are available for each business message.

The following image shows the business messages in the example interaction:



3 messages

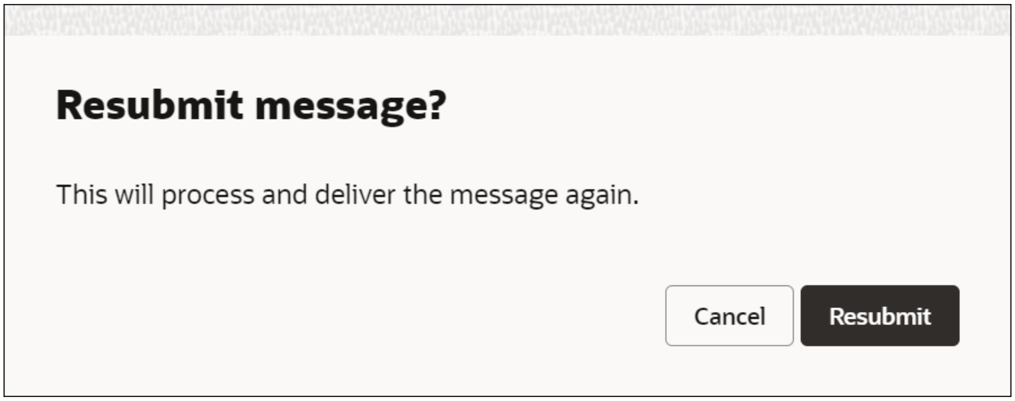
Document	Trading partner	Status	Business identifier	Created
↙ X12 4010 850	Acme-Demo	Successful		3 May 2023 08:42:36 PM IST
↙ X12 4010 850 Resubmitted	Acme-Demo	Successful		◉ ⟳ ⋯
↙ X12 4010 850	Acme-Demo	Successful		3 May 2023 08:12:13 PM IST

Work with Wire Messages

On the Wire messages page, you can resubmit, filter, or view details of wire messages.

1. Resubmit wire messages.

a. For inbound wire messages, the Resubmit feature enables you to reprocess the trading-partner messages from the time of receipt (with older payloads). To reprocess an inbound wire message, click ↻ **Resubmit**. A dialog notifies that the message will be reprocessed and redelivered, click **Resubmit** in the dialog.



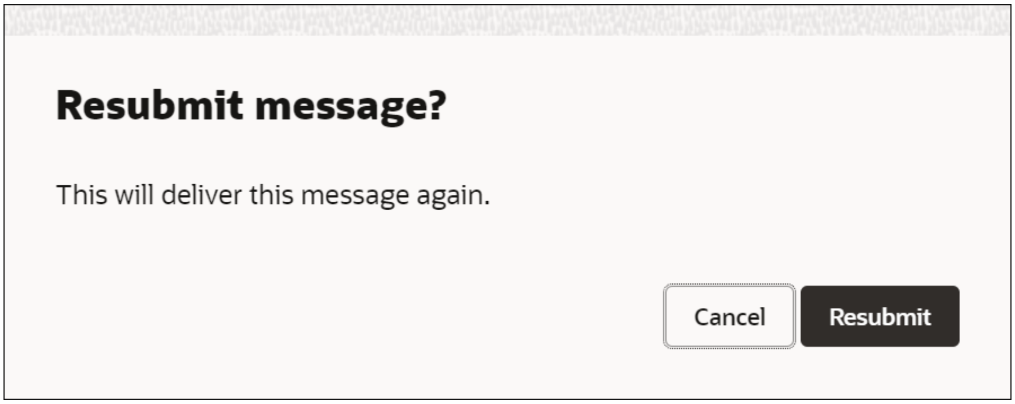
Resubmit message?

This will process and deliver the message again.

Reprocessing an inbound wire message may be required in the following scenarios:

- A failure in unpacking of a wire message
- A failure in generation of business messages for a wire message
- Changes to signatures or certificates

b. For outbound wire messages, the Resubmit feature enables you to redeliver messages to the trading partners. To redeliver an outbound wire message, click ↻ **Resubmit**. A dialog notifies that the message will be redelivered, click **Resubmit** in the dialog.



Resubmit message?

This will deliver this message again.

Redelivering an outbound wire message may be required in the following scenarios:

- Changes in an outbound agreement
- A request from a trading partner for a particular B2B document
- A failure or change in a sent message's encryption or signature

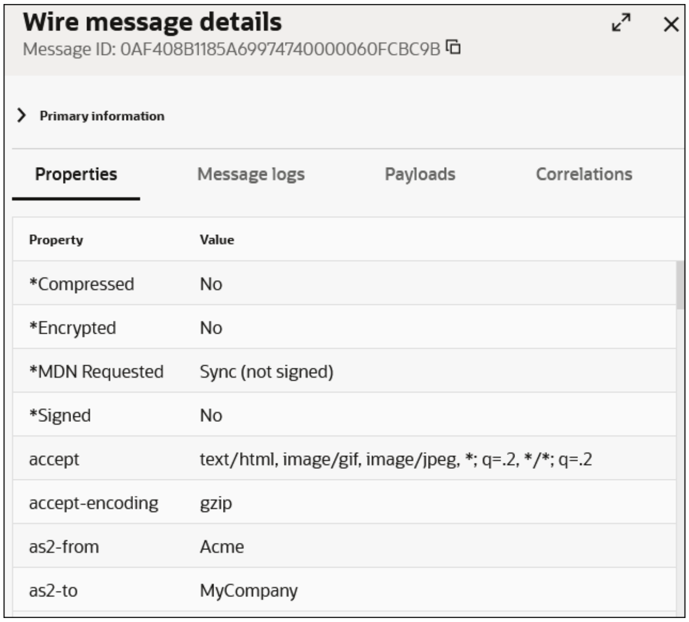
When you resubmit a message, a notification is displayed indicating that it was resubmitted. If you refresh the screen, the message will have the **Resubmitted** label.

Hover the cursor over the label to view the information indicating the user who resubmitted the message and the time at which it was done.

> **Note:**
>
> - You can resubmit both failed and successful transactions.
> - You can resubmit an original message multiple times. However, you cannot resubmit the child messages.
> - You cannot resubmit MDN messages.
> - You can resubmit a message only if the underlying integration has a REST Adapter trigger or an AS2 Adapter trigger, and is allowed to run again.
>   Auto-generated B2B integrations are allowed to run again by default. However, for backend-process integrations, you'll have to manually select the **Allow to run again** option while activating them. See Replay Integration Instances in *Using Integrations in Oracle Integration 3*.

2. View details of wire messages.

   Click **View details** ◎ on a message to open the Wire message details panel.



Wire message details

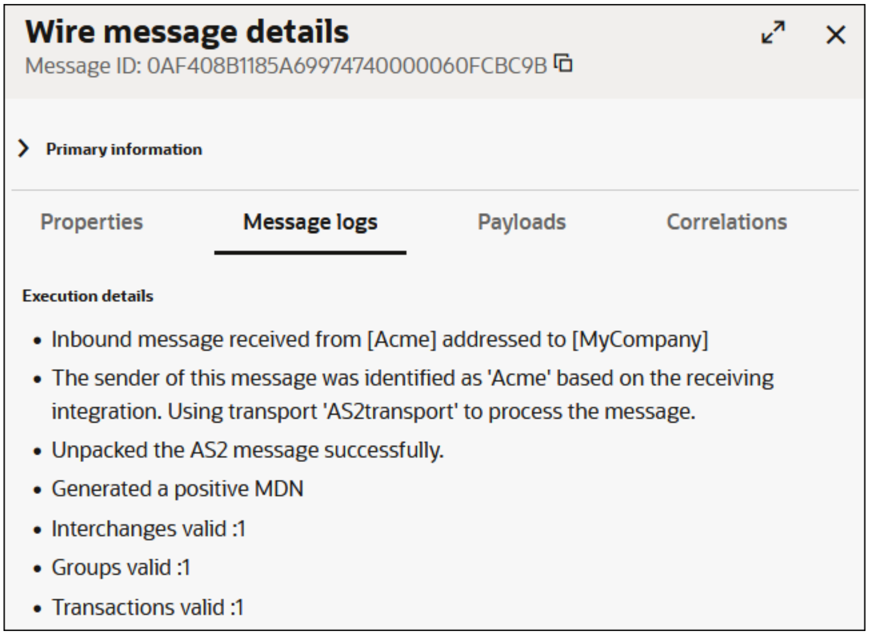
Message ID: 0AF408B1185A69974740000060FCBC9B

Property	Value
*Compressed	No
*Encrypted	No
*MDN Requested	Sync (not signed)
*Signed	No
accept	text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
accept-encoding	gzip
as2-from	Acme
as2-to	MyCompany

The panel contains the following tabs:

- **Properties**: Shows header properties and values (for example, HTTP and MIME).
- **Message logs** (displayed by default): The message log is similar to the activity stream available for integrations. It provides details such as the following:

- The message sent from the trading partner to the host trading partner (for an inbound message). The trading partner names shown are based on the headers that you mapped to the B2B identifiers defined in the agreement. See Create Trading Partners.
- The identity of the trading partner that sent the message.
- The unpacking of the message (preprocessing tasks such as message decryption, signature verification, and others).
- The status of the MDN acknowledgment message sent to the trading partner.
- Additional post-processing tasks.

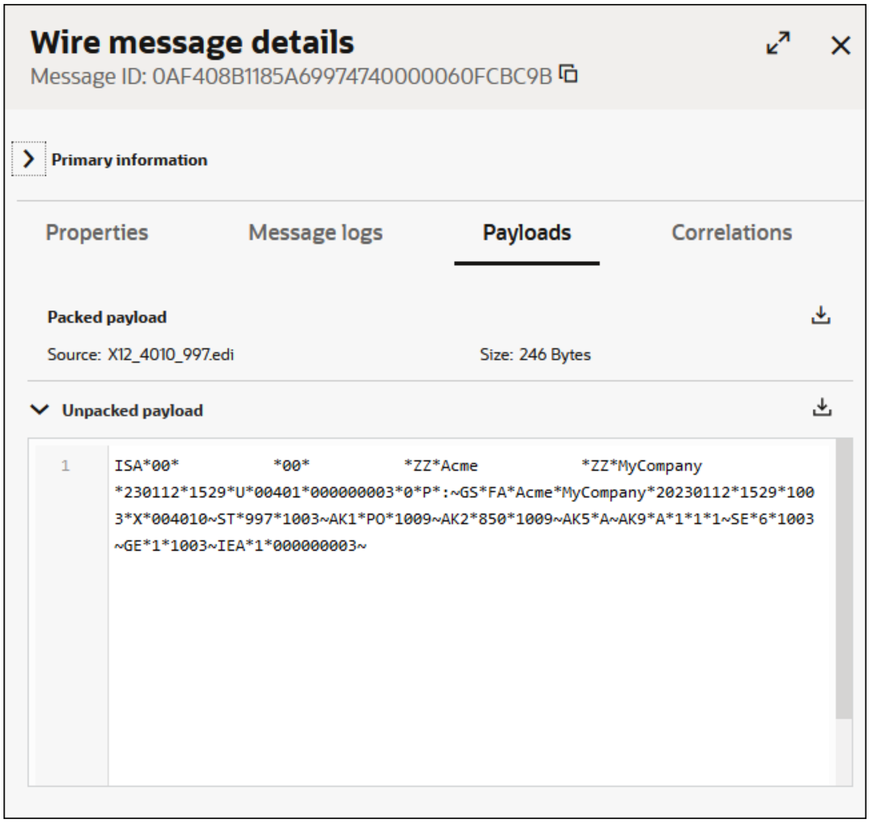


Execution details

- Inbound message received from [Acme] addressed to [MyCompany]
- The sender of this message was identified as 'Acme' based on the receiving integration. Using transport 'AS2transport' to process the message.
- Unpacked the AS2 message successfully.
- Generated a positive MDN
- Interchanges valid :1
- Groups valid :1
- Transactions valid :1

- **Payloads**: Shows the following details:
  - **Packed payload**: Shows the physical EDI payload sent from the trading partner. The payload may be encrypted and signed if both were configured by the trading partner. A signed payload ensures that your message cannot be repudiated.
  - **Unpacked payload**: Shows the raw, unencrypted version of the payload sent from the trading partner. The payload format may be signed if it was configured by the trading partner.

Click **Download** ⤓ to download the packed or unpacked version of the payload.



Wire message details
Message ID: 0AF408B1185A69974740000060FCBC9B

Packed payload

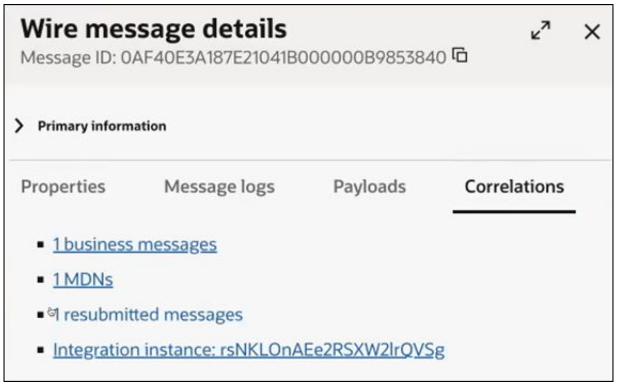
Source: X12_4010_997.edi   Size: 246 Bytes

Unpacked payload

```
ISA*00*          *00*          *ZZ*Acme          *ZZ*MyCompany
*230112*1529*U*00401*000000003*0*P*:~GS*FA*Acme*MyCompany*20230112*1529*100
3*X*004010~ST*997*1003~AK1*PO*1009~AK2*850*1009~AK5*A~AK9*A*1*1*1~SE*6*1003
~GE*1*1003~IEA*1*000000003~
```

- **Correlations**: Shows all messages related to the selected wire message. This tab typically lists the corresponding business messages, MDNs, resubmitted messages, and the integration instance.
  In a resubmitted message, this tab contains a link to the original message. And in the original message, this tab contains a link to all resubmitted messages.

  To view a correlated message, click the corresponding link. Click the integration instance link to view the backend integration that includes this B2B message on the Instances page.



Wire message details
Message ID: 0AF40E3A187E21041B000000B9853840

- 1 business messages
- 1 MDNs
- 1 resubmitted messages
- Integration instance: rsNKLOnAEe2RSXW2lrQVSg

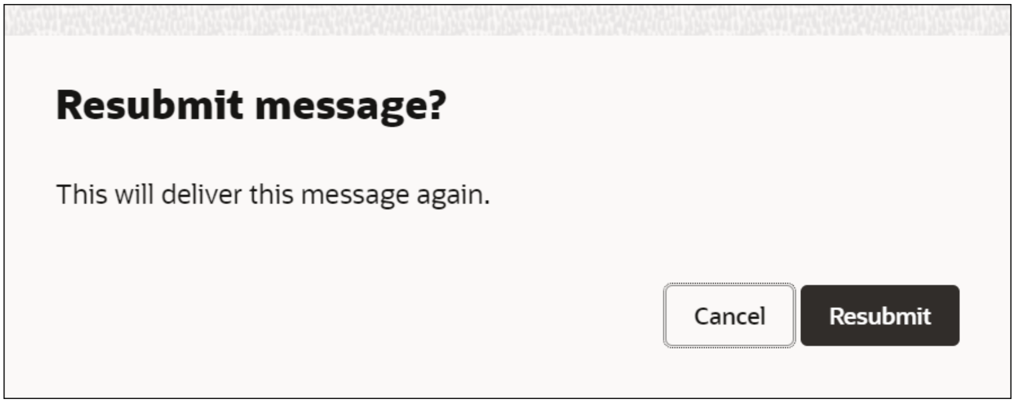
3. Filter wire messages.

Click **Filter** to filter messages by the last message created, time window, trading partner name, direction, message type, status, transport protocol, message ID, transport message ID, integration instance ID, error code, payload name, and parent message ID. In addition, you can also filter messages based on the resubmit status and resubmitted message ID.

Work with Business Messages

On the Business messages page, you can resubmit, filter, or view details of business messages.

1. Resubmit business messages.
   a. For inbound business messages, the Resubmit feature enables you to redeliver messages to the backend applications. To redeliver an inbound business message, click **Resubmit**. A dialog notifies that the message will be redelivered, click **Resubmit** in the dialog.



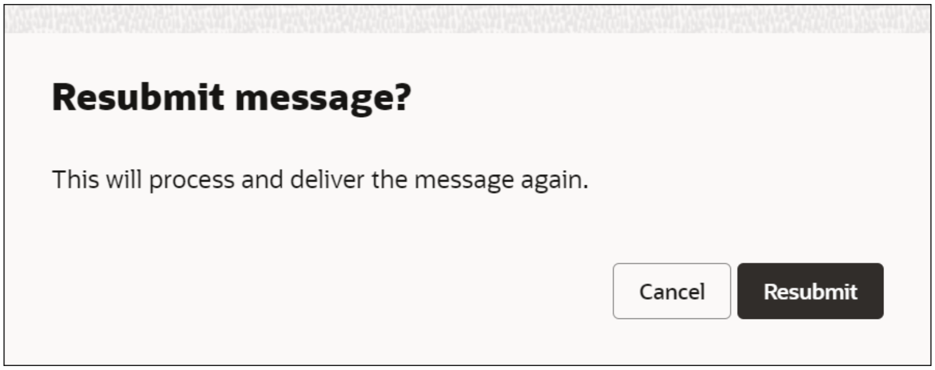
Resubmit message?

This will deliver this message again.

Redelivering an inbound business message may be required in the following scenarios:

- Changes in an inbound agreement
- A request from a backend application for a particular B2B message
- A failure in the backend integration processing

   b. For outbound business messages, the Resubmit feature enables you to reprocess the application messages from the time of receipt (with older payloads). To reprocess an outbound business message, click **Resubmit**. A dialog notifies that the message will be reprocessed and redelivered, click **Resubmit** in the dialog.



Resubmit message?

This will process and deliver the message again.

Reprocessing an outbound business message may be required in the following scenarios:

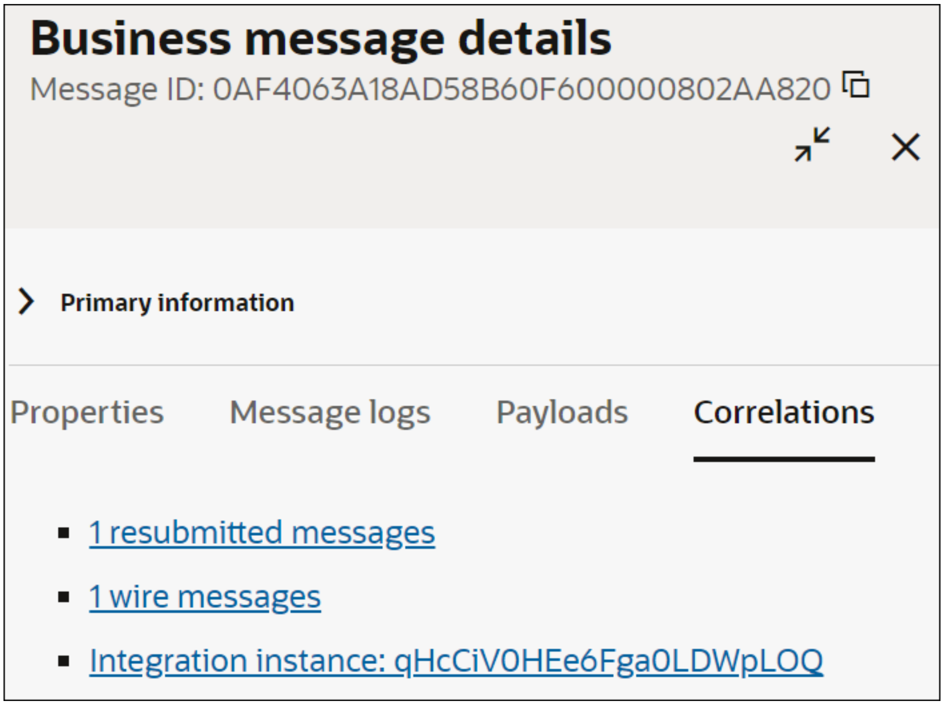
- Issues with the target trading partner's endpoint
- Re-creation of business messages after an update to the agreement or document configurations

When you resubmit a message, a notification is displayed indicating that it was resubmitted. If you refresh the screen, the message will have the **Resubmitted** label. Hover the cursor over the label to view the information indicating the user who resubmitted the message and the time at which it was done.

> **Note:**
>
> - You can resubmit both failed and successful transactions.
> - You can resubmit an original message multiple times. However, you cannot resubmit the child messages.
> - You can resubmit a message only if the underlying integration has a REST Adapter trigger or an AS2 Adapter trigger, and is allowed to run again.
>   Auto-generated B2B integrations are allowed to run again by default. However, for backend-process integrations, you'll have to manually select the **Allow to run again** option while activating them. See Replay Integration Instances in *Using Integrations in Oracle Integration 3*.

2. View details of business messages.

   Click **View details** ◉ on a message to open the Business message details panel.



Business message details

Message ID: 0AF4063A18AD58B60F600000802AA820

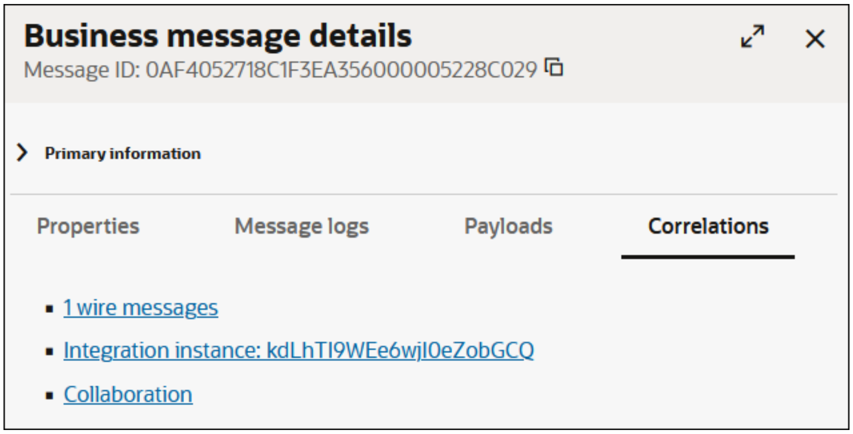
- 1 resubmitted messages
- 1 wire messages
- Integration instance: qHcCiV0HEe6Fga0LDWpLOQ

The panel contains the following tabs:

- **Properties**: Shows the agreement name, document ID, group control number, interchange control number, and transaction control number.
- **Message logs** (displayed by default): The message log is similar to the activity stream available for integrations. It provides the execution details.
- **Payloads**: Shows the payload information related to the business message.
- **Correlations**: Shows all messages related to the selected business message. This tab typically lists the corresponding wire messages, resubmitted messages, and the integration instance.
  In a resubmitted message, this tab contains a link to the original message. And in the original message, this tab contains a link to all resubmitted messages.

  To view a correlated message, click the corresponding link. Click the integration instance link to view the backend integration that includes this B2B message on the Instances page.

  For RosettaNet integrations, a **Collaboration** link is provided.



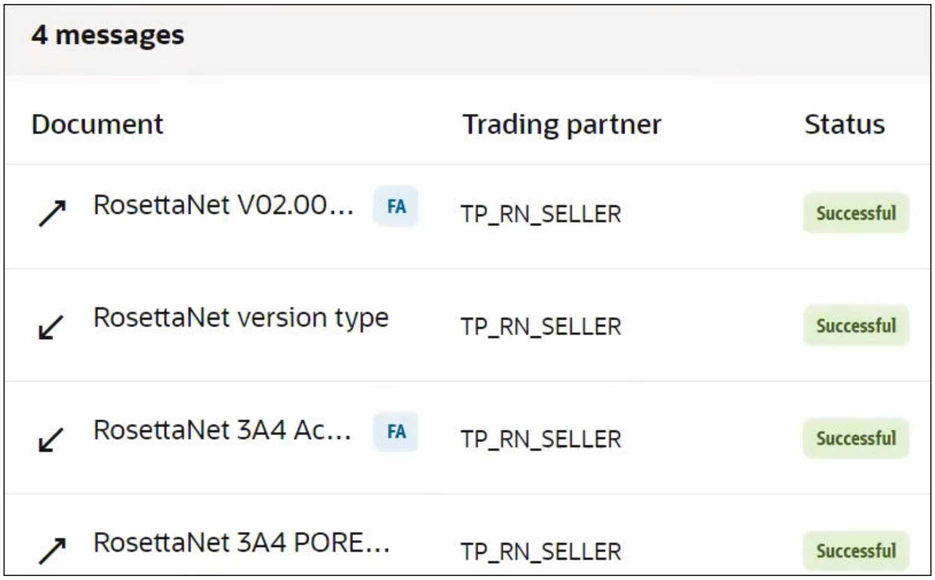
Business message details

Message ID: 0AF4052718C1F3EA356000005228C029

- 1 wire messages
- Integration instance: kdLhTI9WEe6wjl0eZobGCQ
- Collaboration

The **Collaboration** link shows the correlation of messages between buyer and seller. For example, if PIP3A4 is used, the following message correlation is shown.

- Sent by buyer: request(:PurchaseOrderRequestAction)
- Sent by seller: signal(:ReceiptAcknowledgement)
- Sent by seller: response(:PurchaseOrderAcceptanceAction)
- Sent by buyer: signal(:ReceiptAcknowledgement)



4 messages

Document	Trading partner	Status
↗ RosettaNet V02.00… FA	TP_RN_SELLER	Successful
↙ RosettaNet version type	TP_RN_SELLER	Successful
↙ RosettaNet 3A4 Ac… FA	TP_RN_SELLER	Successful
↗ RosettaNet 3A4 PORE…	TP_RN_SELLER	Successful

These messages are visible in both the buyer and seller integrations in the Track B2B messages page.

3. Filter business messages.

   Click **Filter** to filter messages by the last message created, time window, trading partner name, direction, message type, status, document standard, message ID, transport message ID, integration instance ID, error code, parent message ID, and so on.

In addition, you can also filter messages based on the resubmit status and resubmitted message ID.

Monitor the Overall Runtime Health of B2B Transactions

You can view the overall health of B2B for Oracle Integration transactions from the B2B dashboard page. This page provides an easy way to view the number of inbound and outbound messages processed, detect and manage failed messages in your environment, and view the delivery status of trading partner documents.

The B2B dashboard page provides the following capabilities:

- Data visualization: Provides a glimpse into the overall health of your B2B environment. A visual representation of data is shown, making it easier to understand and interpret trends, patterns, and anomalies, which leads to faster decision making and problem solving.
- Real time monitoring: Provides status and performance of B2B transactions for proactive identification and resolution of issues and delays. For example, 100 purchase orders received vs. 90 invoices sent indicates 10 pending invoices due to issues or delays.
- Exception management: Provides a centralized view and highlighting of exceptions that can help you focus on resolving them promptly.

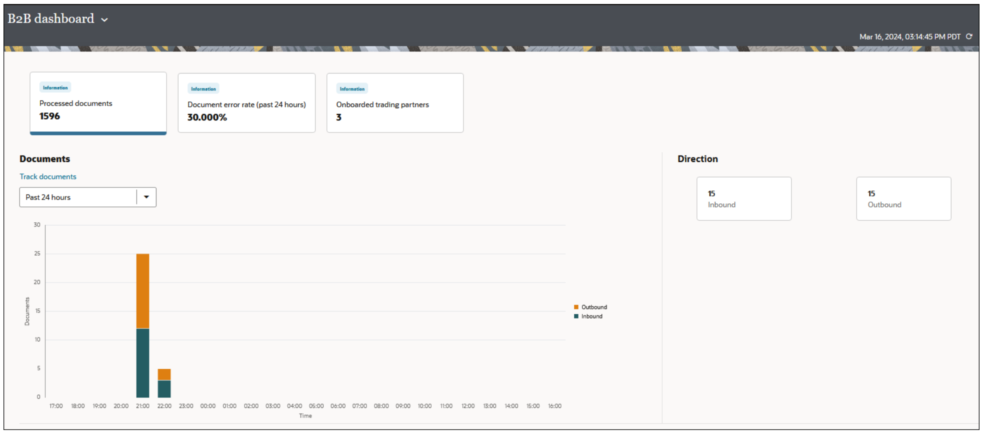
1. In the navigation pane, click **Observability**, then **Dashboards**.
2. From the **Integration dashboard** list at the top, select **B2B**.

   The B2B dashboard is displayed. The following graphical views are available:

   - **Processed documents**: Displays by default the number of inbound and outbound messages processed since the service instance was last started.
   - **Document error rate (past 24 hours)**: Displays by default the percentage of failed messages processed in the last 24 hours.
   - **Onboarded trading partners**: Displays by default the number of trading partners in integrations.

   The B2B dashboard only shows runtime data from Oracle Integration 3 instances. Runtime data migrated from Oracle Integration Generation 2 is not shown in the B2B dashboard.
3. From the drop-down list, select a time range (the past 24 hours, week, month, or the entire retention period of B2B for Oracle Integration).

   For example, if you selected **Past month** for the **Processed documents** view, a chart showing the number of inbound and outbound messages processed daily for that time range is displayed. The **Direction** section shows the total number of processed inbound and outbound messages for that time range.



B2B dashboard

Mar 16, 2024, 03:14:45 PM PDT

Information
Processed documents
1596

Information
Document error rate (past 24 hours)
30.000%

Information
Onboarded trading partners
3

Documents

Track documents

Past 24 hours

Direction

15
Inbound

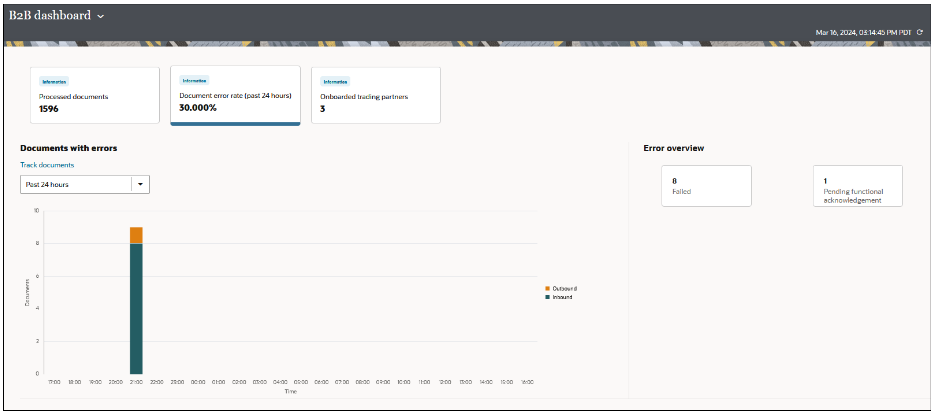
15
Outbound

4. Hover over a bar in the chart to display details about the number of messages processed that day in the inbound and outbound directions.

5. Click the **Document error rate (past 24 hours)** view.

   a. From the drop-down list, select a time range.

      For example, if you selected **Past month**, a chart showing the number of daily failed inbound and outbound messages for that time range is displayed. The **Error overview** section shows the total number of failed messages and pending messages awaiting functional acknowledgment.



B2B dashboard

Mar 16, 2024, 03:14:45 PM PDT

Information
Processed documents
1596

Information
Document error rate (past 24 hours)
30.000%

Information
Onboarded trading partners
3

Documents with errors

Track documents

Past 24 hours

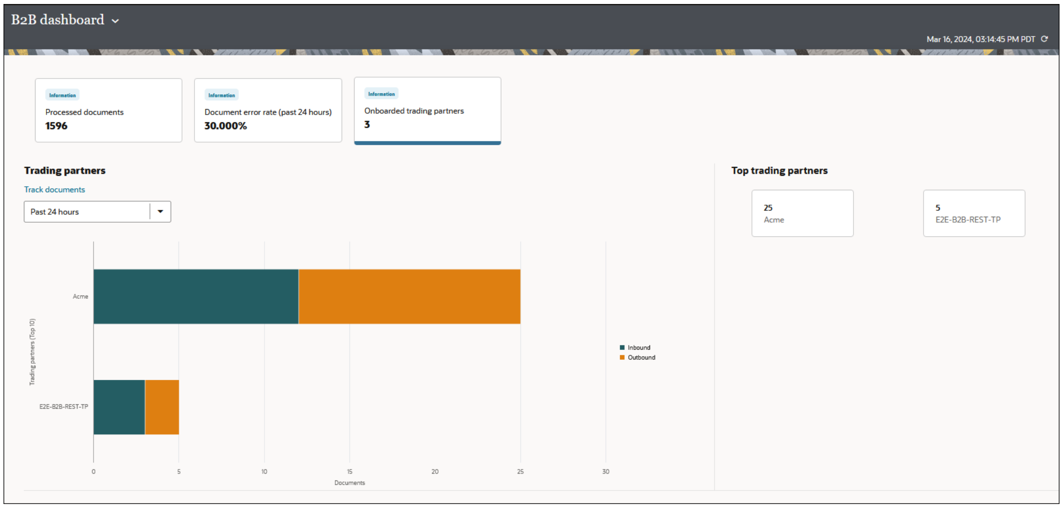
Error overview

8
Failed

1
Pending functional acknowledgement

   b. Hover over a bar in the chart to display details about the number of failed messages processed that day in the inbound and outbound directions.

6. Click the **Onboarded trading partners** view.



B2B dashboard

Mar 16, 2024, 03:14:45 PM PDT

Information
Processed documents
1596

Information
Document error rate (past 24 hours)
30.000%

Information
Onboarded trading partners
3

Trading partners

Track documents

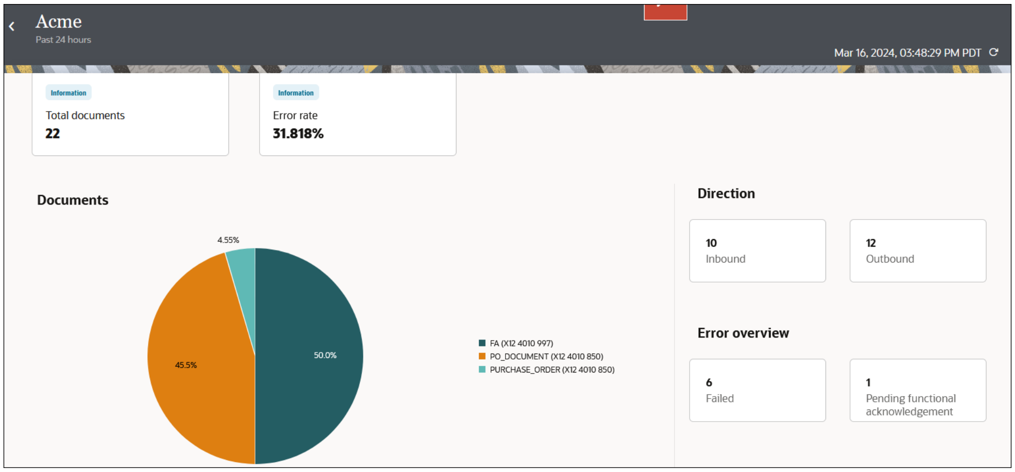
Past 24 hours

Top trading partners

25
Acme

5
E2E-B2B-REST-TP

a. Under **Top trading partners**, click a trading partner to view the number and type of documents delivered (for example, invoices and purchase orders), the number of inbound or outbound documents, and any failed documents or documents waiting for acknowledgment.



Acme
Past 24 hours

Mar 16, 2024, 03:48:29 PM PDT

Information
Total documents
22

Information
Error rate
31.818%

Documents

Direction

10
Inbound

12
Outbound

Error overview

6
Failed

1
Pending functional acknowledgement

b. Hover over a section in the graph to display the total number of messages and the error count.

6
B2B Action XML Schema Reference

This section provides reference information about the B2B action XML schema.

Topics:

- B2B Action Input and Output Schema Reference

B2B Action Input and Output Schema Reference

While configuring data mappings for the B2B action, you use the `TranslateInput` and `TranslateOutput` elements and map the incoming and outgoing data to or from these elements.

This section describes the schema elements contained within the `TranslateInput` and `TranslateOutput` data elements. These schema elements drive the behavior of the B2B action.

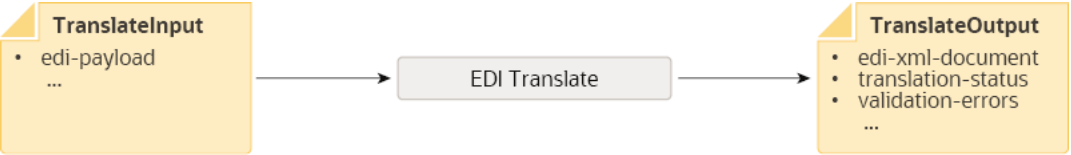
Topics:

- Schema Elements for Inbound EDI
- Sample TranslateOutput XML for Inbound EDI
- Schema Elements for Outbound EDI
- Sample TranslateOutput XML for Outbound EDI

Schema Elements for Inbound EDI

In the inbound EDI use case, the input to the B2B action has the EDI payload element and the output, which is generated after parsing, has the XML form of the EDI document.

The `TranslateInput` data element represents the input message to the B2B action, and the `TranslateOutput` represents the output of the action. Therefore, at a minimum, `TranslateInput` needs to contain the `edi-payload` element to which you'll assign a value using the input map action. The `TranslateOutput` data element then produces the `edi-xml-document` element, returning the parsed data in the XML format along with validation errors, if any.



The following tables describe each element contained in `TranslateInput` and `TranslateOutput` for the inbound EDI scenario.

Elements in TranslateInput

Element	Description
`edi-payload`	You should assign a base64-encoded string value generated from a native EDI X12 payload (starting with the ISA segment and ending with the IEA segment) to this element. Base64 encoding is required because EDI may contain unprintable characters used as delimiters or even binary content such as images. In case the EDI X12 payload contains a batch of transactions, you must use a stage file action along with the B2B action to process the batch.
`tracking-info`	You'll require this element only when you use the B2B action with the stage file action for batch processing. This element contains the tracking information for each EDI transaction that is passed from the stage file action to the B2B action.
`edi-encoding`	The character encoding of the EDI X12 payload (default is UTF-8).

`accept-message-on-errors`
and
`reject-message-on-errors`
	By default, any syntactical error in the input EDI payload is treated as a validation error in translation. You can alter this behavior in two ways:
	1. By turning off validation checks for the current message. Or:
	2. By keeping the validation turned on for the general case. But, be lenient for certain types of validation errors, allowing the message to be processed successfully even if those validation errors were to occur.
	These elements offer the second option listed above by providing a fine-grained control over the types of validation errors to ignore. You may need to have this fine-grained control in specific situations where you want to forward the message to a back-end application; for example, despite the B2B layer detecting invalid data.
	Both these elements are of a string type, and the format of the string allows you to specify one or more validation error types. These elements are semantic opposites of one another.
	The format is as follows:
	`error#<error_code1>,error#<error_code2>,...`
	It is a comma-separated string, where each token starts with `error#` followed by an error code such as `B2B-01752` (for example, a token is: `error#B2B-01752`).
	The `accept-message-on-errors` element defines validation errors to ignore, while the `reject-message-on-errors` element defines validation errors to treat as real errors.
	When a validation error is treated as a real error, the B2B action also generates an acknowledgment message (EDI X12 997) with a rejected flag. However, when a validation error is ignored, the 997 is flagged with the `Accepted, but errors were noted` status.

Element	Description
`validate`	By default, any syntactical error in the input EDI payload is treated as a validation error in translation. You can alter this behavior in two ways: 1. By turning off validation checks for the current message. Or: 2. By keeping the validation turned on for the general case. However, be lenient for certain types of validation errors, allowing the message to be processed successfully even if those validation errors were to occur. This element offers the first option listed above. Setting this element to `false` turns off validation checks, which effectively has the same effect as deselecting the check box **Perform validations on input data?** in the Configure B2B Action Wizard. However, this element overrides the check box setting and can be set for every message (whereas the check box setting is used for all messages). If this element is not set, the check box setting in the wizard controls whether validations are turned on or off.
`functional-ack-required`	If you set this element to `false`, the B2B action does not generate functional acknowledgment messages. If set to `true` or not set, a functional acknowledgment message is included in the `TranslateOutput` element when the B2B action processes the last message in a batch (that is, the last transaction set inside a functional group).
`passthrough-errors`	You'll require this element only when you use the B2B action with the stage file action for batch processing. This element simply transfers certain errors from the stage file action to the B2B action so that a proper functional acknowledgment message can be generated in case of errors. You must not set this element when using the B2B action in standalone mode.
`routing-info`	You'll require this element only when you use the B2B action with the stage file action for batch processing. You must not set this element when using the B2B action in standalone mode.
`input-source-context`	To use this optional element, you can assign any identifier string that the B2B action simply copies to the `TranslateOutput` element. For example, if the B2B action is processing a message from a file, set the file name to this element. The purpose of the element is to make the value available in the output. Therefore, it can be relayed to further actions in your integration or stored inside a tracking variable.

Elements in TranslateOutput

Element	Description
`edi-xml-document`	This element is returned with the translated message in the XML form. See Sample TranslateOutput XML for Inbound EDI.

Element	Description
edi-xml-document > headers > interchange-ctrl edi-xml-document > headers > group	These header elements are returned with values parsed from the EDI X12 ISA and GS envelopes.
edi-xml-document > func-ack-report	This element is returned only when the document type is a 997 (functional acknowledgment), and carries special content specific to a 997. Here are the child elements: • original-tracking-info: The tracking identifier of the original outbound message reconstructed from this functional acknowledgment. This tracking identifier has a substring match with the tracking-info that is generated in TranslateOutput > tracking-info for the outbound EDI case. This is used to correlate an incoming 997 to an outbound message sent to a trading partner originally. For example, the value for original-tracking-info is tc=1013;gc=1013;sn=SenderID;rc=ReceiverID;st=A; and the corresponding value for TranslateOutput > tracking-info is tc=1013;gc=1013;sn=SenderID;rc=ReceiverID;ic=000000015 (these values have a substring match, that means, the part of the string tc=1013;gc=1013;sn=SenderID;rc=ReceiverID from both values have an exact match). • func-ack-status: The overall acceptance status as reported by the trading partner. Values are: Success, Error, or Warning, which correspond to Accepted, Rejected, or Accepted But Errors Were Noted status. • func-ack-details: The interpretation of the functional acknowledgment in a human-readable, plain text format.
edi-xml-document > transaction-data	This element is returned with the hierarchical structure of data segments and elements inside the EDI message, starting with the ST segment and ending with the SE segment. Each segment or loop inside the EDI X12 data becomes a parent XML element and its constituent EDI elements become child XML elements. For example, this may appear inside the transaction-data element: <ST> <ST01>850</ST01> <ST02>1234</ST02> </ST> The above XML fragment represents the data values parsed from the EDI X12 data that has this line: ST*850*1234~ The XML elements <ST01> and <ST02> are assigned values based on their corresponding element positions occurring inside the EDI data.
edi-xml-document > trailers > group edi-xml-document > trailers > interchange-ctrl	The trailer elements are returned with values parsed from the EDI X12 IEA and GE envelopes.

Element	Description
`functional-ack-present`	This element is returned as `true` if an EDI X12 997 functional acknowledgment has been generated in the element `functional-ack`. This element is returned as `false` if no acknowledgment message was needed or generated (if `TranslateInput > functional-ack-required` was set to `false`, then an acknowledgment message is never generated). The B2B action automatically generates a functional acknowledgment message as follows: • For a singleton transaction (for example, when there is a single purchase order inside an EDI message or file): A functional acknowledgment is generated immediately, at the same time as the transaction is processed. • For a batch transaction (for example, when there are fifty purchase orders inside an EDI message or file): A functional acknowledgment is generated when the last transaction inside a batch is processed. A batch means one EDI X12 functional group containing multiple transactions (that is, multiple pairs of `ST`/`SE` segments). The functional acknowledgment then reports the status of all the transactions inside that batch, by including multiple AK2 and AK5 segments. According to the EDI X12 standard, only one functional acknowledgment is expected per batch.
`functional-ack`	If `functional-ack-present` is returned as `true`, the `functional-ack` element is returned with a Base64-encoded EDI X12 997 (functional acknowledgment) document that is ready to be sent to the trading partner. You must apply a Decode64 function to get the native EDI content at the point where you are about to send it out (for example, before writing it to a file to deliver to the trading partner).
`functional-ack-tracking-info`	This element is returned with a string representing a unique transmission identifier for the generated functional acknowledgment message. The string includes control numbers used for the functional acknowledgment. The value is useful for tracking purposes in case the trading partner is unable to process the 997 document.
`validation-errors-present`	This element is returned as `true` if validation errors were detected while parsing the input payload.

`validation-errors` > `error`
`validation-errors` > `validation-error-report`
	The `error` element is returned with validation errors, with each validation error in an XML structure; the error code and error message are returned as separate XML elements. A maximum of 100 validation errors are reported.
	The `validation-error-report` element is returned with a concatenated list of up to 100 validation errors, separated by line break characters.

`tracking-info`	This element is returned with a string representing a unique transmission identifier occurring in the input EDI. This string includes control numbers used in the input EDI message (interchange/group/transaction-set control numbers). This value is useful for tracking purposes.

Element	Description
`translation-status`	This element is returned with the values `Success`, `Warning`, or `Error` as follows: `Success`: Indicates that the input EDI was parsed successfully and no validation errors were detected (also occurs when validations are turned off). `Warning`: Indicates there were validation errors but those were minor or that the user chose to treat certain errors leniently, controlled by the `TranslateInput > accept-message-on-errors` value. `Error`: Indicates critical validation errors were detected while translating and the message is marked as rejected in the functional acknowledgment contents. The integration should not send the message to a backend system for further processing; instead, it should route the message to error handing.
`transaction-summary`	This element is returned with the overall counts of transactions parsed with success, warning, or errors across a batch of transactions.
`input-source-context`	This element is returned as a verbatim copy of the `TranslateInput > input-source-context`, so that it is visible in the integration actions that follow.

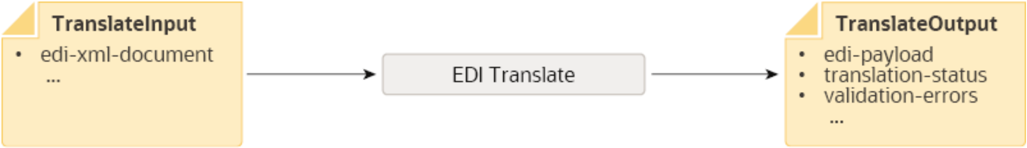
Sample TranslateOutput XML for Inbound EDI

This section provides a sample TranslateOutput XML file for inbound EDI.

```
<ns0:executeResponse xmlns:ns0="http://xmlns.oracle.com/cloud/adapter/
ediAdapter/EDI_Translate_REQUEST/types">
    <TranslateOutput xmlns="http://xmlns.oracle.com/b2b/X12/4010/
transactionset-850/default" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
        <edi-xml-document format="X12-4010-850" ver="1.0.20200326">
            <headers>
                <interchange-ctrl ack-requested-code="0" auth-info=""
auth-info-qualifier="00" ctrl-standards-id="U" ctrl-version-num-
code="00401" date="190312" element-separator="|" interchange-ctrl-
num="000001894" receiver-id="22222" receiver-id-qualifier="01" sec-
info="" sec-info-qualifier="00" segment-terminator="0xa" sender-
id="111111T" sender-id-qualifier="01" subelement-separator="~"
time="0845" usage-indicator-code="T"/>
                <group app-receivers-code="ACME" app-senders-
code="0124578" date="20190312" func-identifier-code="PO" group-ctrl-
num="0123456" resp-agency-code="X" time="084515" version-identifier-
code="004010"/>
            </headers>
            <transaction-data>
                <ST>
                    <ST01>850</ST01>
                    <ST02>1013</ST02>
                </ST>
                <BEG>
                    <BEG01>99</BEG01>
                    <BEG02>BA</BEG02>
                    <BEG03>Purchase Order Number</BEG03>
                    <BEG05>20061103</BEG05>
                    <BEG06>Contract Number</BEG06>
                    <BEG09>01</BEG09>
                </BEG>
                ....
                <SE>
                    <SE01>25</SE01>
                    <SE02>1013</SE02>
                </SE>
            </transaction-data>
            <trailers>
                <group count-of-txnsets="1" group-ctrl-num="1013"/>
                <interchange-ctrl count-of-groups="1" interchange-ctrl-
num="100000020"/>
            </trailers>
        </edi-xml-document>
        <functional-ack-present>true</functional-ack-present>
        <functional-ack>SVNBfDAwfCAgICAgICAgICB8MDB8ICAgICAgICAgI...</
functional-ack>
        <functional-ack-tracking-
info>tc=1016;gc=1016;sn=22222;rc=111111T;ic=000000016;</functional-ack-
tracking-info>
        <validation-errors-present>true</validation-errors-present>
        <validation-errors>
            <error>
                <code>B2B-01757</code>
                <category>data_codelist</category>
                <summary>Element BEG01 (element id 353) contains an invalid
identification code [99] that is not listed in the allowed codes for this
element (for example: 00, 44, 01, 45, 02, 46 ...).</summary>
                <location>segment-BEG &gt; BEG01 (element id 353) | segment
position 2 (starting with ST segment) | element position 1 | character
position 174</location>
            </error>
            <validation-error-report>[1] Error code: B2B-01757 | category:
data_codelist | message: (Severe) Element BEG01 (element id 353) contains an
invalid identification code [99] that is not listed in the allowed codes for
this element (for example: 00, 44, 01, 45, 02, 46 ...). | segment-BEG &gt;
BEG01 (element id 353) | segment position 2 (starting with ST segment) |
element position 1 | character position 174</validation-error-report>
        </validation-errors>
        <tracking-
info>tc=1234;gc=0123456;sn=111111T;rc=22222;ic=000001894;seq=1;tot=1;</
tracking-info>
        <translation-status>Error</translation-status>
    </TranslateOutput>
</ns0:executeResponse>
```

Schema Elements for Outbound EDI

In the outbound EDI use case, the input to the B2B action has the XML form of the EDI document and the output has the document assembled into the EDI format.



The following tables describe each element contained in `TranslateInput` and `TranslateOutput` for the Outbound EDI scenario.

Elements in TranslateInput

Element	Description
`edi-xml-document`	The XML form of the data to use to form the output EDI message during translation. At the minimum, you must assign values to the following child XML elements in the input: • `headers > interchange-ctrl`: – The attribute `sender-id-qualifier` must contain a value for the interchange sender ID qualifier. This value is inserted into the ISA05 element. – The attribute `sender-id` must contain a value for the interchange sender ID qualifier. This value is inserted into the ISA06 element. – The attribute `receiver-id-qualifier` must contain a value for the interchange receiver ID qualifier. This value is inserted into the ISA07 element. – The attribute `receiver-id` must contain a value for the interchange receiver ID qualifier. This value is inserted into the ISA08 element. • `headers > group`: – The attribute `app-senders-code` must contain a value for the application sender's code. This value is inserted into the GS02 element. – The attribute `app-receivers-code` must contain a value for the application receiver's code. This value is inserted into the GS03 element. • `transaction-data:` – The child elements inside it represent the structure of the business message, such as a purchase order. Mapping these elements require knowledge of the EDI X12 segments and elements that form the business message. You must assign values to mandatory segments and elements, and assign values to conditional elements based on the presence of other elements defined by the syntax rules specified for each segment. Other elements and attributes inside headers and trailers are assigned default values, so these are optional. **Note:** The B2B action automatically generates unique control numbers for the interchange, group, and transaction set. **Note:** **Customizing Delimiters for Outbound EDI** By default, the `element-separator` is `*`, the `segment-delimiter` is `~`, and the

Element	Description
	subelement-separator is :. You can override them by specifying values in the input XML as follows: • headers > interchange-ctrl > attribute 'element-separator' • headers > interchange-ctrl > attribute 'subelement-separator' • headers > interchange-ctrl > attribute 'segment-terminator' • headers > interchange-ctrl > attribute 'repetition-separator' (only applies to X12 4020 or later versions) If the delimiter character you want to use is an ASCII printable character, simply assign a string value such as "|" to the corresponding attribute. If you want to use a special character as a delimiter, specify it in a hexadecimal format. For example, 0xA, where 0x is a fixed prefix and A is the hexadecimal representation of the ASCII line-feed character. If your EDI payload uses a different encoding such as UTF-8, you can use the hexadecimal format to specify a Unicode character as a delimiter. For example, specify 0x03A60 to use the Unicode character Φ as a delimiter. Note that the receiving trading partner should be prepared to receive a UTF-8 EDI payload for this to work correctly. Note that the delimiter string value you specify must equate to only a single character. Multiple characters are not allowed. The only exception is the segment-terminator, which allows one character plus an optional ASCII carriage-return and a line-feed character at the end. For example, a segment-terminator value of ~0xd0xa uses ~ followed by <CR> and <LF> as a segment-terminator, which means each EDI segment is output on a separate line, making it more readable.
accept-message-on-errors	See Accept Message on Errors.

Element	Description
`reject-message-on-errors`	See Reject Message on Errors.
`validate`	See Validate.
`input-source-context`	See Input Source Context.

Elements in TranslateOutput

Element	Description
`edi-payload`	This element is returned with the translated message in the EDI format, starting with the X12 ISA header envelope and ending with the IEA trailer envelope. The control numbers are automatically generated and inserted in the EDI content. Note that the EDI content is Base64 encoded.
`validation-errors-present`	See Validation Errors Present.

`validation-errors > error`
`validation-errors > validation-error-report`
	See Validation Errors - Error and Error Report.

`tracking-info`	This element is returned with a string representing a unique transmission identifier occurring in the output EDI. This string includes control numbers used in the output EDI message (interchange/group/transaction-set control numbers). This value is useful for tracking purposes.
`translation-status`	See Translation Status.
`input-source-context`	See Input Source Context.

Sample TranslateOutput XML for Outbound EDI

This section provides a sample TranslateOutput XML file for outbound EDI.

```
<executeResponse xmlns="http://xmlns.oracle.com/cloud/adapter/ediAdapter/
EDI_Generate_REQUEST/types">
    <TranslateOutput xmlns="http://xmlns.oracle.com/b2b/edi/generic/
document">
        <edi-payload>SVNBKjAwKiAgICAgICAgICAqMDAqICAgI...E1fg==</edi-payload>
        <validation-errors-present>true</validation-errors-present>
        <validation-errors>
            <error>
                <code>B2B-01757</code>
                <category>data_codelist</category>
                <summary>Element BEG02 (element id 92) contains an invalid
identification code [00] that is not listed in the allowed codes for this
element (for example: PR, LS, DR, DS, UD, UE ...).</summary>
                <location>edi-xml-document &gt; transaction-data &gt;
segment-BEG &gt; BEG02 (element id 92) | segment position 2 (starting with
ST segment) | element position 2</location>
            </error>
            <validation-error-report>[1] Error code: B2B-01757 |
category: data_codelist | message: (Severe) Element BEG02 (element id
92) contains an invalid identification code [00] that is not listed in
the allowed codes for this element (for example: PR, LS, DR, DS, UD,
UE ...). | edi-xml-document &gt; transaction-data &gt; segment-BEG
&gt; BEG02 (element id 92) | segment position 2 (starting with ST
segment) | element position 2</validation-error-report>
        </validation-errors>
        <tracking-info>tc=1015;gc=1015;sn=Acme Foods;rc=Global
Chips;ic=000000015;</tracking-info>
        <translation-status>Error</translation-status>
    </TranslateOutput>
</executeResponse>
```

7

EDI Standards Reference

This section provides reference information about EDI standards.

Topics:

- EDI Concepts
- How to Access EDI Specifications
- EDI X12
- EDI UN/EDIFACT

EDI Concepts

Electronic Data Interchange (EDI) standards such as X12 and EDIFACT predate XML standards. EDI data is not self-descriptive like XML; it was designed to be machine-readable and compact because networks had very low data bandwidth at the time of their inception. There was no scope to expand the message by adding extra tags or metadata.

To decipher EDI data, you must have a separate schema definition available. B2B for Oracle Integration provides X12 and EDIFACT standard schemas and hides some EDI native format intricacies by translating them to (or from) a canonical XML format. You design the backend integrations and largely work with the canonical format only. However, the logical structure and schema of the canonical XML format closely resembles the EDI structure of segments and elements. That is why you need knowledge of the EDI envelope and data segment structures. You may also need knowledge of the EDI schema if you and your trading partner mutually decide to customize the standard schemas.

This section provides EDI standard-specific reference information, such as envelope structures and code lists for specific elements.

There are standard envelopes used commonly within all X12 document types and within all UN/EDIFACT document types. The envelope structures are also briefly described in the following sections for each standard.

How to Access EDI Specifications

A specification for an X12 or EDIFACT document includes detailed information about the structure and definition of each data segment. Referring to these details, you can decipher the information conveyed in a given EDI document. In B2B for Oracle Integration, the specifications are visible in an interactive schema editor or as an implementation guide that you generate for offline viewing.

Here is how you access these details from your Oracle Integration instance:

- Create a B2B schema for the necessary document type. The schema editor shows an interactive schema for that document in which you can browse the structure and select and view elements, data types, code lists, and other details.

- Generate an implementation guide for the necessary document type from the B2B Schema page (this feature is not available for EDIFACT documents). You can generate an implementation guide for a standard schema or a customized schema.

EDI X12

EDI X12 defines the following envelope structure and envelope code lists.

- Envelope Structure
- Envelope Code Lists

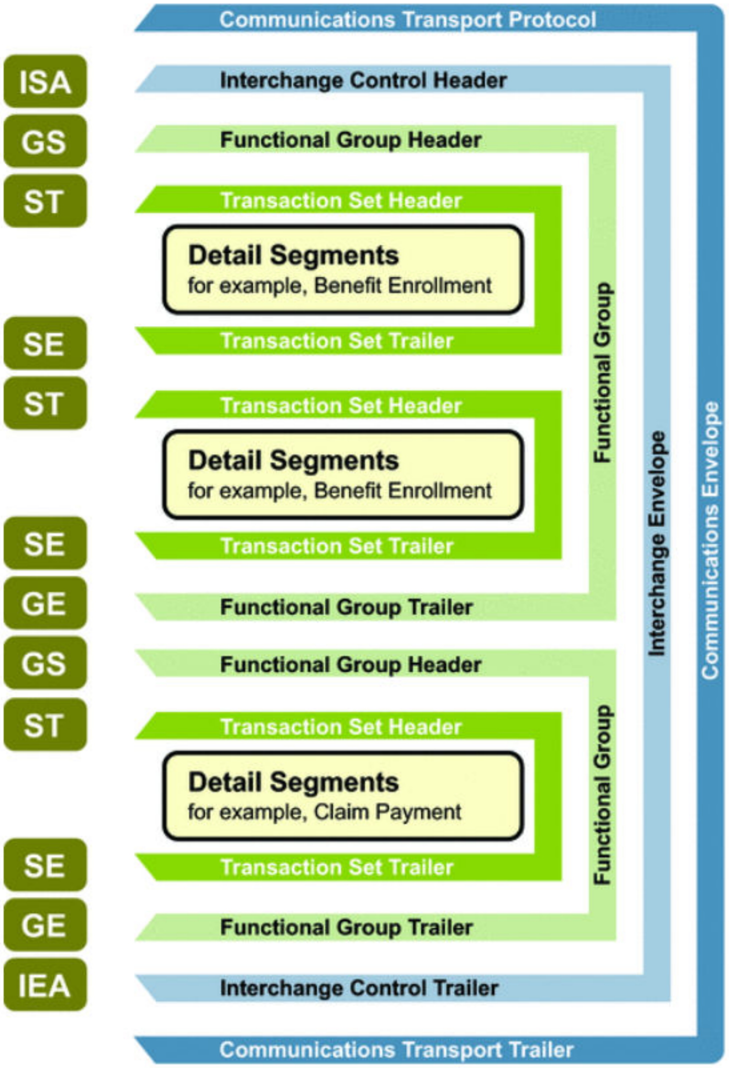
Envelope Structure

The rules for X12 envelope structure ensure the integrity of the data and the efficiency of the information exchange. The actual X12 message structure has primary levels that are hierarchical.

From highest to lowest, the structure is as follows:

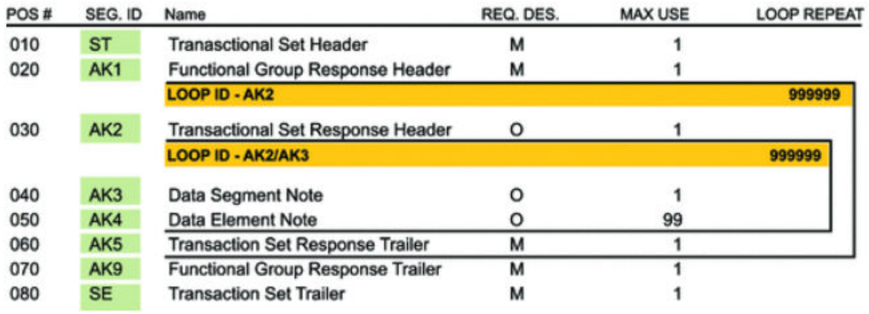
- Interchange envelope
- Functional group
- Transaction set

The following diagram shows a schematic structure of X12 envelopes. Each of these levels is explained in further detail.



Reprinted with permission of X12, Inc. Copyright © 2021, X12, Inc. Format © 2021 Washington Publishing Company (WPC). Exclusively published by WPC under license from X12. All Rights Reserved.

The following example shows the standard segment table for an X12 997 (Functional Acknowledgment) as it appears in the X12 standard and in most industry-specific implementation guides.



POS #	SEG. ID	Name	REQ. DES.	MAX USE	LOOP REPEAT
010	ST	Tranasctional Set Header	M	1	
020	AK1	Functional Group Response Header	M	1	
		LOOP ID - AK2			999999
030	AK2	Transactional Set Response Header	O	1	
		LOOP ID - AK2/AK3			999999
040	AK3	Data Segment Note	O	1	
050	AK4	Data Element Note	O	99	
060	AK5	Transaction Set Response Trailer	M	1	
070	AK9	Functional Group Response Trailer	M	1	
080	SE	Transaction Set Trailer	M	1	

Functional Groups (GS/GE)

Functional groups, often referred to as the inner envelope, consist of one or more transaction sets, all of the same type, which can be batched together into one transmission.

The functional group is defined by the header and trailer segments. The functional group header (designated GS) segment appears at the beginning and the functional group trailer (designated GE) segment appears at the end. Many transaction sets can be included in the functional group, but all local transactions must be of the same type.

Within the functional group, each transaction set is assigned a functional identifier code, which is the first data element of the header segment. The transaction sets that constitute a specific functional group are identified by this functional ID code.

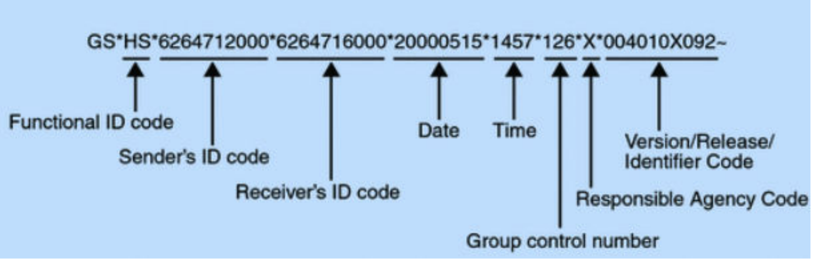
The GS segment contains:

- Functional ID code (the two-letter transaction code; for example, PO for an 850 Purchase Order, HS for a 270 Eligibility, Coverage, or Benefit Inquiry) to indicate the type of transaction in the functional group.
- Identification of the sender and receiver.
- Control information (the functional group control numbers in the header and trailer segments must be identical).
- Date and time.

The GE segment contains:

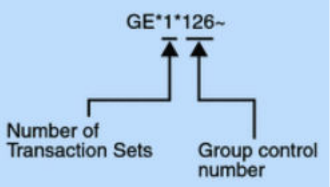
- The number of transaction sets included.
- The group control number (originated and maintained by the sender).

The following example shows a functional group header (GS).



```
GS*HS*6264712000*6264716000*20000515*1457*126*X*004010X092~
```

The following example shows a functional group trailer (GE).



```
GE*1*126~
```

Interchange Envelopes (ISA/IEA)

The interchange envelope, often called the outer envelope, is the wrapper for all the data to send in one transmission. It can contain multiple functional groups. This characteristic means that transactions of different types can be included in the interchange envelope, with each type of transaction stored in a separate functional group.

The interchange envelope is defined by the header and trailer; the interchange control header (designated ISA) appears at the beginning and the interchange control trailer (designated IEA) appears at the end.

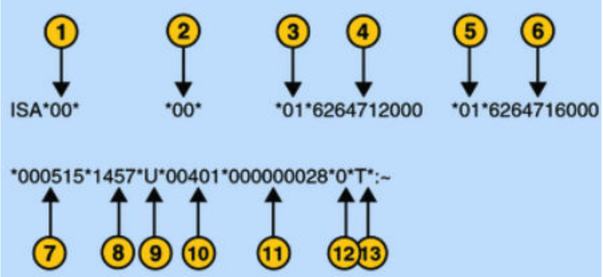
Besides enveloping one or more functional groups, the ISA and IEA segments include:

- Data element separators and data segment terminator
- Identification of sender and receiver
- Control information (used to verify the message was correctly received)
- Authorization and security information, if applicable

The following sequence of information is transmitted:

- ISA
- Optional interchange-related control segments
- Actual message information, grouped by transaction type into functional groups
- IEA

The following example shows an interchange header (ISA).

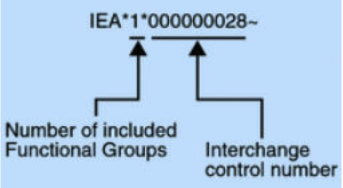


The following list describes the ISA segments:

1. Authorization information qualifier
2. Security information qualifier interchange ID qualifier
3. Interchange sender ID
4. Interchange ID qualifier
5. Interchange receiver ID
6. Date
7. Time
8. Repetition separator
9. Interchange control version number
10. Interchange control number

11. Acknowledgment requested
12. Usage indicator

The following example shows an interchange trailer (IEA).


IEA*1*000000028~
Number of included Functional Groups
Interchange control number

Transaction Sets (ST/SE)

Each transaction set (also known as a transaction) contains:

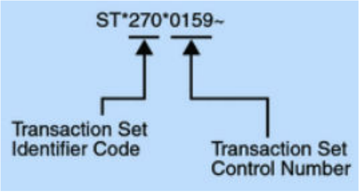
- Transaction set header (designated ST)
- Transaction set trailer (designated SE)
- Single message, enveloped within the header and footer

A transaction set has a three-digit code, a text title, and a two-letter code (for example, 997, Functional Acknowledgment (FA)).

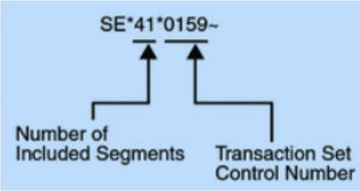
The transaction set is composed of logically-related pieces of information grouped into units called segments. For example, one segment used in the transaction set may convey the address: city, state, postal code, and other geographical information. A transaction set may contain multiple segments. For example, the address segment may be used repeatedly to convey multiple sets of address information.

The X12 standard defines the sequence of segments in the transaction set and also the sequence of elements within each segment. The relationship between segments and elements can be compared to the relationship between records and fields in a database environment.

The following example shows a transaction set header (ST).


ST*270*0159~
Transaction Set Identifier Code
Transaction Set Control Number

The following example shows a transaction set trailer (SE).



Envelope Code Lists

This reference section provides a list of allowed values in X12 envelope segments, ISA and GS, for those elements that must use values from an enumerated code list.

Authorization Information Qualifier (ISA01)

The following table shows the allowed values for the EDI interchange ID qualifier in X12, described in Schema Elements for Outbound EDI.

Value	Description
00 (default)	No Authorization Information Present (No Meaningful Information in I02)
01	UCS Communications ID
02	EDX Communications ID
03	Additional Data Identification
04	Rail Communications ID
05	Department of Defense (DoD) Communication Identifier
06	United States Federal Government Communication Identifier
07 **	Truck Communications ID
08 **	Ocean Communications ID

Note:

The highlighted code values are only valid for X12 versions 5020 and newer.

Security Information Qualifier (ISA03)

The following table shows the allowed values for the security information qualifier in X12, as described in Schema Elements for Outbound EDI.

Value	Description
00 (default)	No Security Information Present (No Meaningful Information in element I04, Security Information)
01	Password

EDI Interchange ID Qualifier (ISA05 and ISA07)

The following table shows the allowed values for the EDI Interchange ID Qualifier in X12, as described in Define the Host Profile and Schema Elements for Outbound EDI (headers > interchange-ctrl > sender-id-qualifier and receiver-id-qualifier).

Value	Description
01	Duns (Dun & Bradstreet)
02	SCAC (Standard Carrier Alpha Code)
03	FMC (Federal Maritime Commission)

Value	Description
04	IATA (International Air Transport Association)
07 **	Global Location Number (GLN) (A globally unique 13 digit code for the identification of a legal, functional, or physical location within the Uniform Code Council (UCC) and International Article Number Association (EAN) numbering system.)
08	UCC EDI Communications ID (Comm ID)
09	X.121 (CCITT)
10	Department of Defense (DoD) Activity Address Code
11	DEA (Drug Enforcement Administration)
12	Phone (Telephone Companies)
13	UCS Code (The UCS Code is a Code Used for UCS Transmissions; it includes the Area Code and Telephone Number of a Modem; it Does Not Include Punctuation, Blanks or Access Code.)
14	Duns Plus Suffix
15	Petroleum Accountants Society of Canada Company Code
16	Duns Number With 4-Character Suffix
17	American Bankers Association (ABA) Transit Routing Number (Including Check Digit, 9 Digit)
18	Association of American Railroads (AAR) Standard Distribution Code
19	EDI Council of Australia (EDICA) Communications ID Number (COMM ID)
20	Health Industry Number (HIN)
21	Integrated Postsecondary Education Data System, or (IPEDS)
22	Federal Interagency Commission on Education, or FICE
23	National Center for Education Statistics Common Core of Data 12-Digit Number for Pre-K-Grade 12 Institutes, or NCES
24	The College Board's Admission Testing Program 4-Digit Code of Postsecondary Institutes, or ATP
25	American College Testing Program 4-Digit Code of Postsecondary Institutions, or ACT
26	Statistics of Canada List of Postsecondary Institutions
27	Carrier Identification Number as assigned by Health Care Financing Administration (HCFA)
28	Fiscal Intermediary Identification Number as assigned by Health Care Financing Administration (HCFA)
29	Medicare Provider and Supplier Identification Number as assigned by Health Care Financing Administration (HCFA)
30	U.S. Federal Tax Identification Number

Value	Description
31	Jurisdiction Identification Number Plus 4 as assigned by the International Association of Industrial Accident Boards and Commissions (IAIABC)
32	U.S. Federal Employer Identification Number (FEIN)
33	National Association of Insurance Commissioners Company Code (NAIC)
34	Medicaid Provider and Supplier Identification Number as assigned by individual State Medicaid Agencies in conjunction with Health Care Financing Administration (HCFA)
35	Statistics Canada Canadian College Student Information System Institution Codes
36	Statistics Canada University Student Information System Institution Codes
37	Society of Property Information Compilers and Analysts NEW
38 **	The College Board and ACT, Inc. 6-Digit Code List of Secondary Institutions
AM	Association Mexicana del Codigo de Producto (AMECOP) Communication ID
NR	National Retail Merchants Association (NRMA) - Assigned
SA **	User Identification Number as assigned by the Safety and Fitness Electronic Records (SAFER) System
SN	Standard Address Number
ZZ	Mutually Defined

Note:

The highlighted code values are only valid for X12 versions 5020 and newer.

There is also no default value used. You must specify a value by adding a B2B identifier to the host profile or trading partner or mapping it in the outbound backend integration.

Interchange Control Standards Identifier (ISA11 - version 4012 and older)

The following table shows the allowed values for the Interchange Control Standards Identifier in X12. See Schema Elements for Outbound EDI.

Note that this element has been repurposed to specify a Repetition Separator in versions 4020 and newer.

Value	Description
U (default)	U.S. EDI Community of ASC X12, TDCC, and UCS

Acknowledgment Requested (ISA14)

The following table shows the allowed values for the Acknowledgment Requested in X12. See Schema Elements for Outbound EDI.

Value	Description
0 (default)	No Interchange Acknowledgment Requested
1	Interchange Acknowledgment Requested (TA1)

Usage Indicator (ISA15)

The following table shows the allowed values for the Usage Indicator in X12. See Schema Elements for Outbound EDI (headers > interchange-ctrl > usage-indicator).

Value	Description
I	Information
P (default)	Production Data
T	Test Data

Responsible Agency Code (GS07)

The following table shows the allowed values for the responsible agency code in X12. See Schema Elements for Outbound EDI (headers > group > resp-agency-code).

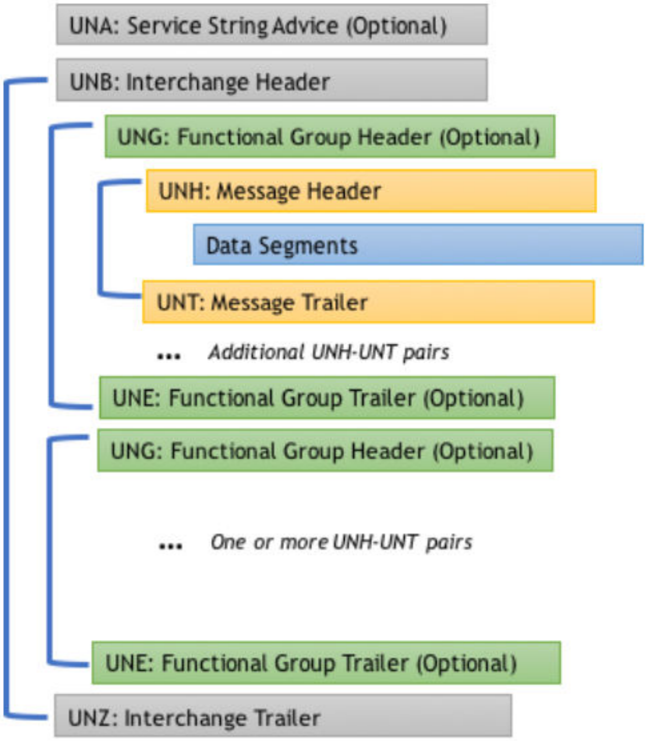
Value	Description
T	Transportation Data Coordinating Committee (TDCC)
X (default)	Accredited Standards Committee X12

EDI UN/EDIFACT

UN/EDIFACT defines the following envelope structure.

Envelope Structure

Full details are available. See Part 4. UN/EDIFACT Rules - Chapter 2.2 Syntax Rules - Annex B.


UNA: Service String Advice (Optional)
UNB: Interchange Header
UNG: Functional Group Header (Optional)
UNH: Message Header
Data Segments
UNT: Message Trailer
... Additional UNH-UNT pairs
UNE: Functional Group Trailer (Optional)
UNG: Functional Group Header (Optional)
... One or more UNH-UNT pairs
UNE: Functional Group Trailer (Optional)
UNZ: Interchange Trailer

EDIFACT envelopes are defined based on a syntax version. You can find details about each element and code lists for each syntax version below:

Version	Reference
Syntax Versions 1, 2, 3	https://www.gefeg.com/jswg/v3/data/v3_docs.htm
Syntax Versions 4	https://www.gefeg.com/jswg/v4/data/v4_docs.htm

8

Migrate B2B Artifacts from Oracle SOA Suite to Oracle Integration

You can migrate B2B artifacts from on-premises Oracle SOA Suite to Oracle Integration.

Topics:

- Migrate B2B Artifacts from Oracle SOA Suite to Oracle Integration

Migrate B2B Artifacts from Oracle SOA Suite to Oracle Integration

You can migrate B2B artifacts from Oracle SOA Suite to Oracle Integration. Migration involves exporting a ZIP file of B2B artifacts, then importing those artifacts into Oracle Integration through use of a REST OpenAPI interface and a REST client tool such as Postman.

- Which B2B Artifacts Can Be Migrated
- How Do You Migrate B2B Artifacts
- Migrate B2B Artifacts to Oracle Integration

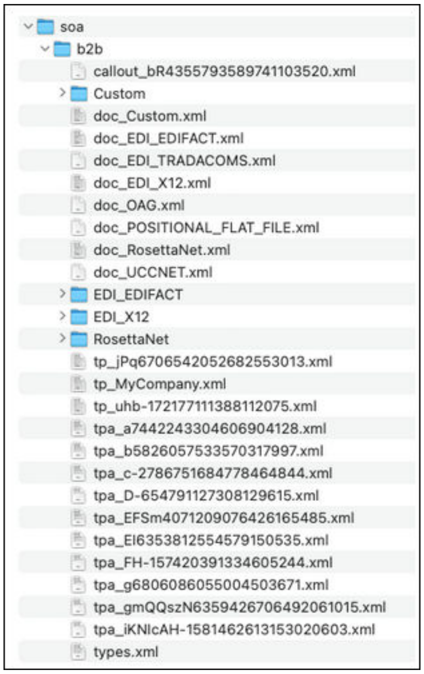
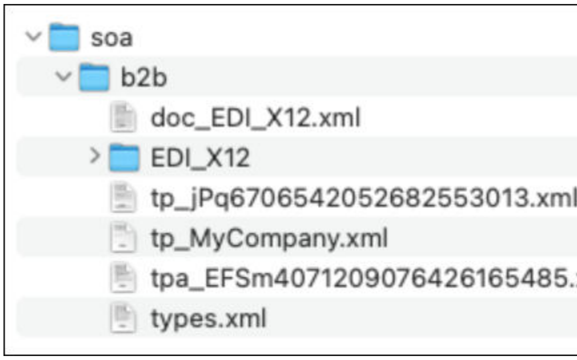
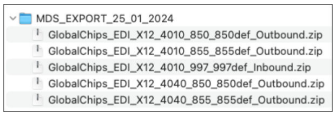
Which B2B Artifacts Can Be Migrated

Before beginning migration, it is important to understand which B2B artifacts in Oracle SOA Suite can be migrated to Oracle Integration.

Can Be Migrated	Cannot Be Migrated
• Host profile • Trading partners (properties, contacts, identifiers) • Transport protocols (AS2, REST, FTP, RosettaNet) • Inbound and outbound agreements • Agreement parameters • Transport connections • Transport template integrations • Schemas and schema parameters (EDI, OAG, custom, and RosettaNet) • Documents and document parameters (EDI, OAG, custom, and RosettaNet)	The following artifacts are ignored if included in the exported ZIP file. • SOA connections / endpoints • SOA composite applications • SOA security configurations • HL7 document protocol • The following delivery/transport protocols: – AS1 – AS4 – ebMS, – MLLP – Generic-AQ – Generic-JMS – Genaric-Email – Generic TCP – Generic MFT

How Do You Migrate B2B Artifacts

You create an export file of B2B artifacts that consists of one or multiple ZIP files. The following types of B2B export files are supported for migration.

Scope of Export	Description	Example
Entire repository (in a ZIP file)	The ZIP file consists of all data in the B2B design-time repository, including agreements in all states, all trading partner configurations, and so on.	 soa b2b callout_bR4355793589741103520.xml Custom doc_Custom.xml doc_EDI_EDIFACT.xml doc_EDI_TRADACOMS.xml doc_EDI_X12.xml doc_OAG.xml doc_POSITIONAL_FLAT_FILE.xml doc_RosettaNet.xml doc_UCCNET.xml EDI_EDIFACT EDI_X12 RosettaNet tp_jPq6706542052682553013.xml tp_MyCompany.xml tp_uhb-1721771113881122075.xml tpa_a7442243304606904128.xml tpa_b5826057533570317997.xml tpa_c-2786751684778464844.xml tpa_D-654791127308129615.xml tpa_EFSm4071209076426165485.xml tpa_EI6353812554579150535.xml tpa_FH-1574203913346052244.xml tpa_g6806086055004503671.xml tpa_gmQQszN6359426706492061015.xml tpa_iKNIcAH-1581462613153020603.xml types.xml
Single agreement (in a ZIP file)	The ZIP file consists of data in one selected active agreement, including all data in the B2B design-time repository such as all agreements in active states, all trading partner configurations, and so on.	 soa b2b doc_EDI_X12.xml EDI_X12 tp_jPq6706542052682553013.xml tp_MyCompany.xml tpa_EFSm4071209076426165485. types.xml
Multiple agreements (one ZIP file per agreement inside a parent ZIP file)	Multiple selected active agreements, including all data in the B2B design-time repository, such as agreements in active states, all trading partner configurations, and so on.	 MDS_EXPORT_25_01_2024 GlobalChips_EDI_X12_4010_850_850def_Outbound.zip GlobalChips_EDI_X12_4010_855_855def_Outbound.zip GlobalChips_EDI_X12_4010_997_997def_Inbound.zip GlobalChips_EDI_X12_4040_850_850def_Outbound.zip GlobalChips_EDI_X12_4040_855_855def_Outbound.zip

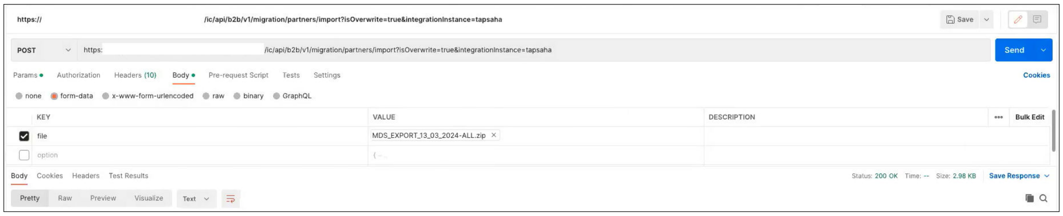
Migrate B2B Artifacts to Oracle Integration

1. Export the Oracle B2B artifacts in Oracle SOA Suite. See Importing and Exporting Data in *Using Oracle B2B*. The export utility performs the following tasks:

- Exports all resources: Exports all active and inaction agreements.
- Exports selected agreements: Allows only active agreements.

When the ZIP file is imported into Oracle Integration, whatever the specific contents are of the file are imported.

2. Invoke the REST OpenAPI interface to import the ZIP file. See Trading Partner Migration from SOA-B2B REST Endpoints in the B2B section of the *REST API for Oracle Integration 3*.
3. Open a REST client. For this example, Postman is used.
4. In the **Value** column, click **Select Files**.
5. Select the ZIP file to import into Oracle Integration, then click **Send**.



https:// /ic/api/b2b/v1/migration/partners/import?isOverwrite=true&integrationInstance=tapsaha

POST https: /ic/api/b2b/v1/migration/partners/import?isOverwrite=true&integrationInstance=tapsaha

○ none ● form-data ○ x-www-form-urlencoded ○ raw ○ binary ○ GraphQL

	KEY	VALUE	DESCRIPTION
☑	file	MDS_EXPORT_13_03_2024-ALL.zip ×	
□	option		

Status: 200 OK Time: -- Size: 2.98 KB

6. Sign in to Oracle Integration.
7. In the navigation pane, click **B2B**, then **Trading partners**.
8. Open each trading partner to see the artifacts populated in the **Properties**, **Contacts**, **B2B identifiers**, and **Transports & agreements** sections.
9. Click **View** ◎ for a transport name in the **Transports & agreements** section and note that a placeholder connection has been populated into the **Trading partner's connection (trigger and invoke)** field.
10. Scroll to the bottom of the **Transports & agreements** section and note that template integrations have been generated.
11. Click **Host profile**, **Schemas**, and **Documents** to see the imported artifacts.
12. In the navigation pane, click **Design**, then **Connections**.
13. Note that a connection has been created for the transport. Update your connection with the endpoint to use. Perform this task for any other connections created by the import.
14. Return to the **Transports & agreements** section.
15. Find the transport name with the placeholder connection.
16. From the **Actions** ••• menu, select **Deploy** to deploy the transport. Perform this task for any other connections created by the import.
17. In the navigation pane, click **Design**, then **Integrations**. to view the template integrations that were created.

9 B2B for Oracle Integration Use Cases

This section describes B2B for Oracle Integration use cases.

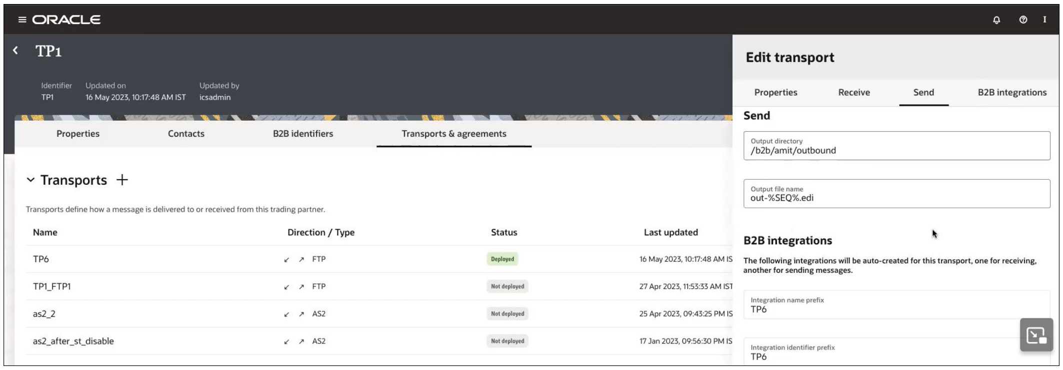
Topics:

- Specify File Name and Directory Name Values at Runtime for Outbound FTP Transport Integrations
- Design a B2B Standalone Mode Integration

Specify File Name and Directory Name Values at Runtime for Outbound FTP Transport Integrations

You can specify the file name and directory name values at runtime for outbound integrations using the FTP transport. If not specified in the integration payload at runtime, the values fall back to the transport property values specified during design time.

1. In the navigation pane, click **B2B**, then **Trading partners**.
2. Configure your trading partner, including selecting the FTP transport protocol for your outbound trading partner and specifying values for the output directory and output file name. For this example:
   - **Output directory**: `/b2b/amit/outbound`
   - **Output file name**: `out-%SEQ%.edi`



Identifier: TP1
Updated on: 16 May 2023, 10:17:48 AM IST
Updated by: icsadmin

Transports define how a message is delivered to or received from this trading partner.

Name	Direction / Type	Status	Last updated
TP6	↙ ↗ FTP	Deployed	16 May 2023, 10:17:48 AM IS
TP1_FTP1	↙ ↗ FTP	Not deployed	27 Apr 2023, 11:53:33 AM IST
as2_2	↙ ↗ AS2	Not deployed	25 Apr 2023, 09:43:25 PM IS
as2_after_st_disable	↙ ↗ AS2	Not deployed	17 Jan 2023, 09:56:30 PM IS

Send

Output directory: /b2b/amit/outbound

Output file name: out-%SEQ%.edi

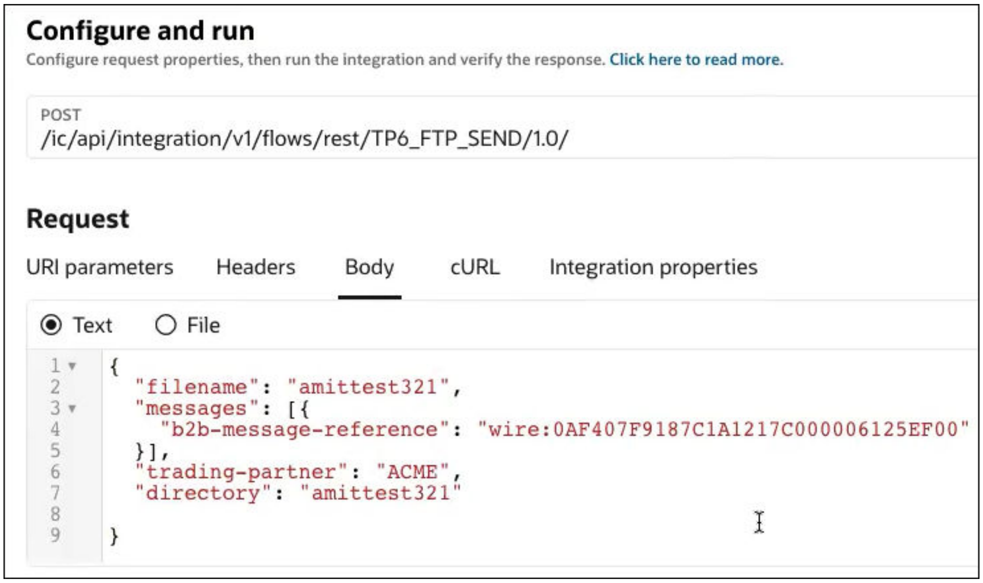
B2B integrations

The following integrations will be auto-created for this transport, one for receiving, another for sending messages.

Integration name prefix: TP6

Integration identifier prefix: TP6

3. Complete your integration design.
4. In the navigation pane, click **Design**, then **Integrations**.
5. Hover your cursor over the integration to run and select **Actions** ⋯, then **Run**.
6. In the **Request** section, click **Body** and specify values for `filename` and `directory`. For this example, `amittest321` is specified for both properties.



Configure and run

Configure request properties, then run the integration and verify the response. Click here to read more.

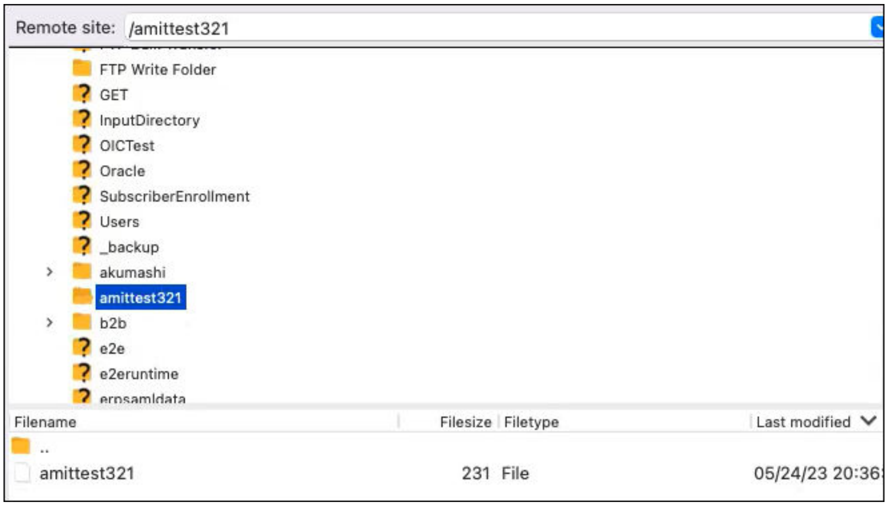
POST
/ic/api/integration/v1/flows/rest/TP6_FTP_SEND/1.0/

Request

```
{
  "filename": "amittest321",
  "messages": [{
    "b2b-message-reference": "wire:0AF407F9187C1A1217C000006125EF00"
  }],
  "trading-partner": "ACME",
  "directory": "amittest321"

}
```

7. Click **Run**.
8. Check the activity stream to verify that the run completed successfully.
9. Open the outbound directory and note that the directory name and file name reflect the values specified at runtime.



Remote site: /amittest321

Filename	Filesize	Filetype	Last modified
..			
amittest321	231	File	05/24/23 20:36

Design a B2B Standalone Mode Integration

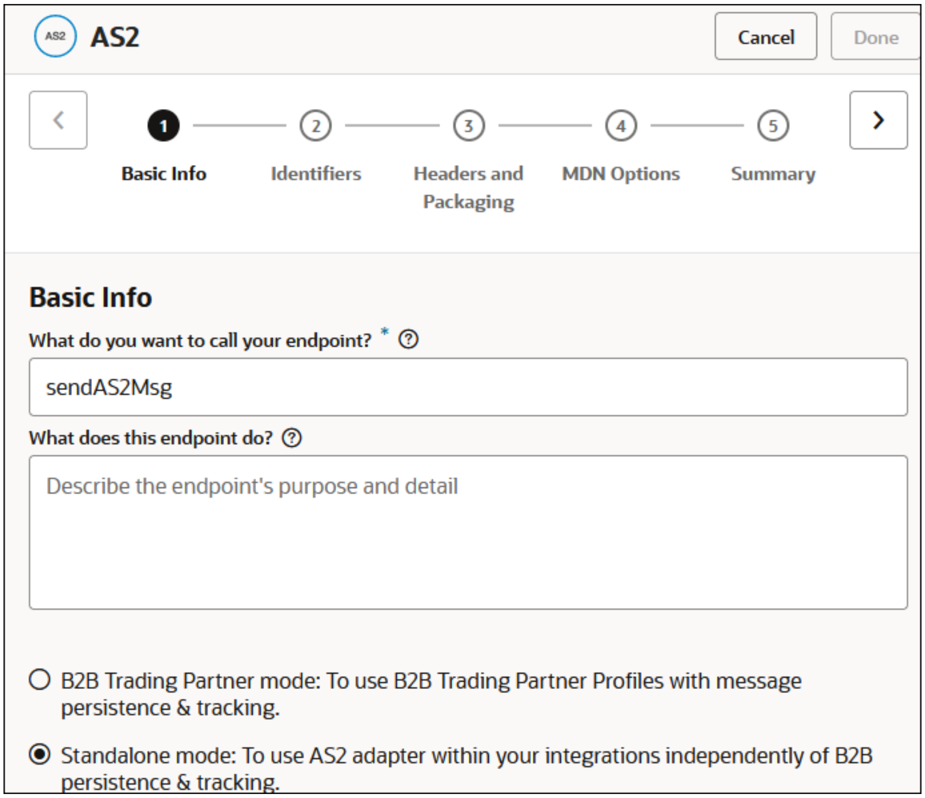
You can design integrations in B2B standalone mode in Oracle Integration without use of trading partner definitions. This section provides a high-level overview of how to configure the AS2 Adapter and the B2B action in a B2B standalone integration to send an outbound EDI message to a business partner.

1. Create an app-driven orchestration integration.
2. Add and configure a REST Adapter as a trigger connection.

3. Add an AS2 Adapter as an invoke connection.

   The Adapter Endpoint Configuration Wizard is displayed.

4. On the Basic Info page, select **Standalone mode**.



Basic Info

What do you want to call your endpoint? *

sendAS2Msg

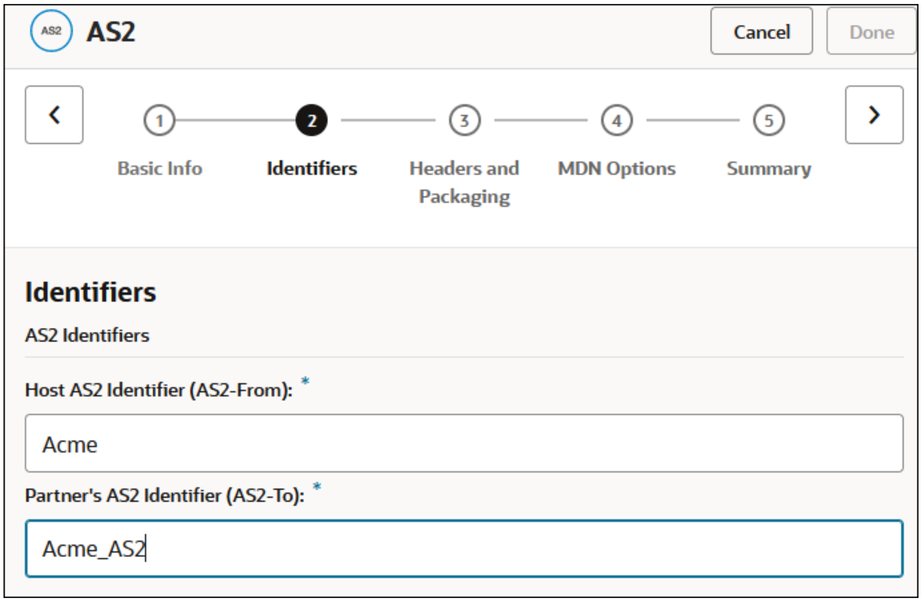
What does this endpoint do?

Describe the endpoint's purpose and detail

○ B2B Trading Partner mode: To use B2B Trading Partner Profiles with message persistence & tracking.

● Standalone mode: To use AS2 adapter within your integrations independently of B2B persistence & tracking.

5. On the Identifiers page, specify the host trading partner identifier (sending the message) and the target trading partner identifier (receiving the message). These fields are required in the outbound direction and optional in the inbound direction.



Identifiers

AS2 Identifiers

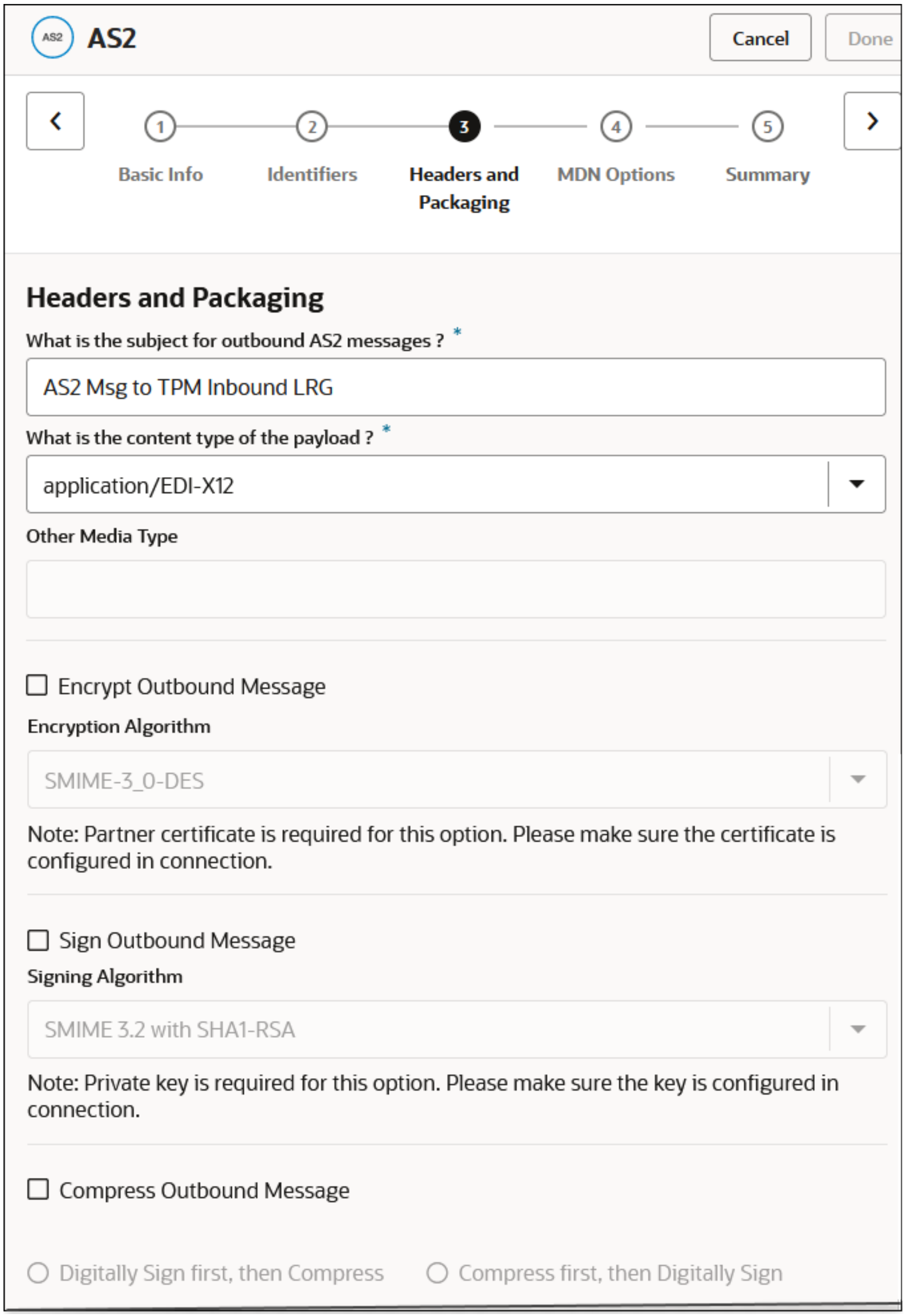
Host AS2 Identifier (AS2-From): *

Acme

Partner's AS2 Identifier (AS2-To): *

Acme_AS2

6. On the Headers and Packaging page, enter the subject, the context type of the payload, and (optionally) the encryption, signing, and compression details, as needed. For this example, EDI-X12 is the content type.



Headers and Packaging

What is the subject for outbound AS2 messages ? *

AS2 Msg to TPM Inbound LRG

What is the content type of the payload ? *

application/EDI-X12

Other Media Type

□ Encrypt Outbound Message

Encryption Algorithm

SMIME-3_0-DES

Note: Partner certificate is required for this option. Please make sure the certificate is configured in connection.

□ Sign Outbound Message

Signing Algorithm

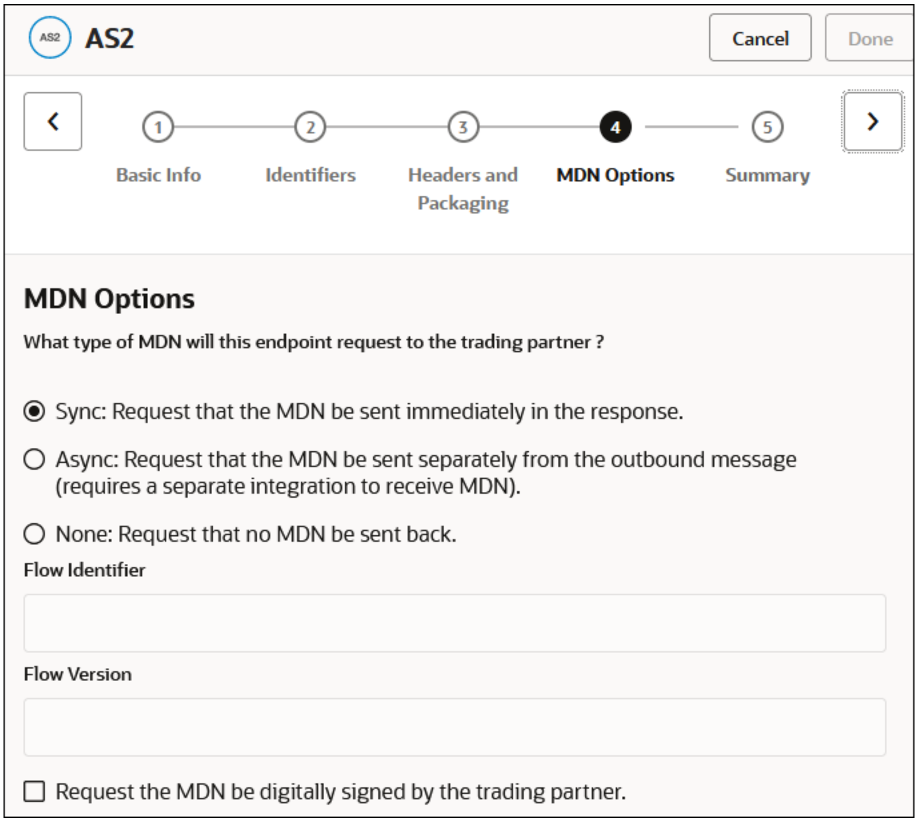
SMIME 3.2 with SHA1-RSA

Note: Private key is required for this option. Please make sure the key is configured in connection.

□ Compress Outbound Message

○ Digitally Sign first, then Compress ○ Compress first, then Digitally Sign

7. On the MDN page, select the type of message delivery network (MDN) for the endpoint to request to the trading partner. For this example, an immediate synchronous response is selected.



MDN Options

What type of MDN will this endpoint request to the trading partner ?

● Sync: Request that the MDN be sent immediately in the response.

○ Async: Request that the MDN be sent separately from the outbound message (requires a separate integration to receive MDN).

○ None: Request that no MDN be sent back.

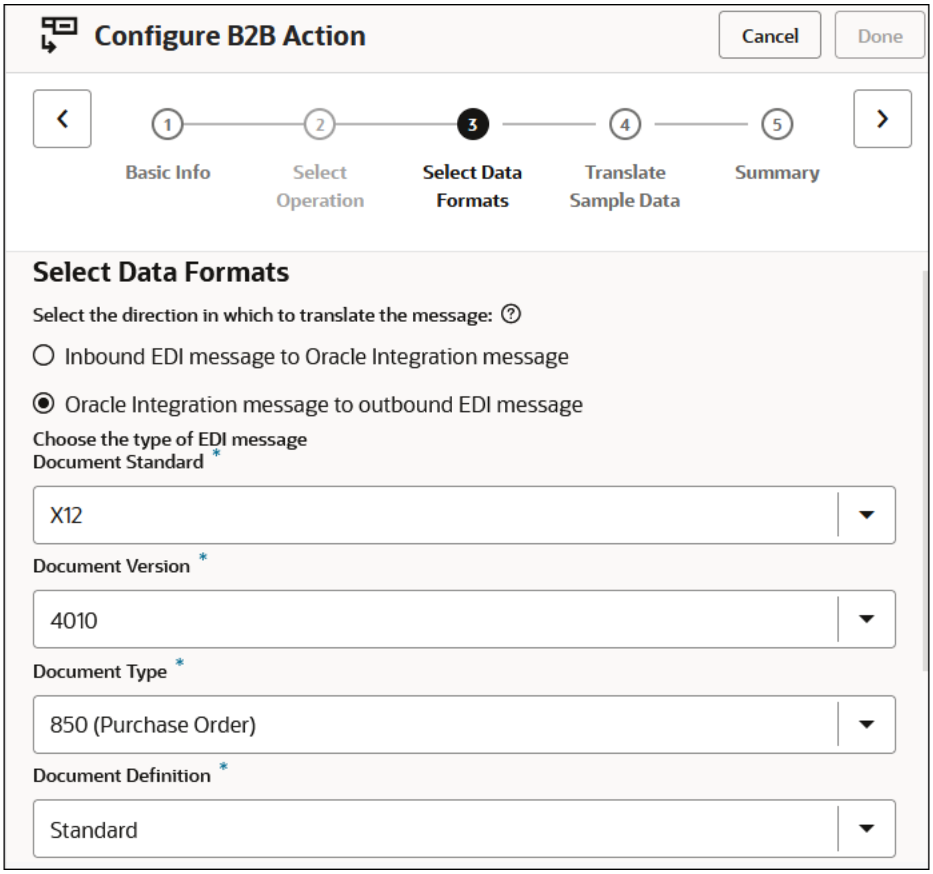
Flow Identifier

Flow Version

□ Request the MDN be digitally signed by the trading partner.

8. On the Summary page, review your selections and click **Done**.
9. In the mapper, perform the necessary mappings between the source and target elements.
10. Drag a **B2B** action into the integration canvas.

    The Configure B2B Action Wizard is displayed.
11. On the Basic Info page, enter a name and select **Standalone mode**.
12. On the Select Data Formats page, select the translation direction and the type of EDI message to send. For this example, an X12 message of document version 4010 and document type 850 is selected.



Select Data Formats

Select the direction in which to translate the message:

○ Inbound EDI message to Oracle Integration message

● Oracle Integration message to outbound EDI message

Choose the type of EDI message

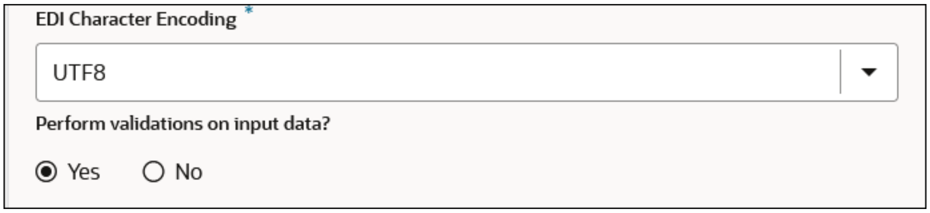
Document Standard *: X12

Document Version *: 4010

Document Type *: 850 (Purchase Order)

Document Definition *: Standard

UTF8 EDI character encoding and input data validation are automatically selected.

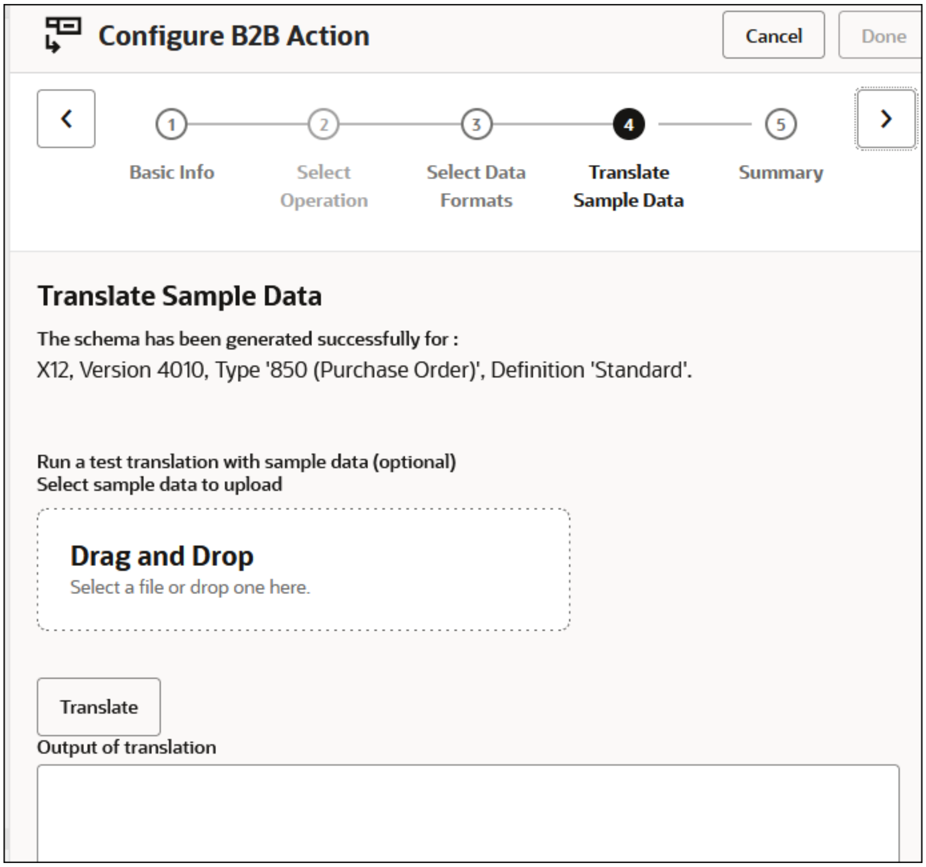


EDI Character Encoding *: UTF8

Perform validations on input data?

● Yes ○ No

13. On the Translate Sample Data page, drag the payload to translate into the wizard and click **Translate** to validate the output payload.



Translate Sample Data

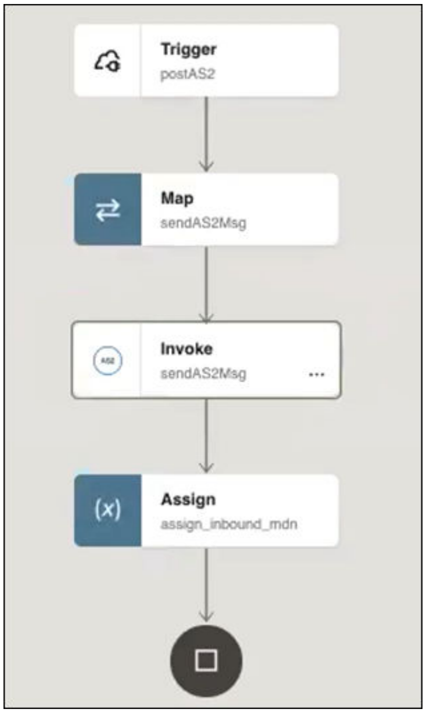
The schema has been generated successfully for :
X12, Version 4010, Type '850 (Purchase Order)', Definition 'Standard'.

Run a test translation with sample data (optional)
Select sample data to upload

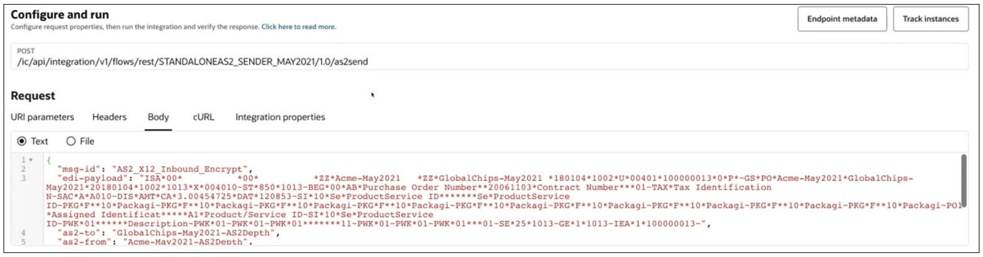
Output of translation

14. Assign business identifiers and save the integration.

    When complete, the integration looks as follows.



15. On the Integrations page, hover your cursor over the integration and select **Actions** ⋯, then **Run**.
16. Click **Body** to view the payload, then click **Run**.



Configure and run

Configure request properties, then run the integration and verify the response. Click here to read more.

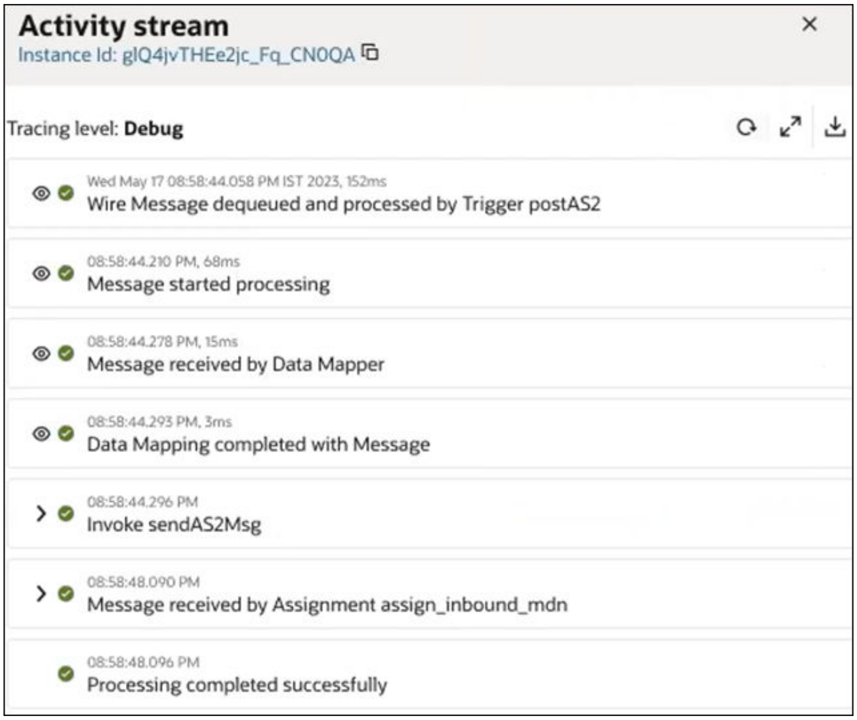
POST
/ic/api/integration/v1/flows/rest/STANDALONEAS2_SENDER_MAY2021/1.0/as2send

Request

● Text ○ File

```
{
  "msg-id": "AS2_X12_Inbound_Encrypt",
  "edi-payload": "ISA*00*          *00*          *ZZ*Acme-May2021   *ZZ*GlobalChips-May2021 *180104*1002*U*00401*100000013*0*P*~GS*PO*Acme-May2021*GlobalChips-
May2021*20180104*1002*1013*X*004010~ST*850*1013~BEG*00*AB*Purchase Order Number**20061103*Contract Number***01~TAX*Tax Identification
N~SAC*A*A010~DIS*AMT*CA*3.00454725*DAT*120853~SI*10*Se*ProductService ID*******Se*ProductService
ID~PKG*F**10*Packagi~PKG*F**10*Packagi~PKG*F**10*Packagi~PKG*F**10*Packagi~PKG*F**10*Packagi~PKG*F**10*Packagi~PKG*F**10*Packagi~PKG*F**10*Packagi~PKG*F**10*Packagi~PO1
*Assigned Identificat*****A1*Product/Service ID~SI*10*Se*ProductService
ID~PWK*01******Description~PWK*01~PWK*01~PWK*01*******11~PWK*01~PWK*01~PWK*01***01~SE*25*1013~GE*1*1013~IEA*1*100000013~",
  "as2-to": "GlobalChips-May2021-AS2Depth",
  "as2-from": "Acme-May2021-AS2Depth",
```

17. View the activity stream when processing completes.



Activity stream

Instance Id: glQ4jvTHEe2jc_Fq_CN0QA

Tracing level: **Debug**

- Wed May 17 08:58:44.058 PM IST 2023, 152ms
  Wire Message dequeued and processed by Trigger postAS2
- 08:58:44.210 PM, 68ms
  Message started processing
- 08:58:44.278 PM, 15ms
  Message received by Data Mapper
- 08:58:44.293 PM, 3ms
  Data Mapping completed with Message
- 08:58:44.296 PM
  Invoke sendAS2Msg
- 08:58:48.090 PM
  Message received by Assignment assign_inbound_mdn
- 08:58:48.096 PM
  Processing completed successfully

Because this is a B2B standalone integration, you view instance status on the Instances page. You cannot view instance status from the Track B2B messages page for B2B standalone integrations.

18. In the navigation pane, click **Observability**, then **Instances**.