TIBCO BusinessConnect™ Container Edition - ebXML Protocol

User Guide - ebMS2 Standard

Version 1.0.0 | October 2023

Contents

Contents 2

Introduction 7

ebXML Architecture 9

Core Components 10

Business Process Specification Schema 10

Collaboration Protocol Profile 10

Collaboration Protocol Agreement 11

ebXML Registry Service 11

ebXML Message Service 11

Features 15

BusinessConnect Container Edition - ebXML Protocol Process Flows and Functionality 17

Public and Private Messages 17

Private Messages and Processes 17

Public Messages and Processes 18

Processing ebXML Transactions 18

Processing a Private Process–Initiated Transaction 18

Processing a Trading Partner–Initiated Transaction 19

Processing Business-Level Signals 20

Delivery Semantics 23

Example of Reliable Delivery with Receipt Acknowledgment 26

Synchronous Reply Modes 28

Illustrations of Synchronous Reply Modes 29

Error Handling 35

Sequenced Messages 35

Processing Outbound Sequenced Messages 36

Processing Inbound Sequenced Messages 37

Duplicate Sequence Numbers 38

Expired Messages 38

Expired Conversations 39

Consistency of Message Settings 39

Preparing to Use BusinessConnect Container Edition - ebXML Protocol 41

Exchanging Information with Your Trading Partners 41

Configuring Your BusinessConnect Container Edition Server 42

Transports 42

Copying Large Files to NFS (Sharing) 42

Domains 44

Managing ebXML Transactions 45

Adding Transactions 45

Notify and Request-Response Transaction Properties 48

Action-Specific Properties 50

Configuring for Delivery Semantics 57

Exporting Transactions 58

Importing Transactions 59

Setting Up Trading Hosts and Partners 60

Configuring ebXML Protocol for Trading Hosts 60

Adding Domain Identities 61

Configuring ebXML Protocol for Trading Partners 62

General Tab 62

Transports Tab 64

Configuring Agreement Protocol Bindings 65

Configuring Business Agreements with ebXML Protocol 65

Configuring Operation Bindings 65

Binding Operations 66

Editing Operation Bindings 66

Operation Settings Tab 67

Action Settings Tab 67

Transports Tab 68

Setting Document Security Properties 69

Configuring Transports 70

Outbound Transports for Host 70

Inbound Transports for Partner 71

Overriding Outbound Transport Settings 72

Overriding Participant Settings 72

Private Processes 73

Configuring Private Processes 73

Standalone Private Processes 73

TIBCO ActiveMatrix BusinessWorks Processes 74

Private Process Message Formats 74

Outbound Request Format 75

Request Acknowledgment Format 81

Inbound Response Format 84

Inbound Request Format 88

Outbound Response Format 91

Response Acknowledgment Format 95

Advisory Signal Message Format 97

Error Message Format 102

Additional Data Classes 105

Request and Response Fields 106

Configuring Private Processes with TIBCO ActiveMatrix BusinessWorks 107

Configuring BCServerConfig Resource 107

Configuring Send and Receive Activities 108

Viewing Logs 111

Transactions Screen 111

Customizing Transactions Screen 111

Sort Transactions 113

Search for Transactions 113

Using Search 114

Transaction Details 115

Tutorial — TIBCO ActiveMatrix BusinessWorks Private Processes 116

Prerequisites 116

Setting Up the Tutorial 117

Configuring Private Processes 117

Running the Tutorial 118

Creating a Process Invoker 118

Send Request 119

Send Response 119

Running Send Response on Machine 2 120

Running Send Request on Machine 1 121

Expected Results 122

Status Codes 123

Property Reference 135

Smart Routing 138

Notes on Sequenced Messages 141

Unsequenced Messages in Active Conversations 141

Reusing Sequence Numbers 141

Public Messages 143

ebXML Public Message Structure 143

SOAP Envelope 144

MessageHeader Element 145

AckRequested Element 149

SyncReply Element 150

Manifest Element 151

MessageOrder Element 152

Business-Level Signals 154

Receipt Acknowledgment Signals 154

Exception Signals 157

TIBCO Documentation and Support Services 161

Legal and Third-Party Notices 163

Introduction

This section provides an overview of TIBCO BusinessConnect™ Container Edition - ebXML Protocol and lists the product features.

> **Note:** This document focuses on the ebMS2 standard of the ebXML protocol. The ebMS2 standard of the protocol is also referenced as ebXML hereinafter.

ebXML (electronic business XML) is a business protocol used by buyers, sellers, and intermediaries to share securely, XML or non-XML business documents and messages reliably over the internet. The ebXML suite of specifications offers companies a standard method for exchanging business messages, conducting trading relationships, communicating data in common terms, and defining and registering business processes. These specifications include:

- ebXML Technical Architecture
- ebXML Business Process Specification Schema (BPSS)
- ebXML Collaboration Protocol Profile and Agreement (CPPA)
- ebXML Message Service (ebMS)

Go to http://www.ebxml.org/specs to access all ebXML specifications and other documentations.

ebXML Architecture

When two companies conduct business by using ebXML protocol, they go through the following sequence of events.

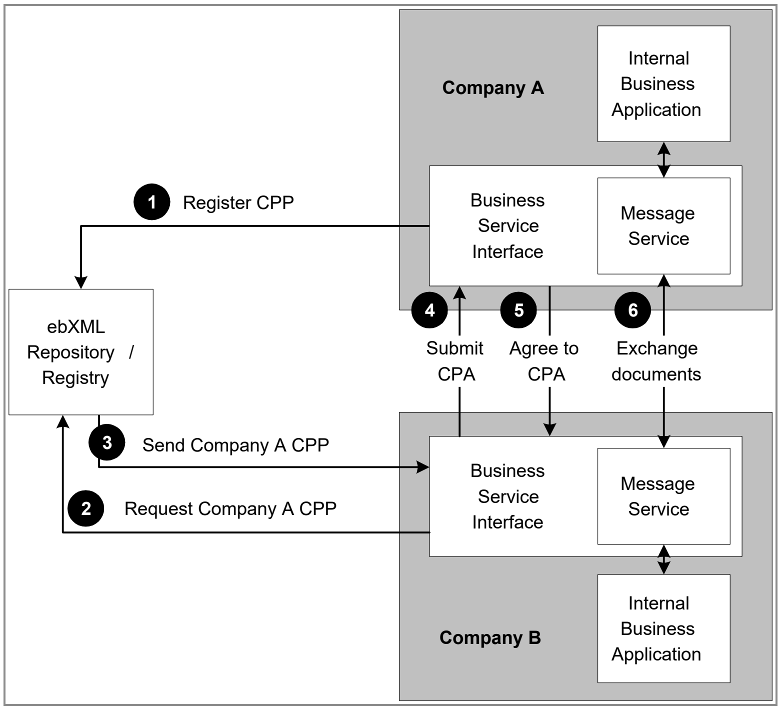
1. Company A registers its Collaboration Protocol Profile (CPP) with an ebXML registry or repository.
2. Company B requests CPP of Company A from the ebXML registry.
3. The ebXML registry sends the requested CPP to Company B.
4. Company B uses the CPP of Company A to formulate a Collaboration Protocol

Agreement (CPA) and submits it to Company A.

5. Company A agrees to the CPA.
6. The two companies start conducting ebXML transactions by exchanging business documents with their implementations of the ebXML Message Service Handler.

For more information on the ebXML architecture and how the ebXML-defined components work together, see ebXML Architecture, or ebXML Technical Architecture Specification.

Figure 1: ebXML Architecture



BusinessConnect™ Container Edition - ebXML Protocol Architecture

BusinessConnect Container Edition - ebXML Protocol is the TIBCO implementation of the ebXML Message Service, which defined by ebMS 2.0, see step 6 in ebXML Architecture. As a plug-in to TIBCO BusinessConnect™ Container Edition, you can use BusinessConnect Container Edition - ebXML Protocol to handle one-to-one request-response and notification transactions defined by BPSS 1.05. In addition, when not strictly adhering to the other ebXML specifications, BusinessConnect Container Edition - ebXML Protocol embraces many concepts defined in BPSS 1.05 and CPPA 2.0.

BusinessConnect Container Edition - ebXML Protocol offers advanced message processing such as:

- ebXML conversations
- Sequenced transactions
- Delivery semantics
- Duplicate detection
- Synchronous reply modes
- Time-to-live
- Error handling
- Validation of business documents in the message payload

You can also use its security functions to carry out transactions that require authentication, authorization, integrity, privacy, and non-repudiation.

See Features for a detailed list of the features of BusinessConnect Container Edition - ebXML Protocol.

ebXML Architecture

The following components defined by ebXML are key to understanding and using ebXML:

- Core components. These are business documents that a trading partner would exchange with a partner. See Core Components.
- Business Process Specification Schema (BPSS). A company uses a BPSS to define its overall business processes. See Business Process Specification Schema.
- Collaboration Protocol Profile (CPP). A company uses a CPP to define how it can do business electronically. See Collaboration Protocol Profile.
- Collaboration Protocol Agreement (CPA). Two companies create this from their merged CPPs to define how they can do business electronically together. See Collaboration Protocol Agreement.
- Registry. A file-based or database repository where partners share public documents. See ebXML Registry Service.
- ebXML Message Service. This defines message formats. See ebXML Message Service.

Core Components

The core components define a set of business documents that a trading partner would exchange with a partner. Example: Purchase Order.

These can be stored in the registry when complete.

Business Process Specification Schema

The BPSS is an XML-based schema definition that describes the business process and information models of a company. BusinessConnect Container Edition - ebXML Protocol uses the BPSS 1.05 specification.

Instances of a BPSS can be stored in the registry when complete.

Collaboration Protocol Profile

The CPP is an XML-based definition that describes how a company does business electronically.

The CPP contains the following details:

- Party information
    - Party name
    - Contact info
- Transport Endpoints
- Security Parameters
- Messaging Properties
- Link to a Business Parameters Process Specification Document
- Time-out and retry
- Packaging information, and other information.

Instances of this schema can be stored in the registry when complete.

Collaboration Protocol Agreement

The CPA is an XML-based definition that specifies how two companies have agreed to do business electronically. A CPA document is based on the CPPs of two companies.

Instances of this schema can be stored in the registry when complete.

ebXML Registry Service

The ebXML Registry Service is a public repository for the following documents:

- Core components. See Core Components.
- BPSS. See Business Process Specification Schema.
- CPP. See Collaboration Protocol Profile.
- CPA. See Collaboration Protocol Agreement.

Trading partners use the registry to upload and download BPSS, CPP, CPA, and core component documents.

ebXML Message Service

The ebMS specification is a component that defines the message packaging, routing, and transport facilities for the ebXML infrastructure. ebMS is defined as an abstraction of a process, not a physical component. It is a definition for an ebXML Message Service Handler (MSH). An MSH is the implementation of the ebMS specification.

ebMS provides ebXML trading partners a standard and secure means of exchanging business messages. This service also prescribes formats for all messages between distributed ebXML components, including registry mechanisms and compliant user applications.

ebMS is defined as a set of layered extensions to the base Simple Object Access Protocol (SOAP) and SOAP messages with attachments specifications. The ebMS specification is available at http://www.ebxml.org/specs

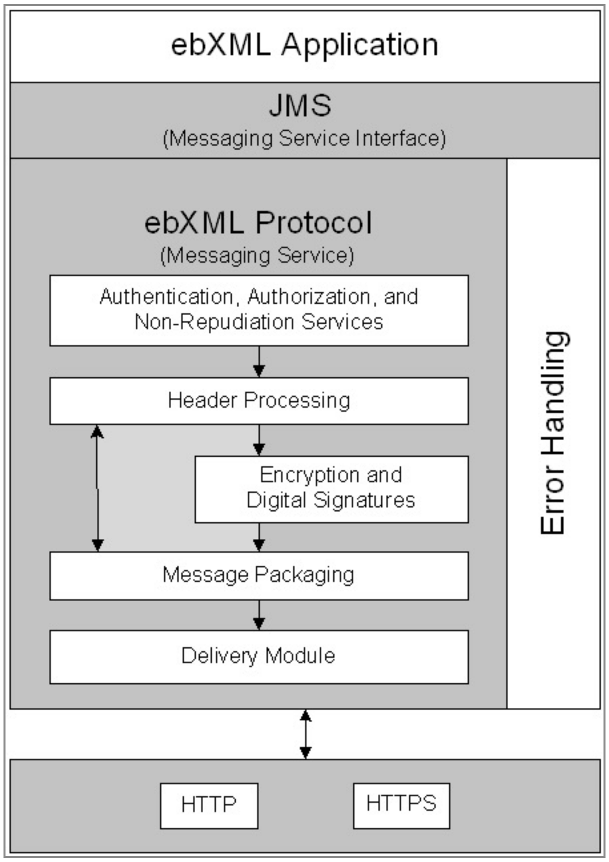
The XSD is available at http://www.oasis-open.org/committees/ebxml-msg/schema/msg-header-2_0.xsd

ebMS can be divided into the following conceptual parts:

- Message service interface. See Messaging Service Interface.
- Message service layer. See ebXML Message Service Layer.
- Transport services. See Transport Services.

The following figure is a graphical representation of the BusinessConnect Container Edition - ebXML Protocol implementation of ebMS components.

Figure 2: ebMS Components in BusinessConnect Container Edition - ebXML Protocol



Messaging Service Interface

This is an abstract service interface that ebXML-compliant applications use to interact with the message service to send and receive messages. The message service also uses this to interface with applications handling received messages.

Applications use this abstract service interface to interact with the MSH to send and receive messages. The MSH uses the Message Delivery module in this abstract service interface to interface with applications handling received messages.

ebMS provides ebXML with an abstract interface whose functions, at an abstract level, include the following information:

- **Send**: Sends an ebXML message. Values for the parameters are derived from the ebXML message headers.
- **Receive**: Receives an ebXML message.
- **Notify**: Provides notification of expected and unexpected events.
- **Inquire**: Provides a method of querying the status of the particular ebXML message interchange.

For BusinessConnect Container Edition - ebXML Protocol, this service interface is the Java Message Service (JMS). See Configuring Private Processes for a description of this abstract interface.

ebXML Message Service Layer

The message service layer maps the abstract interface to the underlying transport services. This includes the following functions:

- Header processing. See Header Processing.
- Encryption and digital signatures. See Encryption and Digital Signatures.
- Message packaging. See Message Packaging.
- Message delivery. See Message Delivery Module.
- Transport mapping. See Message Delivery Module.
- Error handling. See Error Handling.

Header Processing

The message header is necessary for routing and delivery.

For outbound messages, when creating the ebXML header elements, the message service layer uses input from the application, information from the CPA, and generated information such as digital signature, timestamps, and unique identifiers.

For inbound messages, this module includes the following duties:

- Extracting digital signatures
- Checking routing information such as from and to information

- Performing basic validation
- Extracting service action values

See ebXML Public Message Structure for information on how the ebXML header fits into the overall message structure.

Encryption and Digital Signatures

The MSH performs all security-related functions, including the following functions:

- Identification
- Authentication (verification of identity)
- Authorization (access controls)
- Integrity (message signing)
- Privacy
- Non-repudiation
- Logging

Note: In addition to implementing the other security and messaging functions defined in the ebMS specification, BusinessConnect Container Edition - ebXML Protocol supports payload encryption.

Message Packaging

The ebXML message contains ebXML header elements and might contain one or more payloads. In ebXML, the payload can take any digital form, including XML, ASC X12, HL7, AIAG E5, database tables, and binary image files.

For outbound messages, this is the final enveloping of an ebXML message into its SOAP messages with attachments container before it is sent as a public message.

For inbound messages, the MSH separates the message into its components and determines how many payloads are in the message.

See ebXML Public Message Structure.

Message Delivery Module

This module handles message delivery-related properties, and includes the following duties:

- Reliable messaging
- Duplicate detection
- Synchronous reply elements
- Sequence numbers
- Time to live
- Send and receive
- Transport mapping and binding

See ebXML Public Message Structure for information on how the transport protocol envelope fits into the overall message structure.

Error Handling

This component handles the reporting of errors encountered during message processing.

Transport Services

This part of the MSH uses transport protocols, which deliver public messages.

For the BusinessConnect Container Edition - ebXML Protocol implementation of ebXML, the transport protocols available are HTTP, HTTPS, and HTTPSCA.

Features

The following are the major features of BusinessConnect Container Edition - ebXML Protocol:

- Conforms to ebMS 2.0.
- Request-response and notification transactions specified by ebXML BPSS 1.05, along with the following business-level signals:

    - Receipt Acknowledgment
    - Receipt Exception
    - Acceptance Exception
    - General Exception
- Message Service Handler with all delivery semantics.
- Synchronous reply modes specified by ebXML CPPA 2.0: none, mshSignalsOnly, responseOnly.
- ebXML Message Ping and StatusRequest services.
- Payload encryption by using S/MIME with a user-selectable algorithm.
- Payload encryption by using XML encryption.
- SOAP and MIME public message packaging.
- Message and payload schema validation by using DTDs and XSDs.
- Advisory signals to private processes for inbound MSH acknowledgments and business-level signals.
- Sending and receiving attachments.
- HTTP and HTTPS (also with client authentication) transport protocols.
- UTF-8 encoding.
- Interoperability. This version has undergone interoperability certification with the Drummond Group.
- Support for integration with TIBCO ActiveMatrix BusinessWorks.
- The ability to import business document XSDs that themselves import other XSDs.
- The ability to specify certain timing constraints, such as how long to wait for a response from a private process.

BusinessConnect Container Edition - ebXML Protocol Process Flows and Functionality

This section describes process flows and relevant functionality when using BusinessConnect Container Edition - ebXML Protocol for business transactions.

Public and Private Messages

In an ebXML transaction, two partners exchange business documents over the internet based on the pre-defined rules of ebXML. ebXML specifies what message formats and transport protocols that the partners have agreed to use, among other options. For more information on the partner agreement, see *TIBCO BusinessConnect™ Container Edition Trading Partner Management*.

The exchange of business documents is known as the *process flow*. In any process flow, two types of messages are exchanged:

- Private messages
- Public messages

Private Messages and Processes

Private messages are exchanged between private processes and BusinessConnect Container Edition - ebXML Protocol. Private messages carry business documents, including request, response, or notification documents, or alternatively, advisory and error signals. For a detailed description of BusinessConnect Container Edition - ebXML Protocol private messages, see Private Process Message Formats.

Private processes handle conversion from internal data to and from the formats defined acceptable to the public process.

- **Outbound** The private processes convert internal data into business documents in the formats acceptable to the public process.

- **Inbound** The private processes receive business documents in the formats defined by the public processes, convert them to internal company formats, and process the data or forward to other internal resources for processing.

See Private Processes for more information on private processes.

Public Messages and Processes

BusinessConnect Container Edition - ebXML Protocol acts as the public process and exchanges documents with a trading partner over the internet by using the message formats and protocols defined by the ebXML Message Service specification. For more information on public processes, see Public Messages.

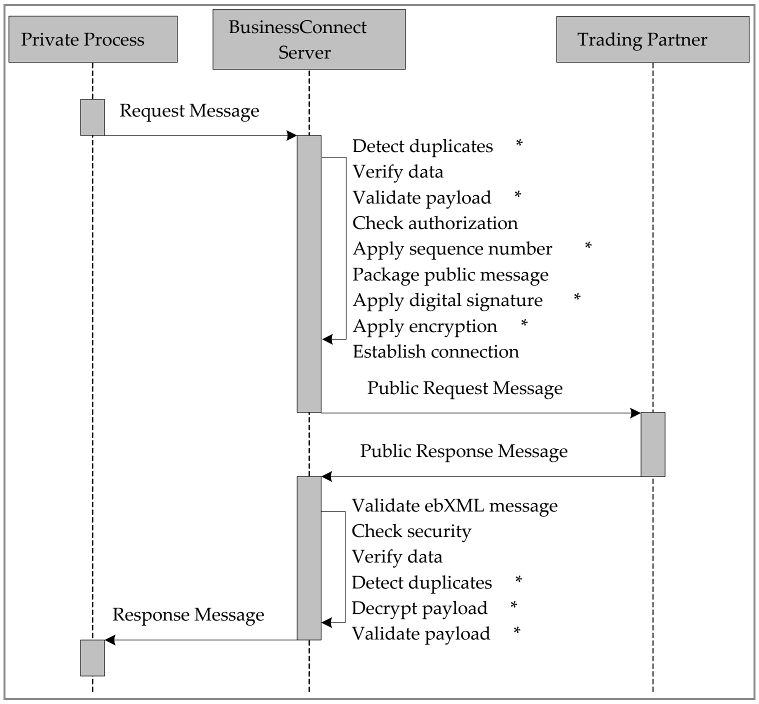
Processing ebXML Transactions

This section describes how BusinessConnect Container Edition - ebXML Protocol processes simple transactions.

Processing a Private Process–Initiated Transaction

The following figure shows how BusinessConnect Container Edition - ebXML Protocol processes a private process–initiated transaction. The transaction depicted is an asynchronous request-response transaction with best-effort delivery semantics. See Delivery Semantics. In the case of a notification transaction, the processing ends when BusinessConnect Container Edition - ebXML Protocol sends out the public request message.

Figure 3: Processing a Private Process–Initiated Transactions

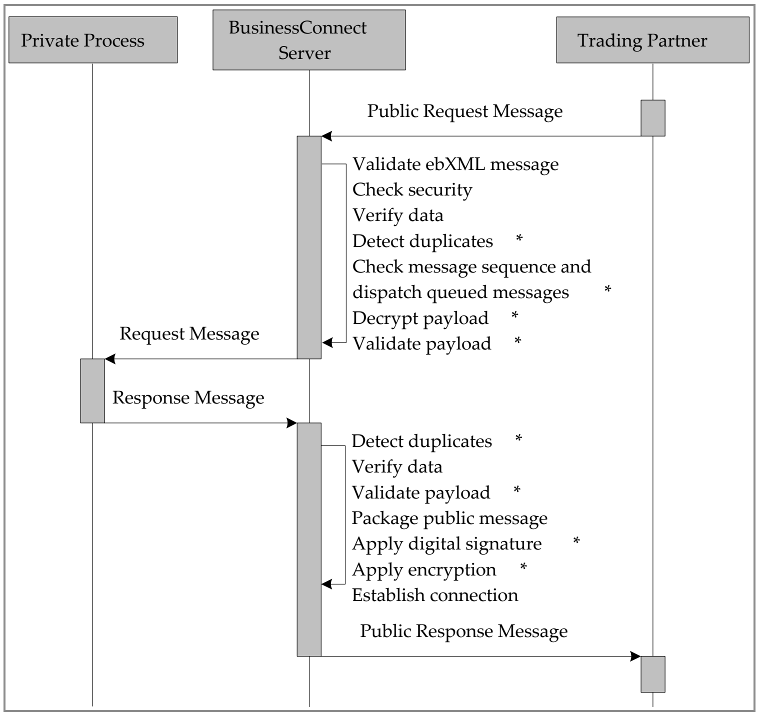


Note that items marked with an asterisk (*) are run conditionally, depending on field settings in the private message or how you configure a particular transaction. See Managing ebXML Transactions for information on configuring transactions and Private Processes for information on using private messages.

Processing a Trading Partner–Initiated Transaction

The following figure shows how BusinessConnect Container Edition - ebXML Protocol processes a trading partner–initiated transaction. The transaction depicted is an asynchronous request-response transaction with best-effort delivery semantics. See Delivery Semantics. In the case of a notification transaction, the processing ends when BusinessConnect Container Edition - ebXML Protocol forwards the request message to the private process.

Figure 4: Processing a Trading Partner–Initiated Transactions



Note that items marked with an asterisk (*) are run conditionally, depending on field settings in the private message or how you configure a particular transaction. See Managing ebXML Transactions for information on configuring transactions and Private Processes for information on using private messages.

Processing Business-Level Signals

This section describes how BusinessConnect Container Edition - ebXML Protocol processes business-level signals. Two types of business-level signals are: exception messages and receipt acknowledgments.

See Error Handling for information on MSH error messages.

Exception Messages

When BusinessConnect Container Edition - ebXML Protocol encounters exceptions, either from the private process (see Outbound Response Format) or when processing transactions, it sends out an *exception* message to the trading partner. The exception message is simply a response message that contains an exception as the payload.

Outbound Exception Messages

BusinessConnect Container Edition - ebXML Protocol sends exception messages to the trading partner for the following exceptions:

- **Receipt Exception** BusinessConnect Container Edition - ebXML Protocol sends a ReceiptException message when it fails to validate the inbound payload documents.
- **Acceptance Exception** When the private process generates this exception (indicating that the message is not accepted), BusinessConnect Container Edition - ebXML Protocol forwards an AcceptanceException message to the trading partner.
- **General Exception** When the private process generates this exception in the event that an exception occurs after the message is accepted, BusinessConnect Container Edition - ebXML Protocol forwards a GeneralException message to the trading partner.

See Exception Signals for the format of the exception message.

Inbound Exception Messages

If BusinessConnect Container Edition - ebXML Protocol receives an exception message from the trading partner, it also updates the private process by publishing the following private messages:

- **In case of trading partner exception while processing request message:** An `ae/ebXML/InitiatorResponse` message that packages the entire exception payload (for example, `<AcceptanceException>`) in the **response** field. See Inbound Response Format.
- **In case of trading partner *receipt* exception while processing response message:** If an `ae/ebXML/ResponderAck` message has not been sent to the private process, an `ae/ebXML/ResponderAck` message with an error code in the **statusCode** field. See Response Acknowledgment Format.
- An `ae/ebXML/Advisory` message on the advisory signal subject. See Advisory Signal Message Format.
- An `ae/ebXML/Advisory` message on the error subject. See Error Message Format.

Note: The `ae/ebXML/Advisory` message is sent on both the error and advisory signal subjects for inbound business-level exceptions, whereas it is sent only on the error subject for inbound MSH errors. See Error Handling.

Receipt Acknowledgments

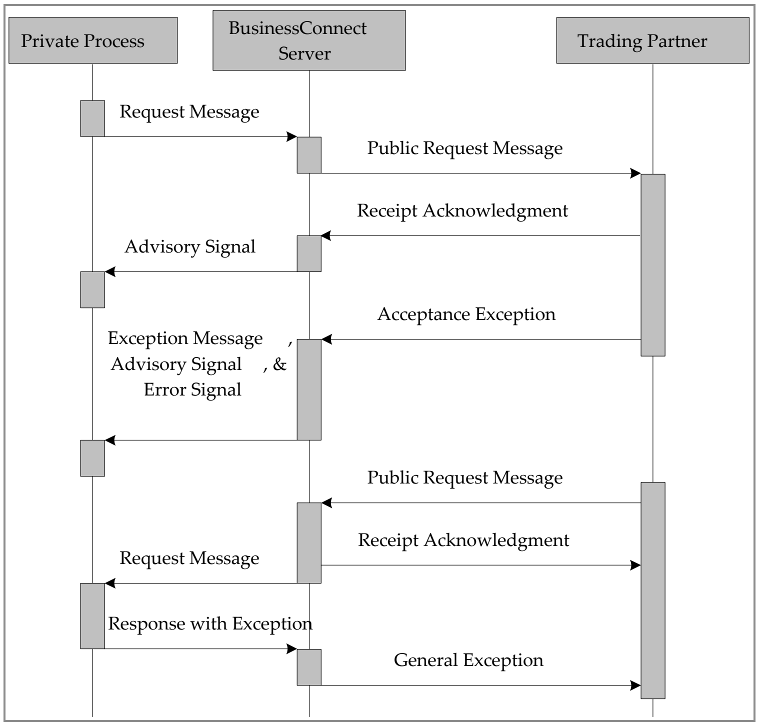
You can also utilize the ability of BusinessConnect Container Edition - ebXML Protocol to process *receipt acknowledgment* messages to be updated on the progress of a transaction. The receipt acknowledgment message is used to inform a message sender that the message has been received and acknowledged. Note that in the processing flow of BusinessConnect Container Edition - ebXML Protocol, it sends out the receipt acknowledgment message after it successfully validates the business document in the message payload.

You can configure BusinessConnect Container Edition - ebXML Protocol to send or receive receipt acknowledgments for a request action or response action of a transaction, or both.

Note: The exception message is *not* eligible for a receipt acknowledgment, as it is also a business-level signal.

See the following figure for an example of how BusinessConnect Container Edition - ebXML Protocol processes exception messages and receipt acknowledgment messages. In this example, the transaction is configured to require receipt acknowledgments for both the request action and the response action. Note that by requiring receipt acknowledgments, the private process receives additional messages on the advisory signal subject, and is updated on the receipt of the request and response messages by the trading partner.

Figure 5: Processing Business-level Signals



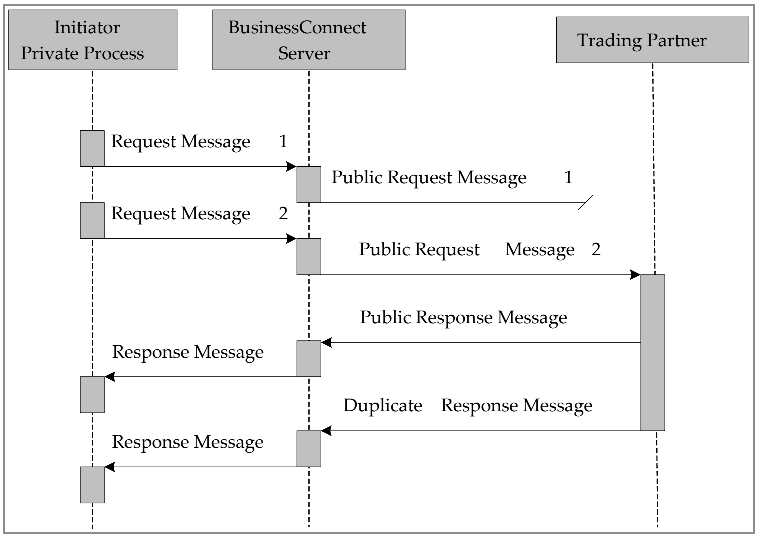
Delivery Semantics

With BusinessConnect Container Edition - ebXML Protocol, you can customize the level of reliability in message delivery by requiring *MSH acknowledgment* messages. Unlike receipt acknowledgments, MSH acknowledgments confirm the successful delivery of messages, not the receipt of messages on the business level.

See Configuring for Delivery Semantics for information on how to configure each of the following delivery semantics:

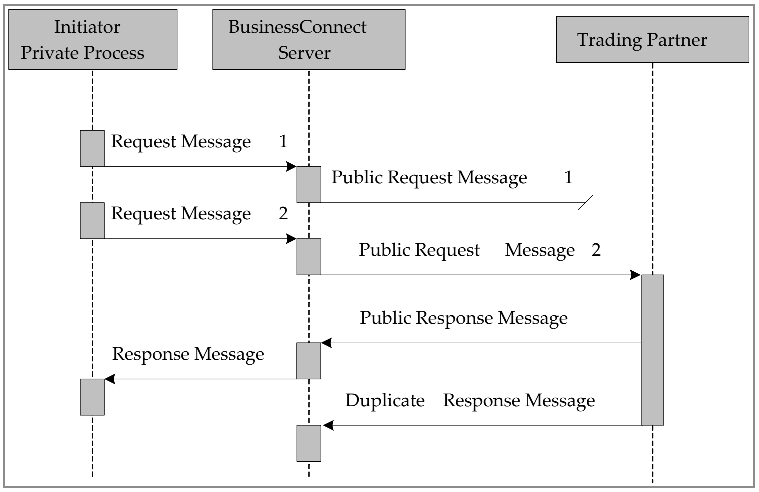
- **Best Effort** Deliver messages as they are received. Duplicate messages are not eliminated, and no MSH acknowledgment is used to ensure that messages are delivered. In the following figure, BusinessConnect Container Edition - ebXML Protocol does not resend the first request message even though it is not received by the trading partner, and does not check that the second response message is a duplicate, sending the message to the private process twice.

Figure 6: Best Effort Delivery Semantics



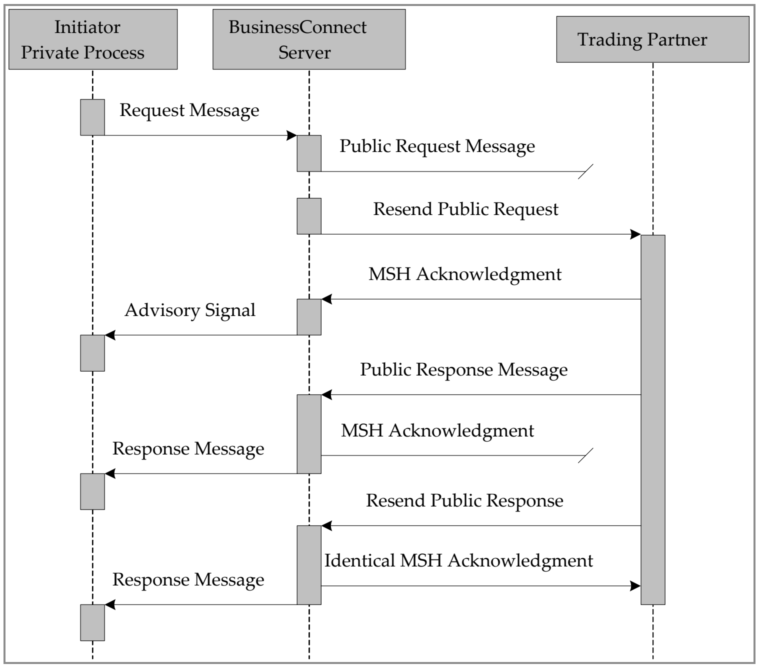
- **At-Most-Once** Deliver each message no more than once. Duplicate messages are eliminated, and no MSH acknowledgment is used to ensure that messages are delivered. In the following figure, BusinessConnect Container Edition - ebXML Protocol does not resend the first request message even though it is not received by the trading partner, and checks that the second response message is a duplicate to ensure that the message is delivered no more than once.

Figure 7: At-Most-Once Delivery Semantics



- **At-Least-Once (ALO)** Deliver each message at least once. Duplicate messages are not eliminated, and the MSH acknowledgment is required, making it possible for a message to be resent until it is acknowledged. This means that the private process can receive duplicate messages. See the following figure for an example of at-least-once delivery semantics.

Figure 8: At-Least-Once Delivery Semantics

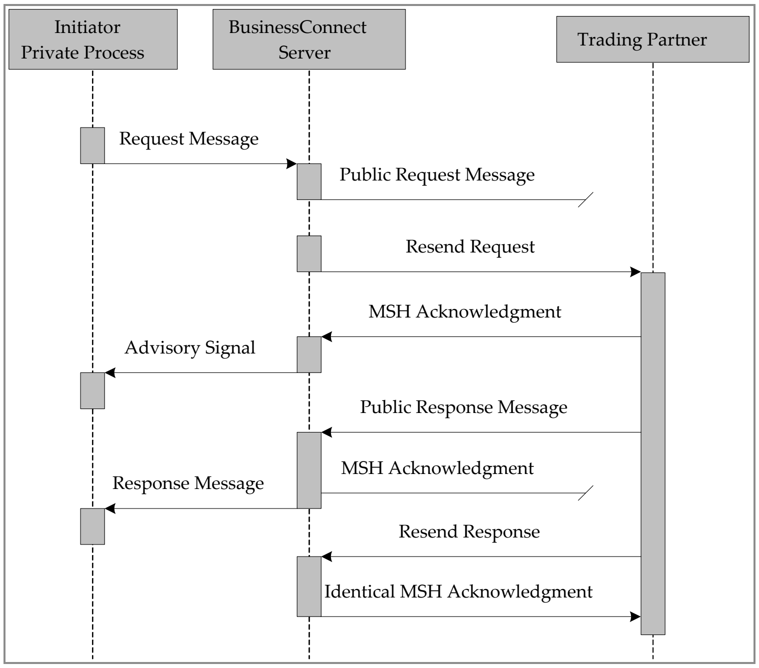


- **Once-And-Only-Once** Deliver each message exactly once. Duplicate messages are eliminated to ensure that the message is delivered not more than once, and the MSH acknowledgment is required to ensure that the message is delivered at least once.

  If BusinessConnect Container Edition - ebXML Protocol does not receive an MSH acknowledgment for a public message, it resends the message for a specified number of times. Also, if BusinessConnect Container Edition - ebXML Protocol receives a duplicate public message, it does not forward the duplicate message to the private process, but simply replies with an exact copy of the original MSH acknowledgment (saved in persistent storage), which it used when replying to the first public message.

  See the following figure for an example of once-and-only-once semantics. Notice that BusinessConnect Container Edition - ebXML Protocol does not send the duplicate response message to the private process, but replies with the MSH acknowledgment that is saved in persistent storage.

Figure 9: Once-And-Only-Once Delivery Semantics



Example of Reliable Delivery with Receipt Acknowledgment

The receipt acknowledgement can be used along with the MSH acknowledgment to let the sender confirm that its request was received correctly. Furthermore, the MSH acknowledgement is used for acknowledge the arrival of the receipt acknowledgement. This example illustrates the complex message exchanges when these two signal messages are used together.

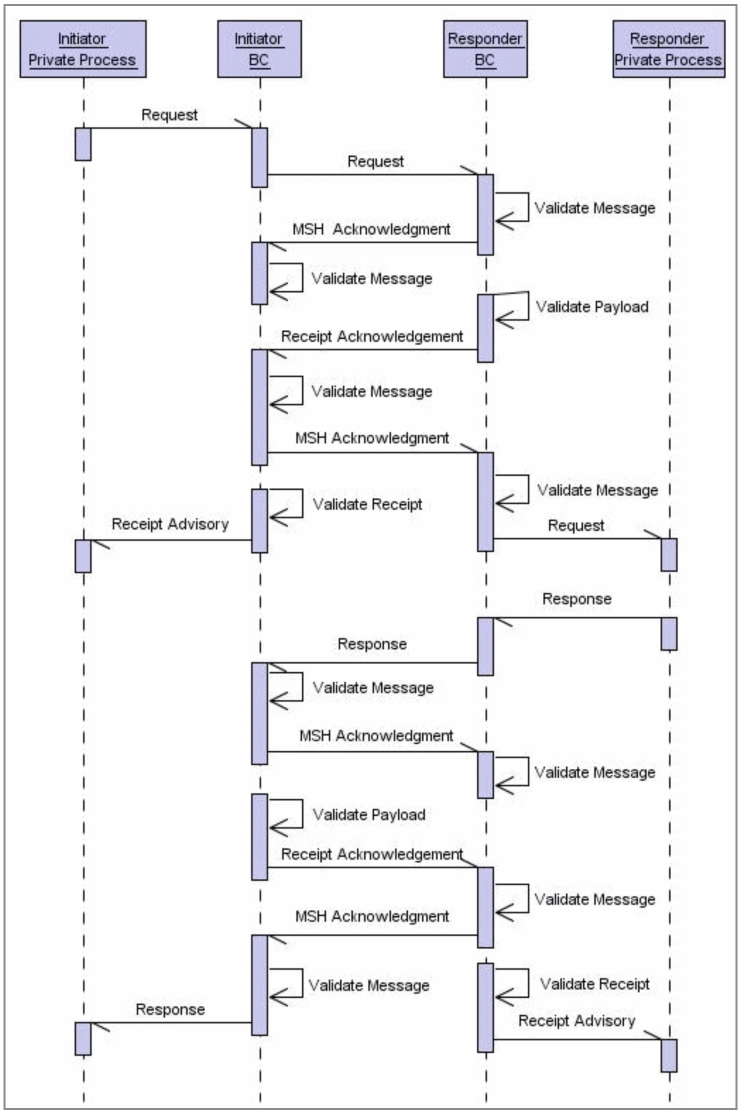
The MSH acknowledgment tells the sender that the recipient has received the message and validated that it conforms to the ebXML schema definition, when the business-level receipt acknowledgment indicates to the sender that the recipient has received the business document, validated that it conforms to the business schema definition, and acknowledged its receipt.

ebXML defines another business-level acknowledgment message called acceptance acknowledgment. In this signal message, the application of the recipient is required to process the business document at the business level. After the application of the recipient accepts the message at the business level, the application of the recipient sends a message indicating that the message is accepted for processing. Currently, BusinessConnect

Container Edition - ebXML Protocol does not implement the acceptance acknowledgment message.

The following figure depicts the message exchanges and actions taken on both sides. For simplicity, we assume that BusinessConnect Container Edition - ebXML Protocol is used on both sides.

Figure 10: Reliable Delivery with Receipt Acknowledgment

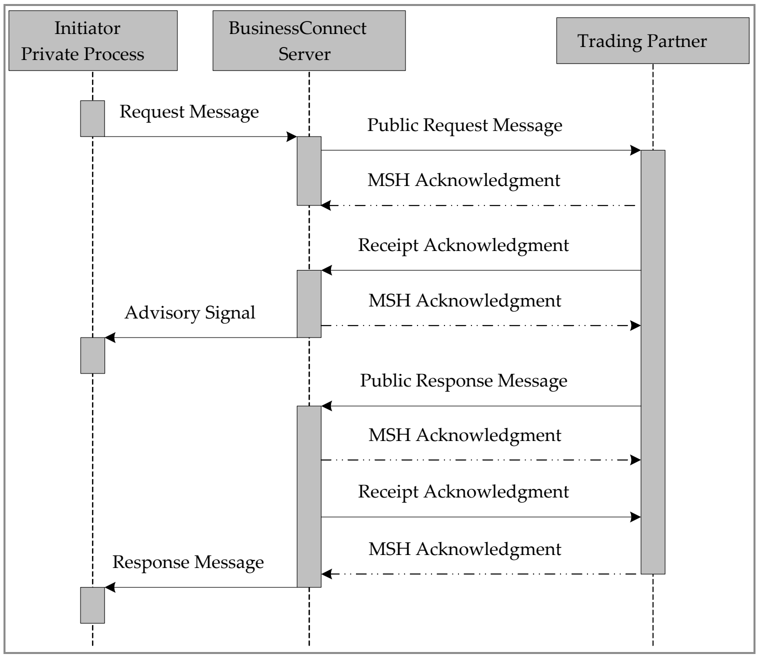


Synchronous Reply Modes

You can use BusinessConnect Container Edition - ebXML Protocol to define three synchronous reply modes for ebXML transactions: none, mshSignalsOnly, and responseOnly. These modes are defined by ebXML Collaboration Protocol Profile and Agreement 2.0. You can select one of them in the Synchronous Reply Mode property in the Operations Editor panel. See Action-Specific Properties for information on the property.

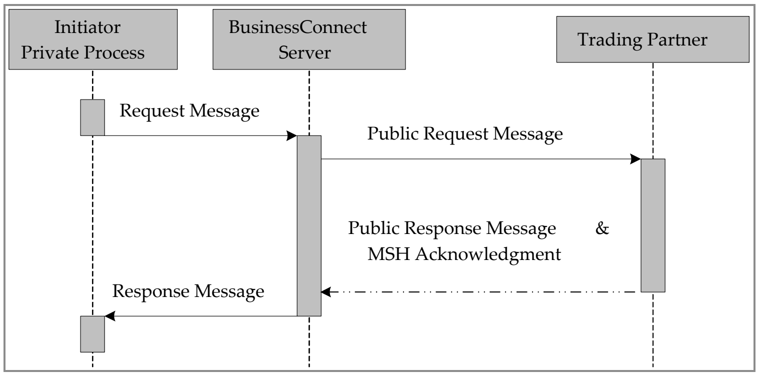
- **none** Send all messages in a transaction asynchronously. See previous sections for examples.
- **mshSignalsOnly** Send MSH acknowledgments (if configured) in synchronous mode. All other messages and signals are sent asynchronously. See the following figure for an example of this mode.

Figure 11: mshSignalsOnly Synchronous Reply



- **responseOnly** Send the MSH acknowledgment (if configured) and response message synchronously *in the same message*. In this mode, you cannot use the receipt acknowledgment. See the following figure for an example of this mode.

Figure 12: responseOnly Synchronous Reply



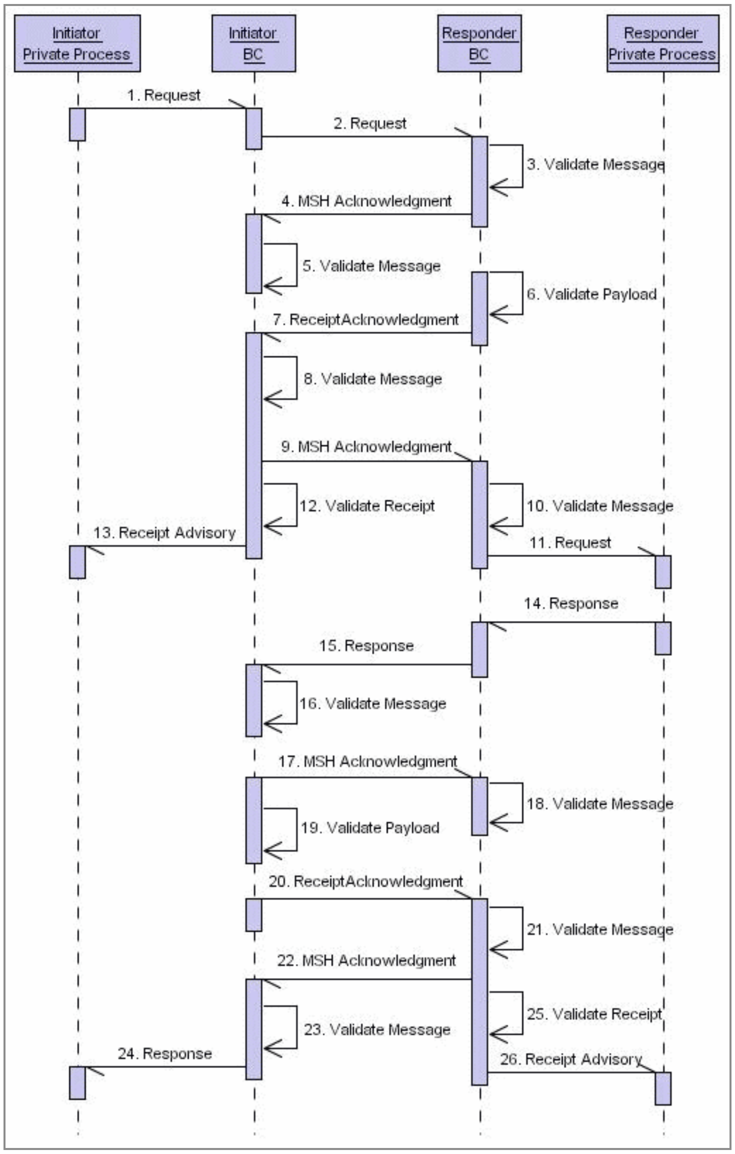
Illustrations of Synchronous Reply Modes

This section describes some examples of the action sequences and message exchanges in different synchronous reply modes. In these examples we assume that both the initiating side and responder side use BusinessConnect Container Edition - ebXML Protocol, however it would be similar if other ebXML products are used.

Example 1: Synchronous Reply Mode — none

In this example, the MSH acknowledgment and receipt acknowledgment are required for both the request and response actions. As described earlier, all messages and signals are sent asynchronously.

Figure 13: Example 1: Synchronous Reply Mode — none



The following are the action sequence on both sides and the messages that are exchanged:

1. A private process in the initiating company sends a request to the local BusinessConnect Container Edition server.
2. The initiator BusinessConnect Container Edition server encapsulates the business request into an ebXML request message and sends to the responder BusinessConnect Container Edition server. This message is sent asynchronously, meaning that all the possible response and signal messages come back from separate channels.
3. The responder BusinessConnect Container Edition server validates the message.

4. The responder BusinessConnect Container Edition server sends the MSH acknowledgment message back to the initiator BusinessConnect Container Edition server if the validation is successful. Otherwise, it sends back an ebXML ErrorList message (MSH error).
5. The initiator BusinessConnect Container Edition server, on receiving the MSH acknowledgment message, validates it and logs it for audit purposes.
6. The responder BusinessConnect Container Edition server, after sending an MSH acknowledgment message, continues to validate the inbound business document payload that is part of the request message.
7. The responder BusinessConnect Container Edition server then sends a receipt acknowledgment message back to the initiator BusinessConnect Container Edition server asynchronously if the validation is successful. Otherwise, it sends back a receipt exception signal that is a business-level exception.
8. The initiator BusinessConnect Container Edition server, on receiving a receipt acknowledgment message, validates it and logs it for audit purposes.
9. The initiator BusinessConnect Container Edition server then sends an MSH acknowledgment message back to the responder BusinessConnect Container Edition server if the validation is successful. Otherwise, it sends back an ebXML ErrorList message.
10. The responder BusinessConnect Container Edition server, after receiving an MSH acknowledgment message, validates it and logs it for audit purposes.
11. The responder then forwards the payload business document in the request message to the local private process and waits for the local private process to respond.
12. The initiator BusinessConnect Container Edition server, after sending an MSH acknowledgment message, validates the actual receipt acknowledgment signal, which is the payload part of the receipt acknowledgment message.
13. The initiator BusinessConnect Container Edition server then publishes an advisory message to its private process indicating that the request has been accepted and validated by the responder side.
14. The responder private process responds to its BusinessConnect Container Edition server.
15. The responder BusinessConnect Container Edition server forwards the business response from the local private process to the initiator BusinessConnect Container

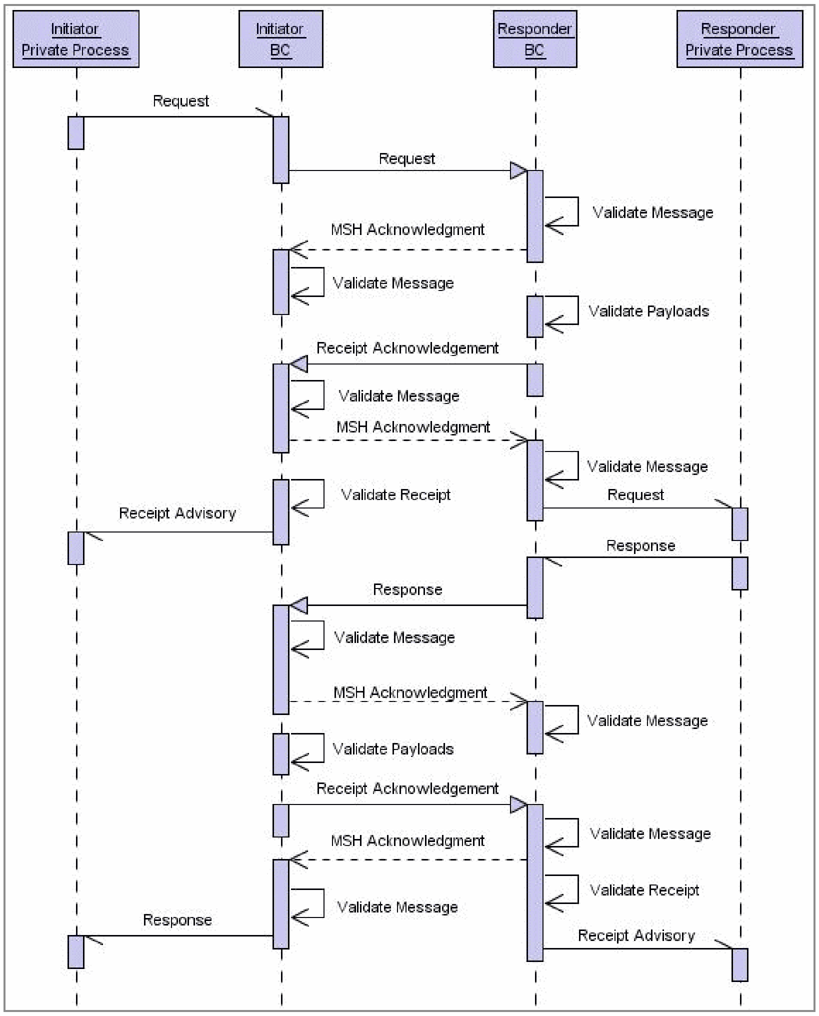
Edition server. This message is sent asynchronously.

16. The initiator BusinessConnect Container Edition server validates the response ebXML message.
17. The initiator BusinessConnect Container Edition server sends the MSH acknowledgment message back to the responder asynchronously if the validation is successful. Otherwise, it sends back an ebXML ErrorList message. The message is sent asynchronously.
18. The responder BusinessConnect Container Edition server, on receiving the MSH acknowledgment message, validates it and logs it for audit purposes.
19. The initiator BusinessConnect Container Edition server, after sending an MSH acknowledgment message, continues to validate the payload business document in the inbound response message.
20. The initiator BusinessConnect Container Edition server sends a receipt acknowledgment message back to the responder BusinessConnect Container Edition server if the validation is successful. Otherwise, it sends back a receipt exception signal. The signal is sent asynchronously.
21. The responder BusinessConnect Container Edition server, on receiving the receipt acknowledgment message, validates it and logs it for audit purposes.
22. The responder BusinessConnect Container Edition server sends an MSH acknowledgment message back to the initiator BusinessConnect Container Edition server if the validation is successful. Otherwise, it sends back an ebXML ErrorList message.
23. The initiator BusinessConnect Container Edition server, on receiving the MSH acknowledgment message, validates it and logs it for audit purposes.
24. The initiator BusinessConnect Container Edition server then forwards the business response to the local private process, and then the processing of a single transaction is completed.
25. The responder BusinessConnect Container Edition server, after sending an MSH acknowledgment message, validates the payload signal document in the receipt acknowledgment message.
26. The responder BusinessConnect Container Edition server then publishes an advisory message to its private process, and then the processing of a single transaction is completed.

Example 2: Synchronous Reply Mode — mshSignalsOnly

In this example, only the MSH acknowledgment is sent synchronously. The response and the receipt acknowledgement are sent asynchronously.

Figure 14: Example 2: Synchronous Reply Mode — msySignalsOnly



The following points are noteworthy:

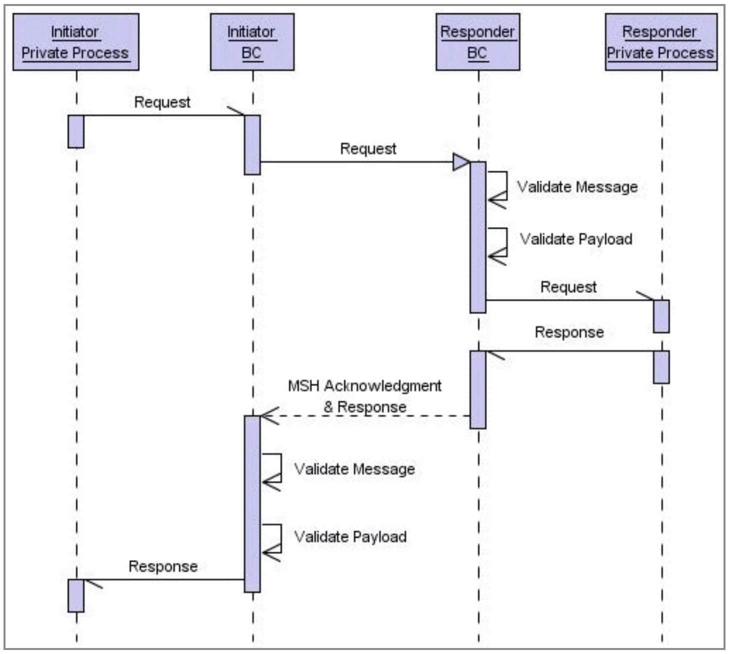
- The MSH acknowledgment signal is sent synchronously, meaning that it is sent by using the reply wing of the request-reply channel used by each of the request, response and signal messages.
- The following messages and signals are sent asynchronously: request, response, receipt acknowledgement, and exception.
- The mshSignalsOnly mode also applies to ErrorList messages. If an inbound message

or business signal is not validated successfully, the ErrorList is also sent back synchronously.

Example 3: Synchronous Reply Mode — responseOnly

As described earlier, in this mode, the response is sent back synchronously. The MSH acknowledgement, if required, is sent back in the same message as the response.

Figure 15: Example 3: Synchronous Reply Mode — responseOnly



The following points are noteworthy:

- The response message, along with the MSH acknowledgement (if applicable), is sent back synchronously, meaning that the response message is sent back by using the original connection that is used to send the request.
- In responseOnly mode, the responder side cannot require an MSH acknowledge to acknowledge the response action. This is because the response is sent synchronously and cannot demand an MSH acknowledgment signal.
- No business signals, such as a receipt acknowledgment message, can be used in this mode.

Error Handling

BusinessConnect Container Edition - ebXML Protocol sends *MSH error* signals to the trading partner whenever it encounters an error when processing the inbound ebXML action or business-level signal messages. See Processing Business-Level Signals. Reasons for an MSH error message include, but are not limited to the following information:

- A delivery failure occurred.
- The ebXML message envelope is not well formed, cannot be validated, and fails to resolve external references, and so on.
- The transaction is not authorized.
- The message time-to-live has expired.
- The ebXML message data does not match up with the settings in the BusinessConnect Container Edition - ebXML Protocol configuration store, such as settings for trading partners, duplicate elimination, digital signatures, encryption, acknowledgments, synchronous reply mode, and so on.
- The message did not pass security checks, such as digital signature verification.

  When BusinessConnect Container Edition - ebXML Protocol receives an MSH error message from the trading partner in response to an outbound message, it updates the private process by publishing the following private messages:

- **In case of trading partner MSH error while processing response message:** if an `ae/ebXML/ResponderAck` message has not already been sent to the private process, an `ae/ebXML/ResponderAck` message with an error code in the **statusCode** field. See Response Acknowledgment Format.
- An `ae/ebXML/Advisory` message on the error subject. See Error Message Format.

Note: The `ae/ebXML/Advisory` message is sent only on the error subject for inbound MSH errors, whereas it is sent on both the error and advisory signal subjects for inbound business-level exceptions. See Exception Messages.

Sequenced Messages

The BusinessConnect Container Edition - ebXML Protocol processes the sequenced ebXML messages, which you can use to employ ebXML conversations for messages that are ordersensitive.

Note:

- BusinessConnect Container Edition - ebXML Protocol does not support sequenced messages for the ebMS Ping and StatusRequest services.
- Sequenced messages must be delivered asynchronously (see Synchronous Reply Modes) and with once-and-only-once semantics (see Delivery Semantics). See the description for Synchronous Reply Mode and Configuring for Delivery Semantics for more information on how these are configured in BusinessConnect Container Edition.

Processing Outbound Sequenced Messages

A conversation can be initiated in two ways:

- The private process initiates an `ae/ebXML/InitiatorRequest` message with the **sequence** field set to `true` and (optionally) the **conversationID** field specified.
- The trading partner initiates a public request message with a conversation ID and a sequence number of `0`.

BusinessConnect Container Edition - ebXML Protocol sends an outbound sequenced request or response message for a given conversation. It specifies the conversation ID in the SOAP header of the outbound message and generates a sequence number to indicate the order by which the message is sent, beginning with the first message, in the following sequence: `0`, `1`, `2`, `3`, and so on. See Notes on Sequenced Messages.

Note: When an outbound response message is sequenced, it depends on whether the inbound request message contains a sequence number.

See Private Process Message Formats for information on private message fields related to sequenced messages: `sequenced`, `conversationID`, and `sequenceNo`.

See MessageHeader Element for formatting information on the `<ConversationID>` element, and MessageOrder Element for formatting information on the `<MessageOrder>` element.

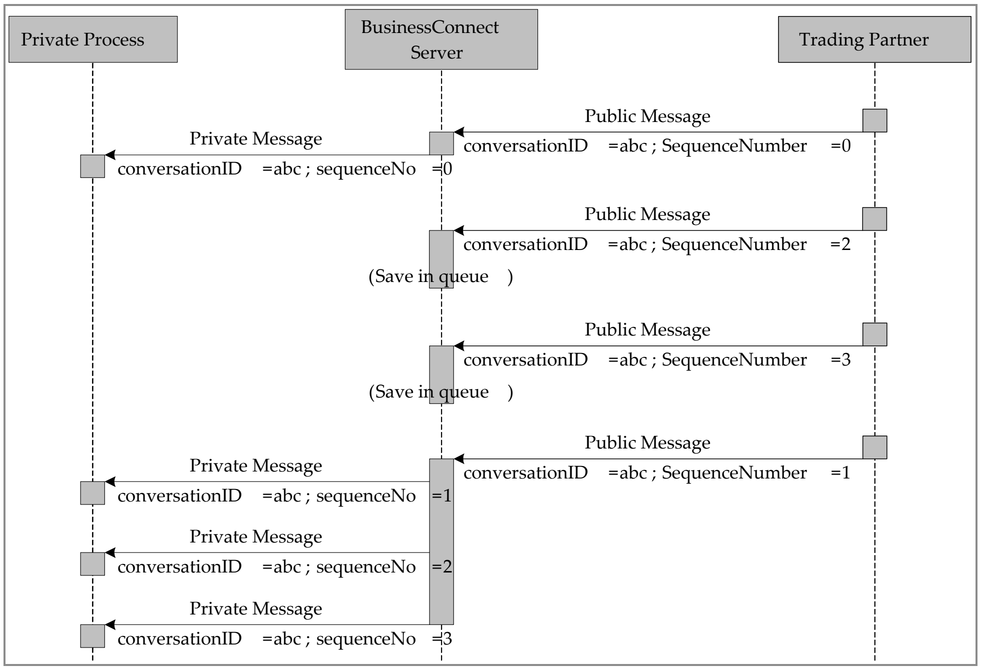
Processing Inbound Sequenced Messages

Because of the dynamics of the internet, over which the public messages are transmitted, it cannot be guaranteed that inbound sequenced messages arrive in sequence from the trading partner. When BusinessConnect Container Edition - ebXML Protocol receives inbound request or response messages that are out of sequence, it keeps them in a message queue and dispatches them to the private process when the in-sequence messages arrive.

Note: BusinessConnect Container Edition - ebXML Protocol processes sends applicable MSH acknowledgments and receipt acknowledgments for out-of-sequence inbound messages to the trading partner *before* storing the messages in the message queue.

The following figure describes how BusinessConnect Container Edition - ebXML Protocol processes sequenced messages:

Figure 16: Processing Sequenced Messages


Private Process
BusinessConnect Server
Trading Partner
Public Message
conversationID =abc ; SequenceNumber =0
Private Message
conversationID =abc ; sequenceNo =0
Public Message
conversationID =abc ; SequenceNumber =2
(Save in queue)
Public Message
conversationID =abc ; SequenceNumber =3
(Save in queue)
Public Message
conversationID =abc ; SequenceNumber =1
Private Message
conversationID =abc ; sequenceNo =1
Private Message
conversationID =abc ; sequenceNo =2
Private Message
conversationID =abc ; sequenceNo =3

When BusinessConnect Container Edition - ebXML Protocol receives the inbound messages with sequence numbers 2 and 3, it stores the out-of-sequence messages into the message queue. When it receives the inbound message with the sequence number 1, it sends the three messages in sequence to the private process.

Duplicate Sequence Numbers

In the ebXML protocol, sequence numbers are unique for an active conversation ID and the MSH that sends the sequenced messages. The two sides keep independent message sequences for their own outbound messages, and each must keep track of the active conversation ID and the sequence numbers of the other side.

When BusinessConnect Container Edition - ebXML Protocol receives a message from the trading partner with a sequence number that has already been used by the trading partner for a given active conversation ID, it processes the message as follows:

- If the sequence number is the duplicate of a message that is already forwarded to the private process, and then the inbound message is simply discarded.
- If the sequence number is the duplicate of a message that is kept in the message queue, and then the inbound message is also stored in the message queue. When the in-sequence message arrives and messages in the message queue are dispatched, one among the messages of the same sequence number is dispatched. Note that BusinessConnect Container Edition - ebXML Protocol does not determine which one is dispatched among messages of the same sequence number.

Expired Messages

Messages that are kept in the message queue might expire when waiting for the in-sequence messages to arrive. In this case, BusinessConnect Container Edition - ebXML Protocol does not dispatch the expired messages in the queue to the private process, and sends a business-level signal to the trading partner to indicate the message expiration.

However, it records the expired messages as having been dispatched to the private process, so that subsequent messages in the sequence can be processed and dispatched to the private process.

Therefore, private processes can receive out-of-sequence messages, and it is up to the private process to determine whether the out-of-sequence messages are useful or not.

BusinessConnect Container Edition - ebXML Protocol removes expired messages by polling the message queue periodically. You can configure how often BusinessConnect Container Edition - ebXML Protocol polls the message queue by setting protocol-specific properties. See Property Reference.

Expired Conversations

In BusinessConnect Container Edition - ebXML Protocol, you can also define an expiration period for active conversations. After a conversation expires, BusinessConnect Container Edition - ebXML Protocol resets the sequence number to `0` the next time it processes messages with the same conversationID. For more information, see the description for the **Queued Conversation Expiration Period (days)** field in Configuring ebXML Protocol for Trading Hosts.

Consistency of Message Settings

In ebXML, it is imperative that you and your trading partner use messages and message elements that are consistent with MSH configuration of each other. This is best achieved if you and your trading partner reach a detailed agreement for each of the business transaction that you perform together. This business agreement is the equivalent of an ebXML CPA.

It must include the following details for *every* transaction:

- Transaction name
- Role and service information
- Timeout settings
- Duplicate elimination
- Digital signatures
- Encryption
- MSH acknowledgments
- Receipt acknowledgments
- Synchronous reply mode

When BusinessConnect Container Edition - ebXML Protocol receives an inbound message that is inconsistent with any of the settings, it ends the transaction and sends an MSH error to the trading partner.

Example: If BusinessConnect Container Edition - ebXML Protocol is configured *never* to require an MSH acknowledgment for the request action, and a request message from the trading partner contains an `<AckRequest>` element, and then BusinessConnect Container

Edition - ebXML Protocol ends the transaction and sends an MSH error to the trading partner.

Preparing to Use BusinessConnect Container Edition - ebXML Protocol

With the preliminary information, you must configure your BusinessConnect Container Edition server for ebXML transactions. It also provides instructions for configuring some of this information in your BusinessConnect Container Edition server. You must perform all the tasks in this section before configuring BusinessConnect Container Edition participants with the ebXML protocol.

Exchanging Information with Your Trading Partners

Before you start conducting electronic business with your trading partner, you must exchange the following details:

- **CPA** Or the equivalent of a CPA. This is the ebXML business agreement between you and your trading partner. See Consistency of Message Settings.
- **Document Validation Schemas** The schemas used to validate each business document.
- **Domain Identity** The domain identity that you and your trading partner use to identify each other.
- **Server URLs** You and your trading partner must use the server URL of each other to conduct electronic business transactions. If you agree on multiple protocols, you must exchange server URLs for each of the agreed protocols.

  Server URLs for BusinessConnect Container Edition - ebXML Protocol follow the formats:

  protocol`://`*hostName*`:`*portNumber*`/dmz/ebXML`

  Example:

  ```
  http://www.myhost.com:6700/dmz/ebXML
  https://www.myhost.com:6705/dmz/ebXML
  ```

- **Public Certificates** To use HTTPS for your ebXML transactions, and utilize digital signatures for document authentication, you must also exchange public certificates.

Configuring Your BusinessConnect Container Edition Server

You must perform a few tasks in the BusinessConnect Container Edition console before you can successfully configure a BusinessConnect Container Edition participant with the ebXML protocol.

Transports

Before you can configure a BusinessConnect Container Edition transport protocol of a participant, you must configure the transport protocols. BusinessConnect Container Edition - ebXML Protocol supports the following two types of transports: HTTP and HTTPS (with or without client authentication).

For more information on how to configure the outbound transport to send ebXML messages to your trading partner, see the "HTTP, HTTPS, and HTTPSCA Transports" topic in *TIBCO BusinessConnect™ Container Edition Trading Partner Management*.

For more information on how to configure the transport services with your trading partner, see the *TIBCO BusinessConnect™ Container Edition* documentation.

Copying Large Files to NFS (Sharing)

To copy files (greater than 5 MB as configured by default) to the NFS (sharing) folders, perform the following steps:

Procedure

1. Go to **System Settings** > **Others** > **Activate Protocol Plugins**.
2. Click **ebXML**.
3. In the Edit ebXML Plugin Settings dialog, click **Add Property**.

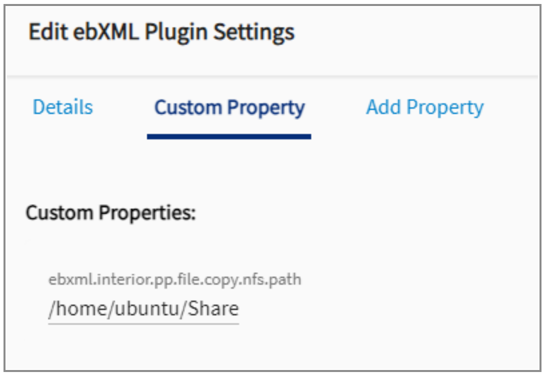
4. Set the following fields.

Field	Value
Property Name	`ebxml.interior.pp.file.copy.nfs.path`
Property Type	String

5. Click **Add**.

6. On the **Custom Property** tab, configure the `ebxml.interior.pp.file.copy.nfs.path` field to NFS path.

   Example: `/home/ubuntu/Share`



Edit ebXML Plugin Settings

Custom Properties:

ebxml.interior.pp.file.copy.nfs.path
/home/ubuntu/Share

Note: You must set the `ebxml.fileref.threshold` value to `5000000` (5 MB).

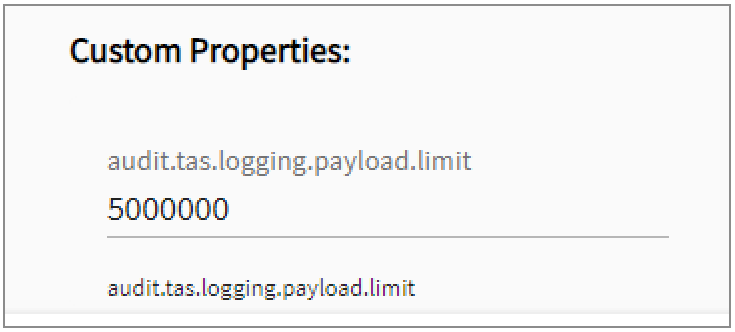
7. Click **Save**.

8. Similarly, create a custom property for BCCE with the following values.

Field	Value
Property Name	`audit.tas.logging.payload.limit`
Property Type	Integer

This limits the view or downloadable payload size on AuditSafe. The maximum limit is 5 MB.

Note: You must set the `audit.tas.logging.payload.limit` value to `5000000`.



Custom Properties:

audit.tas.logging.payload.limit

5000000

audit.tas.logging.payload.limit

Domains

To add a domain for your trading partner, perform the following steps:

Procedure

1. Go to **System Settings** > **Others** > **Metadata Type Configuration**.
2. On the **Domains** tab, click the **Add** icon ⊕ .
3. In the **Create Domain Type** dialog:
   a. In the **Name** field, enter a domain name.
   b. In the **Description** field, enter a description.
4. Click **Add**.

Managing ebXML Transactions

This section explains how to manage ebXML transactions in the **Operations Editor**.

The Operations Editor panel organizes ebXML transactions in containers of various levels. In Operations Editor, transactions defined by the same organization are grouped into the same *organization* container.

Example:

- `RosettaNet.org`: Within an organization container, they are organized into *business process* containers.
- `3A4`: Transactions further organized by their document version into *version* containers.
- `1.1`: A *transaction* is the most basic unit for an ebXML transactions in the Operations Editor panel.
- `Create Purchase Order`.

> **Note:** When you give a container any name you want, follow the naming convention to ensure that both you and your trading partner are using the same naming conventions for your ebXML transactions.

Before you use ebXML in business transactions, you must add the transactions in the Operations Editor panel and configure each of them accordingly. See Adding Transactions.

After adding the transactions in the Operations Editor panel, you can export them to a file. When you upgrade or reinstall BusinessConnect Container Edition - ebXML Protocol, you can simply import the data into the Operations Editor panel. See Exporting Transactions and Importing Transactions.

Adding Transactions

The section provides the instructions on adding transactions in the Operations Editor panel, which consists of the following tasks:

- Adding Organization Containers
- Adding Business Process Containers
- Adding Version Containers
- Adding Transactions

Adding Organization Containers

To add an organization container, perform the following steps:

Procedure

1. On the **B2B Administration** tile, click **Operations Editor**.
2. From the **Protocol** list, click **ebXML**.
3. Click the **Add** icon ⊕.
4. In the **New Organization** dialog:
    1. In the **Name** field, enter the organization name.
    2. Optional: In the **Description** field, enter a description.
    3. Optional: In the **Business Process Agency ID** field, enter the business process agency ID.
5. Click **Add**.

Adding Business Process Containers

To add a business process container, perform the following steps:

Procedure

1. Hover over the organization container that you created in Adding Organization Containers and click the **three-dot** menu.
2. Click **New Business Process**.
3. In the **New Business Process** dialog:
    1. In the **Name** field, enter the name of the business process as defined by the

organization you specified in Adding Organization Containers.

   b. Optional: In the **Description** field, enter a description.

4. Click **Add**.

Adding Version Containers

To add a version container, perform the following steps:

Procedure

1. Hover over the business process that you created in Adding Business Process Containers and click the **three-dot** menu.
2. Click **New Version**.
3. In the **New Version** dialog:
   a. In the **Name** field, enter the version of the business transaction.
   b. Optional: In the **Description** field, enter a description.
   c. Optional: To add a schema file for this version container, click **Upload file** and browse to locate the schema.

   > ℹ **Note:** If you have one schema file for each individual transaction, you can also add it when configuring the action specific properties of a transaction. See Document Tab.

4. Click **Add**.

Adding Transactions

To add a transaction, perform the following steps:

Procedure

1. Hover over the version container that you created in Adding Version Containers and click the **three-dot** menu.
2. Select either **New Notify Transaction** or **New Request-Response Transaction**.
3. Configure the transaction on each tab, see Notify and Request-Response Transaction

Properties.

4. Click **Add**.

Notify and Request-Response Transaction Properties

Depending on the type of transaction you specify in Adding Transactions, you see either the **New Notify Transaction** dialog or the **New Request-Response Transaction** dialog.

The following table lists the description of each field in the **New Notify Transaction** and **New Request-Response Transaction** dialogs.

✓ **Tip:** You can override certain transaction settings when configuring protocol bindings for a business agreement. See Operation Settings Tab.

Field	Description
General Tab	
Name	The displayed name of the transaction.
Description	Optional. A short description of the transaction.
Roles Tab	
Initiating Role	The role of the initiator in this transaction. This field is mapped to the `<Role>` element in the SOAP header of the public message.
Responding Role	The role of the responder in this transaction. This field is mapped to the `<Role>` element in the SOAP header of the public message.
Service Information Tab	
Service	The service that acts on this message, as defined by ebMS. This field is mapped to the `<Service>` element in the SOAP header of the public

Field	Description
	message. You must specify a URI in this field if the **Service Type** field is empty.
Service Type	The service type, as defined by ebMS. This field is mapped to the `Type` attribute in the `<Service>` element of the public message. You must specify a URI in the **Service** field if this field is empty.

Document Tab

Field	Description
Validate Outgoing Document	Select this checkbox to validate outbound documents.
Validate Incoming Document	Select this checkbox to validate inbound documents.
Encrypt Document	The type of encryption to use for outbound payload and all attachments. The options are as follows: • **NONE** No encryption. • **SMIME** Use SMIME encryption. • **XMLEncryption** Use XML Encryption. When this option is selected, the entire business document is encrypted and contained in a single XML `<EncryptedData>` element, and the value of the **Content-Type** field in the MIME header is determined by your selection from the MIME Content-Type of XMLEncryption list. If your message payload includes attachments, each attachment is encrypted separately and contained in its own XML document, and also in an `<EncryptedData>` element.
MIME Content-Type of XMLEncryption	The value of the **Content-Type** field in the MIME header when XML Encryption is used. The options are as follows: • text/xml

Field	Description
	- application/xml - application/xenc+xml

Action-Specific Properties

For request-response transaction, **Request Action** and **Response Action** tabs are additionally displayed in the **New Request-Response Transaction** dialog. Whereas for a notify transaction, **Notify Request Action** tab is displayed in the **New Notify Transaction** dialog.

You can use these action-specific tabs to specify details of individual actions in the transaction.

Each of these tabs has three sub-tabs. See the sections for each of these sub-tabs.

- General tab. See General Tab.
- Signal tab. See Signal Tab.
- Document tab. See Document Tab.

General Tab

The settings on the **General** tab determine how BusinessConnect Container Edition - ebXML Protocol processes request and response actions. See the following table for a description of each field on the **General** tab.

Note that the fields on the **General** tab vary slightly, depending on whether it is under the **Request Action** tab, **Response Action** tab, or **Notify Request Action** tab.

Action Tab: General Tab

Field	Description
Name	The name of the action.
Description	Optional. A description of the action.

Field	Description
Direction	The direction of this action.
Max. Response Wait Time (min.)	Configure on the **Request Action** tab only. The maximum length of time to wait for the response message from the trading partner, counting from the time the action message is sent. If this time expires, BusinessConnect Container Edition - ebXML Protocol sends an `ae/ebXML/Advisory` message on the error subject. **Note:** This function does not take effect if you select **responseOnly** from the **Synchronous Reply Mode** list.
Max. Private Process Response Wait Time (min.)	Configure on the **Response Action** tab only. The maximum length of time to wait for the response message from the private process, counting from the time the action message is sent to the private process. If this time expires, BusinessConnect Container Edition - ebXML Protocol sends an MSH error to the trading partner.
Lifetime of Message (min.)	The maximum length of time for the action message to be delivered. This field is used to calculate the value of the `<TimeToLive>` element in the SOAP header of outbound action messages. If the value of this field is `0`, BusinessConnect Container Edition - ebXML Protocol does not generate an `<TimeToLive>` element in the SOAP header of outbound action messages.
Require Non-repudiation of Request or Require Non-repudiation of Response	Require the action message to be digitally signed for non-repudiation logging. If this checkbox is selected, BusinessConnect Container Edition - ebXML Protocol behaves as follows: • Signs the outbound action message and stores the message in the non-repudiation log. • Verifies the signature of the inbound action message and stores the message in the non-repudiation log. Also sends an MSH error to the trading partner if the inbound action message is not signed.
Synchronous	The mode in which to receive replies to the outbound action message and

Field	Description
Reply Mode	to send replies to the inbound action message. The options are as follows: • **none** Send all reply messages in asynchronous mode. • **mshSignalsOnly** Send the MSH acknowledgment or MSH error synchronously. The receipt acknowledgment and the response message are sent asynchronously, if applicable. • **responseOnly** Configure on the **Request Action** tab only. Send the MSH acknowledgment and the response message synchronously and in the same package. See Synchronous Reply Modes. **Note:** • If you select **responseOnly**, you cannot use the BusinessConnect Container Edition console to save your settings with the **Require Receipt Acknowledgment Signal** checkbox selected on the **Signal** tab. See Signal Tab. • You cannot set **BusinessConnect Container Edition > B2B Administration > Operations Editor > ebXML > transaction > Request Action > General > Synchronous Reply Mode** to **responseOnly** and **BusinessConnect Container Edition > Operations Editor > ebXML > transaction > Response Action > Signal > Require MSH Acknowledgment Signal** to **always** or **perMessage** for the same transaction. If you do, you cannot use the BusinessConnect Container Edition console to save your settings. This is because the synchronous response message cannot require an MSH acknowledgment.
Eliminate Duplicate Message	The receiver of the action message must eliminate duplicate messages. BusinessConnect Container Edition - ebXML Protocol can remove duplicates of action messages that are kept in persistent storage. The options are as follows: • **never** Do not eliminate duplicate messages. If this option is selected, BusinessConnect Container Edition - ebXML Protocol does not include the `<DuplicateElimination>` element in the SOAP header of the outbound action message, and also verifies that the inbound action message does not contain the

Field	Description
	`<DuplicateElimination>` element. • **always** Always eliminate duplicate messages. If this option is selected, BusinessConnect Container Edition - ebXML Protocol always includes the `<DuplicateElimination>` element in the SOAP header of the outbound action message, and also verifies that the inbound action message contains the `<DuplicateElimination>` element. • **perMessage** If this option is selected, BusinessConnect Container Edition - ebXML Protocol does the following operations: — Includes the `<DuplicateElimination>` element in the SOAP header of the outbound action message if the **dupElimination** field in the `ae/ebXML/InitiatorRequest` or `ae/ebXML/ResponderResponse` message is set to `true`. — Eliminates duplicate messages only if the inbound action message contains the `<DuplicateElimination>` element.
Persist Duration (min.)	The maximum length of time the action message for both outbound and inbound is kept in persistent storage. BusinessConnect Container Edition - ebXML Protocol sends an MSH error to the trading partner if the outbound action message cannot be successfully delivered before Persist Duration has passed. **Note:** The value of this field must be greater than the value in the **Lifetime of Message** field. Also, if the **Require MSH Acknowledgment Signal** list is selected in the **Signal** tab, the value of this field must likewise be greater than the following formula based on the values in the **Time to Wait for MSH Acknowledgment** and **Maximum Number of Retries** fields in the **Signal** tab: (**Maximum Number of Retries** + 1) x S.

Signal Tab

On the **Signal** tab, specify the properties related to sending or receiving signal messages.

Field	Description
MSH Acknowledgment	
Require MSH Acknowledgment Signal	The receiver of the action message must send an MSH acknowledgment to the message sender. This setting applies to all action messages and the receipt acknowledgment signal. The options are as follows: • **never** Do not send MSH acknowledgments. If this option is selected, BusinessConnect Container Edition - ebXML Protocol does not include the `<AckRequested>` element in the applicable outbound message, and also verifies that the applicable inbound message does not contain the `<AckRequested>` element. • **always** Always send MSH acknowledgments. If this option is selected, BusinessConnect Container Edition - ebXML Protocol always includes the `<AckRequested>` element in the SOAP header of the applicable outbound message, and also verifies that the applicable inbound message contains the `<AckRequested>` element. • **perMessage** If this option is selected, BusinessConnect Container Edition - ebXML Protocol does the following operations: — If the **ackRequested** field in the `ae/ebXML/InitiatorRequest` or `ae/ebXML/RespoderResponse` message is set to `true`: Includes the `<AckRequested>` element in the outbound action message. — If the applicable inbound message contains the `<AckRequested>` element: Sends the MSH acknowledgment; and includes the `<AckRequested>` element in the outbound receipt acknowledgment if a receipt acknowledgment is required.
Time to Wait for MSH Acknowledgment	The duration of time to wait for an MSH acknowledgment from the trading partner. After this time expires, the action message is resent to the trading partner as many times as specified in the **Maximum**

Field	Description
(min.)	has passed, BusinessConnect Container Edition - ebXML Protocol sends an MSH error to the trading partner. **Note:** This field has no effect when the **Synchronous Reply Mode** list on the **General** tab is set to `responseOnly` or `mshSignalsOnly`. See Action Tab: General Tab . In synchronous transactions, BusinessConnect Container Edition - ebXML Protocol lets the transport layer to deliver the message for as long as possible, including multiple connection attempts. For more information on configuring the transport layer, see Transports Tab.
Maximum Number of Retries	The number of times to retry sending the action message before receiving an MSH acknowledgment.
Require Non-repudiation of MSH Acknowledgment	Require the MSH acknowledgment to be digitally signed and stored into the non-repudiation log. The options are as follows: • **none** Do not use digital signatures for MSH acknowledgments. If this option is selected, BusinessConnect Container Edition - ebXML Protocol does not include the `signed` attribute in the `<AckRequested>` element of the applicable outbound message, and does not include the `<Signature>` element in the outbound MSH acknowledgment. Same operations are also required from the trading partner. • **always** Always use digital signatures in MSH acknowledgments. If this option is selected, BusinessConnect Container Edition - ebXML Protocol always includes the `signed` attribute with the value of `true` in the `<AckRequested>` element of the applicable outbound message, and includes the `<Signature>` element with a valid digital signature in the outbound MSH acknowledgment. Same operations are also required from the trading partner. • **perMessage** If this option is selected, BusinessConnect Container Edition - ebXML Protocol does the following operations: — If the **ackRequested** field in the `ae/ebXML/InitiatorRequest` or

Field	Description
	`/RespoderResponse` message is set to `true`: sets the `signed` attribute in the `<AckRequested>` element to `true` of the outbound action message, and also verifies the `<Signature>` element in the inbound MSH acknowledgment. — If the `signed` attribute in the applicable inbound message’s `<AckRequested>` element is set to `true`: Includes the `<Signature>` element with a valid digital signature in the outbound MSH acknowledgment. Also, when a receipt acknowledgment is required, sets the `signed` attribute in the `<AckRequested>` element of the outbound receipt acknowledgment, and verifies the `<Signature>` element in the inbound MSH acknowledgment.
Receipt Acknowledgment	
Require Receipt Acknowledgment Signal	The receiver of the action message must send a receipt acknowledgment to the message sender. If this checkbox is selected, BusinessConnect Container Edition - ebXML Protocol sends a receipt acknowledgment for each inbound action message, and a receipt acknowledgment for each outbound action message is required. If this checkbox is not selected, BusinessConnect Container Edition - ebXML Protocol does not send receipt acknowledgments, and the same operations are also required from the trading partner.
Time to Wait for Receipt Acknowledgment Signal (min.)	The length of time to wait for a receipt acknowledgment from the trading partner. After this time has elapsed, BusinessConnect Container Edition - ebXML Protocol sends an MSH error to the trading partner.
Require Non-repudiation of Receipt	Require the receipt acknowledgment to be digitally signed and stored into the non-repudiation log. If this checkbox is selected, BusinessConnect Container Edition - ebXML Protocol digitally signs the outbound receipt acknowledgment and requires the inbound receipt acknowledgment to be digitally signed.

Field	Description
	If this checkbox is not selected, BusinessConnect Container Edition - ebXML Protocol does not sign the outbound receipt acknowledgment, and the same operations are also required from the trading partner.

Document Tab

Field	Description
Business Document Name	The name of the business document in the action message. **Note:** This name cannot contain spaces.
Schema	
Schema Definition Reference	The validation schema for this business document. If you have added a schema file in the version container, the schema file you add here overrides the one in the version container.
For TIBCO ActiveMatrix BusinessWorks™ Plug-in for BusinessConnect™ Use Only	
XML Validation Type	The type of the validation to use for XML documents.
Root XML Element Name	The XML root element name for this business document, as defined by the schema file in Schema Definition Reference.

Configuring for Delivery Semantics

You can configure BusinessConnect Container Edition - ebXML Protocol to send messages using different delivery semantics by using different settings for the following GUI elements:

- **BusinessConnect Container Edition** > **B2B Administration** > **Operations Editor** > **ebXML** > **`transaction`** > **Action tab** > **General** > **Eliminate Duplicate Message**
- **BusinessConnect Container Edition** > **B2B Administration** > **Operations Editor** > **ebXML** > **`transaction`** > **Action tab** > **Signal** > **Require MSH Acknowledgment Signal**

The following table lists how to configure the following two properties for different delivery semantics. For more information on each delivery semantics, see Delivery Semantics.

Delivery Semantics Configuration

Delivery Semantics	Eliminate Duplicate Message	Require MSH Acknowledgment Signal
Best Effort	`never`	`never`
At-Most-Once	`always`	`never`
At-Least-Once	`never`	`always`
Once-And-Only-Once	`always`	`always`

Exporting Transactions

You can export all the ebXML transaction configurations. You can also export a subset of the transactions: by individual transaction (Export Transaction), by version container (Export Version), by business process container (Export Business Process), or by organization container (Export Organization).

To export the ebXML transactions, perform the following steps:

Procedure

1. On the **B2B Administration** tile, click **Operations Editor**.
2. To export all the ebXML transactions:
   a. Select the **ebXML** checkbox.
   b. Click the **Export** icon.

   c. On the **Export Operations** dialog, click **Export**.

      You can now see the File Download dialog with `operations.bcce` as the suggested filename.

3. Click **Save** on the File Download dialog.
4. Alternatively, to export a subset of the ebXML transactions:

   a. On the Operations Editor | ebXML page, select the organization, business process, version, or transaction you want to export.

   b. Click the **Export** icon.

   c. On the **Export Operations** dialog, click **Export**.

      You can now see the File Download dialog with `operations.bcce` as the suggested filename.

5. Change to a desired filename and click **Save**.

Importing Transactions

To import the ebXML transactions, perform the following steps:

Procedure

1. On the **B2B Administration** tile, click **Operations Editor**.
2. From the **Protocol** list, click **ebXML**.
3. Click the **Import** icon.
4. On the **Import Operations Data** dialog, click **Upload file**.

   Enter the password, if required.

5. Locate the exported `.bcce` file and click **Open**.
6. Click **Import**.

Setting Up Trading Hosts and Partners

This section explains how to set up trading hosts and partners in BusinessConnect Container Edition - ebXML Protocol.

Configuring ebXML Protocol for Trading Hosts

This section explains how to configure the ebXML protocol for a trading host in BusinessConnect Container Edition. It assumes that you have already set up a host participant. For more information about setting up a trading host, see *TIBCO BusinessConnect™ Container Edition Trading Partner Management*.

To configure the ebXML protocol for a trading host, perform the following steps:

Procedure

1. On the **Partner Management** tile, click **Hosts**.
2. Click the name of a host participant.
3. On the **Protocols** tab select the **ebXML** checkbox.
4. Click **Edit Configurations** next to the **ebXML** checkbox.
5. Configure the properties on the **General** tab. See the following table for a description of each property.

Trading Host General Properties

Field	Description
Default Domain Identity	The default domain identity to use for this host. Select from the list of domain-identity pairs that you created in the Domain Identities dialog. See Adding Domain Identities to add or edit a domain identity.
Valid Email	Currently not used.

Field	Description
Address List	
Queued Conversation Expiration Period (days)	The number of days a sequenced conversation is kept in the system before it is deleted. To keep sequenced conversations indefinitely, type **`0`** or a negative number, such as **`-1`** in this field.

6. Click **Save**.

Adding Domain Identities

You must configure at least one domain-identity pair for your host participant. The domain value in the domain-identity pair corresponds to the `type` attribute of the `<PartyId>` element in the public message, and the identity value corresponds to the `<PartyId>` element itself.

To add a domain-identity pair for the ebXML protocol, perform the following steps:

Procedure

1. On the **General** tab, click the **Add** icon ⊕ next to the **Default Domain Identity** field.
2. On the **Add New** tab, from the **Domain Type** dropdown list, select a domain.

   This is mapped to the domain string in the host domain-identity pair. If the domain you want to use does not exist, see Domains for instructions on adding domains.
3. In the **ID** field, enter the identity of the host participant.
4. Click **Add**.

> **Note:** If the domain you specify is `URI`, you must specify a valid URI in the **ID** field. Example: **`urn:123456789`**.

Configuring ebXML Protocol for Trading Partners

This section explains how to configure the ebXML Standard protocol for a trading partner in BusinessConnect Container Edition. It assumes that you have already set up a partner participant. For more information about setting up a trading partner, see *TIBCO BusinessConnect™ Container Edition Trading Partner Management*.

To configure the ebXML protocol for a trading partner, perform the following steps:

Procedure

1. On the **Partner Management** tile, click **Partners**.
2. Click the name of a trading partner.
3. On the **Protocols** tab select the **ebXML** checkbox.
4. Click **Edit Configurations** next to the **ebXML** checkbox.
5. Configure the following tabs:
    - **General**

      See General Tab for more information on configuring this tab.
    - **Transports**

      See Transports Tab for more information on configuring transports.
6. Click **Save**.

General Tab

To manage general information for the ebXML trading host, configure the **General** tab.

Trading Partner General Properties

Field	Description
Default Domain Identity	The default domain identity to use for this partner. Select from the list of domain identities created in the Domain Identity dialog. See Adding

Field	Description
	Domain Identities to add or edit a domain identity.
Valid Email Address List	Currently not used.
Digital Signature KeyInfo Type	The information to include in the `<KeyInfo>` element of the SOAP header when sending a signed message to this trading partner. The options are as follows: • **None** Do not include the `<KeyInfo>` element. • **X509Cert** Include the public certificate of the trading host. • **X509Chain** Include the entire public certificate chain of the trading host, including trusted certificate authorities. • **KeyValue** Include only the public key of the trading host. • **All** Include all the previous options.
Specification ID	The URI of the CPA between the host and this trading partner. It corresponds to the `<CPAId>` element in the SOAP header of the public message.

Adding Domain Identities

You must configure at least one domain-identity pair for your partner participant. The domain value in the domain-identity pair corresponds to the `type` attribute of the `<PartyId>` element in the public message, and the identity value corresponds to the `<PartyId>` element itself.

To add a domain-identity pair for the ebXML protocol, perform the following steps:

Procedure

1. On the **General** tab, click the **Add** icon ⊕ next to the **Default Domain Identity** field.

2. On the **Add New** tab, from the **Domain Type** dropdown list, select a domain.

   This is mapped to the domain string in the host domain-identity pair. If the domain you want to use does not exist, see Domains for instructions on adding domains.

3. In the **ID** field, enter the identity of the host participant.
4. Click **Add**.

> **Note:** If the domain you specify is `URI`, you must specify a valid URI in the **ID** field. Example: **`urn:123456789`**.

Transports Tab

On the **Transports** tab, configure outbound transport settings for sending ebXML messages to the trading partner.

To add an outbound transport, perform the following steps:

Procedure

1. On the **Transports** tab, click **Add Outbound Transport**.
2. In the **Add Transport** dialog:
   a. In the **Transport Name** field, enter the transport name.
   b. From the **Transport Type** dropdown list, select the transport type.

> **Note:** The following transports are available to use with BusinessConnect Container Edition - ebXML Protocol:
>
> - HTTP
> - HTTPS (HTTPSCA is included.)

The steps required for configuring transports are the same for all protocols. For more information, see the transports section in *TIBCO BusinessConnect™ Container Edition Trading Partner Management*.

Configuring Agreement Protocol Bindings

This section describes setting up business agreements to use BusinessConnect Container Edition - ebXML Protocol.

Configuring Business Agreements with ebXML Protocol

To configure a business agreement with the ebXML protocol binding, perform the following steps:

Procedure

1. On the **Partner Management** tile, click **Business Agreements**.
2. Click the **`<host-partner>`** agreement that you want to edit.
3. On the **Bind Protocol** tab, select the **ebXML** checkbox.
4. Click **Edit Configurations** next to the **ebXML** checkbox.
5. Configure the following tabs:
    - **Operations** See Configuring Operation Bindings.
    - **Document Exchange** Setting Document Security Properties.
    - **Transports** See Configuring Transports.
    - **Override Configuration** See Overriding Participant Settings.
6. Click **Save**.

Configuring Operation Bindings

Use the **Operations** tab to configure the ebXML transactions that each party in a business agreement can initiate and respond to.

- **Allow All Operations**

  All transactions configured in the Operations Editor between participants can be used. If you select this option, you can still modify the behavior of one or more transactions by binding the transactions in the **Initiating Operations** and **Responding Operations** sections.

  If you do not select this option, you must explicitly bind each transaction for each party in the **Initiating Operations** and **Responding Operations** sections.

Binding Operations

The **Initiating Operations** section lists the operations that the host can initiate in this agreement. The **Responding Operations** section lists the operations that the partner can initiate.

To bind operations, perform the following steps:

Procedure

1. Click **Add Operations**.
2. Click the **Caret** icon ▶ to expand the browser tree.
3. Select the ebXML transactions that you want to bind.
4. Click **Done**.

Editing Operation Bindings

To edit an ebXML operation binding, perform the following steps:

Procedure

1. Click the name of an operation in the **Initiating Operations** or **Responding Operations** section.
2. Configure this particular operation binding on the following tabs:
   a. **Operation Settings**

      Override the default settings for this transaction. See Operation Settings Tab.

   b. **Action Settings**

      Override settings chosen in Action Settings Tab.

   c. **Transports**

      Override the default transport settings for this transaction. See Transports Tab.

3. Click **Save**.

Operation Settings Tab

You can override the default operation settings in Operations Editor that are on the **Request-Response Transaction** tab or **Notify Transaction** tab of a transaction. For information on configuring the default settings of a transaction in the Operations Editor, see Notify and Request-Response Transaction Properties.

To override the default operation settings, perform the following steps:

Procedure

1. On the **Override Configuration** tab, enable the **Override Settings** toggle.
2. Select one of the following items from the list to override operation settings for this transaction:

   - **Roles**

     Selecting this item to override the fields on the **Roles** tab for this transaction in the Operations Editor panel. See the Roles section in Notify and Request-Response Transaction Properties for a description of each field.

   - **Document**

     Selecting this item to override the fields on the **Document** tab for this transaction in the Operations Editor panel. See the Document section in Notify and Request-Response Transaction Properties for a description of each field.

Action Settings Tab

You can also override the default settings of a specific action. You can find the default action settings of a transaction in the Operations Editor.

The Operations Editor is on the **General** and **Signal** tabs of the **Request Action** tab or **Notify Request Action** tab. For more information on configuring the default action settings of an operation in the Operations Editor, see Action-Specific Properties.

> ℹ **Note:** Operations you select in the **Initiating Operations** section can only have their **Request Action** or **Notify Request Action** tab settings overridden, while operations you select in the **Responding Operations** section can only have their **Response Action** tab settings overridden.

To override action settings, perform the following steps:

Procedure

1. Click the **Action Settings** tab.
2. Select the **Override Action Settings** checkbox.
3. Select one of the following items from the list to override operation settings for this transaction:
   a. **General** Select this item to override the fields on the **General** tab for this action. See Action Tab: General Tab for a description of each field.
   b. **Signal** Select this item to override the fields on the **Receipt** tab for this action. See Action-Specific Properties for a description of each field.

Transports Tab

After you define transport settings for a business agreement, you can override the transport settings for a specific transaction.

To override outbound transport settings, perform the following steps:

Procedure

1. Click the **Transports** tab.
2. Select the **Override Transports** checkbox.
3. Select the transport that you want to use from the **Primary Transport** list.

Setting Document Security Properties

On the **Document Exchange** tab, specify security information for the transacted ebXML documents between the participants in this business agreement. The keys and certificates selected on the tab are configured on the **Credentials** tab of a participant. For information about configuring participant credentials, see *TIBCO BusinessConnect™ Container Edition Trading Partner Management*.

Follow the instructions in Configuring Business Agreements with ebXML Protocol to get to the **Document Exchange** tab. See the following table to configure the document security properties.

Document Security Properties

Field	Description
Signing Key	The private key of the selected host. BusinessConnect Container Edition - ebXML Protocol uses this key to sign outbound documents in this business agreement. You must have already installed a private key for this host during participant configuration. See *TIBCO BusinessConnect™ Container Edition Trading Partner Management* for more information on installing private keys for host participants.
Digest Algorithm	BusinessConnect Container Edition - ebXML Protocol supports SHA1, SHA256, SHA384, and SHA512 as the digest algorithms for XML signatures.
Encryption Certificate	The public certificate of the selected partner. BusinessConnect Container Edition - ebXML Protocol uses this certificate to encrypt outbound messages in this business agreement. You must have already installed a certificate for this partner during participant configuration. See *TIBCO BusinessConnect™ Container Edition Trading Partner Management* for more information on installing certificates for partner participants.
Encryption Algorithm	The algorithm used to encrypt documents. The available options are DES3, AES-128, AES-192, AES-256.
Inbound Doc Exchange	

Field	Description
Verification Certificate	The public key of the selected partner. BusinessConnect Container Edition - ebXML Protocol uses this key to verify the signed inbound documents in this business agreement. You must have already installed a certificate for this partner during participant configuration. See *TIBCO BusinessConnect™ Container Edition Trading Partner Management* for more information on installing certificates for partner participants.
Decryption Key	The private key of the selected host. BusinessConnect Container Edition - ebXML Protocol uses this key to decrypt inbound messages in this business agreement. You must have already installed a private key for this host during participant configuration. See *TIBCO BusinessConnect™ Container Edition Trading Partner Management* for more information on installing private keys for host participants.

Configuring Transports

As part of the ebXML protocol binding in a business agreement, you must specify which transports can be used for the host and the partner.

Follow the instructions in Configuring Business Agreements with ebXML Protocol to get to the **Transports** tab, and read this section for more information on how to configure this tab.

Outbound Transports for Host

On this section of the **Transports** tab, assign a transport for sending business transaction messages to the trading partner. The transport methods are configured as a part of the partner configuration: **BusinessConnect Container Edition** > **Partner Management** > **Partners** > **Name** > **Protocols** > **ebXML** > **Edit Configurations** > **Transports**. Click **Add Outbound Transport** and add.

For more information about configuring transports, see *TIBCO BusinessConnect™ Container Edition Trading Partner Management*.

The following table lists and describes the options available for transport assignment with ebXML.

Outbound Transports for Host

Field	Description
Primary Transport	The transport to use for sending outbound messages.
Client Authentication Identity for HTTPS, FTPS, HTTPSCA	The key identity used when the remote server requires client authentication for an HTTPS connection.

Inbound Transports for Partner

On this section of the **Transports** tab, specify the transport types your partner can use to send ebXML messages to your BusinessConnect Container Edition. The transport types available for selection here reflect the transports configured for this BusinessConnect Container Edition deployment. Transports for the deployment are configured in BusinessConnect Container Edition: **BusinessConnect Container Edition** > **System Settings** > **Transport Protocols** > **Inbound Protocols**.

For more information about configuring transports in the Inbound Public Transport Types panel, see *TIBCO BusinessConnect™ Container Edition Trading Partner Management*.

Depending on your deployment configuration, the following transports might be available for selection:

Inbound Transport Fields

Field	Description
HTTPSCA	Client Authentication can be used when you select HTTPS as the allowed inbound transport.
HTTPS	HTTPS connections can be used from this partner directly.
HTTP	HTTP connections can be used from this partner directly.

Overriding Outbound Transport Settings

The selected outbound transport is the default transport of this Business Agreement. However, you can override this with another transport for a particular operation.

To override the outbound transport settings, perform the following steps:

Procedure

1. In the Edit Protocol Bindings: ebXML panel, click the **Operation Bindings** tab.
2. Click an operation in the **Initiating Operations** or **Responding Operations** section.
3. Continue by following the instructions in Transports Tab.

Overriding Participant Settings

You can use the ebXML panel to override the general protocol settings of each participant, which are configured on the **General** tab of the participant ebXML protocol configuration panel. For more information about configuring these default settings, see Configuring ebXML Protocol for Trading Hosts and Configuring ebXML Protocol for Trading Partners.

To override the participant settings for a business agreement, perform the following steps:

Procedure

1. Follow the instructions in Configuring Business Agreements with ebXML Protocol to get to the Edit Protocol Bindings: ebXML panel.
2. Go to the **Override Configuration** tab.

   Two configuration sections are displayed, one for each participant in this business agreement.
3. Enable the **Override Settings** toggle in the configuration section for the desired participant.
4. Configure the settings in this section. See Trading Host General Properties for a host or Trading Partner General Properties for a partner.
5. Click **Save**.

Private Processes

This section describes the interaction between BusinessConnect Container Edition - ebXML Protocol and its private processes and provides specifications for private process messages.

Configuring Private Processes

You can use the following two types of private processes with BusinessConnect Container Edition - ebXML Protocol:

- **Standalone**

  Standalone private processes must directly use JMS to communicate with BusinessConnect Container Edition.

- **TIBCO ActiveMatrix BusinessWorks**

  TIBCO ActiveMatrix BusinessWorks processes can send requests to BusinessConnect Container Edition or receive replies from BusinessConnect Container Edition by using the BusinessConnect palette resources in TIBCO Business Studio.

Standalone Private Processes

If you want to design standalone private processes for use with BusinessConnect Container Edition - ebXML Protocol, you must understand JMS message exchange. BusinessConnect Container Edition - ebXML Protocol uses TIBCO ActiveExchange™ messages to exchange ebXML messages with private processes. See Private Process Message Formats for formatting information of ebXML messages.

See Tutorial — Standalone Private Processes for an example of using BusinessConnect Container Edition - ebXML Protocol with standalone private processes.

TIBCO ActiveMatrix BusinessWorks Processes

The BusinessConnect palette, which is installed into TIBCO ActiveMatrix BusinessWorks during the TIBCO ActiveMatrix BusinessWorks™ Plug-in for BusinessConnect™ installation, encapsulates the message exchange configurations. You can also use the variety of TIBCO ActiveMatrix BusinessWorks palettes to define elaborate private processes to suit the needs of your business transactions.

For more information about using TIBCO ActiveMatrix BusinessWorks private processes, see Configuring Private Processes with TIBCO ActiveMatrix BusinessWorks.

See Tutorial — TIBCO ActiveMatrix BusinessWorks Private Processes for configuring TIBCO ActiveMatrix BusinessWorks private processes to send and receive messages with BusinessConnect Container Edition - ebXML Protocol.

Private Process Message Formats

BusinessConnect Container Edition - ebXML Protocol uses ActiveExchange™ messages to exchange ebXML messages with private processes through JMS. This section describes the ebXML-specific data classes that are used in the messages. You must format outbound messages and include all required information as prescribed by the data classes.

For JMS transport, it transmits message data in the message body of JMS ObjectMessage messages. See the "Private Processes" section in *TIBCO BusinessConnect™ Container Edition Concepts*.

Note: The JMS topics and queues that BusinessConnect Container Edition - ebXML Protocol uses do not contain the protocol name. Ensure that the `JMSType` attribute in the `header` element contains the string "ebXML" when exchanging JMS messages with BusinessConnect Container Edition - ebXML Protocol.

See the following sections for formatting details of the ebXML message type:

- Outbound Request Format
- Request Acknowledgment Format
- Inbound Response Format
- Inbound Request Format

- Outbound Response Format
- Response Acknowledgment Format
- Advisory Signal Message Format
- Error Message Format

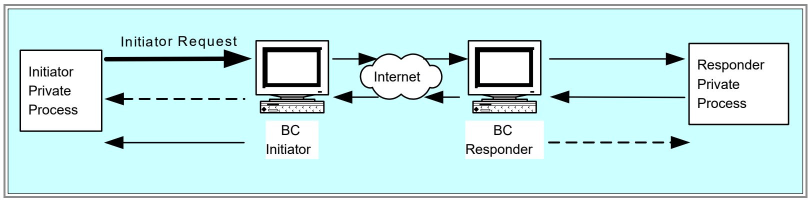
Data Types and Data Fields

See the following sections for formats of specific message fields and other data objects:

- Additional Data Classes
- Request and Response Fields

Outbound Request Format

The initiator private process uses the following message class to send outbound request or notify messages to BusinessConnect Container Edition - ebXML Protocol.



Class

`ae/ebXML/InitiatorRequest`

Example: `AX.BC.ACME_SERVER.ebXML.INITIATOR.REQUEST`

prefix.installation`.INITIATOR.REQUEST` (JMS queue)

Message fields

See the following table for a description of each message field.

ae/ebXML/InitiatorRequest Message Fields

Field	Type	Required	Description
standardID	String	Yes	The string "ebXML".
hostDomain	String	No	The domain name of the trading host. BusinessConnect Container Edition - ebXML Protocol uses the **hostDomain** field and the **hostID** field as follows: • If **hostDomain** and **hostID** together match a domain-identity pair of a trading host, and then that host is used along with the matching domain-identity pair. • If only **hostDomain** is specified, the first domain-identity pair of any trading host that has a matching domain value is used. • If neither **hostDomain** nor **hostID** is specified, or if a match cannot be found for any host in the configuration store, the default domain identity of the default trading host is used.
hostID	String	No	The identity value of the trading host domain-identity pair. See the description for **hostDomain** on how this field is used.
tpName	String	Yes	The name of the trading partner in this transaction, as defined in the Participants panel in the

Field	Type	Required	Description
			BusinessConnect Container Edition console.
tpDomain	String	No	The domain name of the trading partner. Specify a domain here if you want to use a domain identity other than the one configured in the **Default Domain Identity** field in the Participants panel.
operationID	String	No	The identifier for the transaction. Example: `rosettanet.org/3A4/2.0/Request Purchase Order`. This field is ignored when values are in all of the following fields: • **organization** • **businessProcess** • **businessProcessVersion** • **transactionName**
organization	String	No	The organization name for the transaction. Example: `rosettanet.org`. When a value is in this field, there must be values in the **businessProcess**, **businessProcessVersion**, and **transactionName** fields. When this combination does not exist, the **operationID** field is used.
businessProcess	String	No	The name of the business process as defined by the organization you use.

Field	Type	Required	Description
			Example: `3A4`. When a value is in this field, there must be values in the **organization**, **businessProcessVersion**, and **transactionName** fields. When this combination does not exist, the **operationID** field is used.
businessProcess Version	String	No	The business process version. Example: `2.0`. When a value is in this field, there must be values in the **organization**, **businessProcess**, and **transactionName** fields. When this combination does not exist, the **operationID** field is used.
transactionName	String	No	The name of the transaction. Example: `Request Purchage Order`. When a value is in this field, there must be values in the **organization**, **businessProcess**, and **businessProcessVersion** fields. When this combination does not exist, the **operationID** field is used.
cpaID	String	No	The value of the `<CPAId>` element in the SOAP header of the outbound message.
binaryRequest	base64Binary	No	Private processes use the **binaryRequest** field to send binary data to BusinessConnect Container Edition.

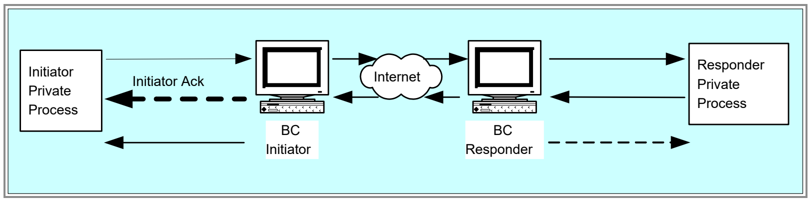
Field	Type	Required	Description
request	String	Yes	The request payload, which is usually an XML string. See Request and Response Fields for formatting information. **Note:** Text-based payloads and attachments must be encoded in UTF-8.
attachment	sequence of `ebXML/Info/Attachment`	No	A sequence of attachments. See ae/BC/Attachment Fields for formatting information.
conversationID	String	No	A unique identifier unique to this conversation. If you do not supply a value here, it is automatically generated for the public message.
messageID	String	No	A unique identifier for the outbound ebXML message. It must conform to the format described in RFC 2822. Otherwise, your trading partner cannot process the message correctly. If you do not supply a value here, it is automatically generated for the public message.
dupElimination	Boolean	No	Request that the trading partner (responder) eliminate duplicate messages. This field is `false` by default.
ackRequested	Boolean	No	Request for a receipt acknowledgment from the Message Service Handler of the trading

Field	Type	Required	Description
			partner (responder). This field is `false` by default.
ackSignature Requested	Boolean	No	Request that the Message Service Handler of the trading partner (responder) digitally sign the receipt acknowledgment. This field is `false` by default.
sequenced	Boolean	No	Specify `true` to indicate that messages in this conversation must be sequenced and include the `<MessageOrder>` element in the SOAP header. This field is ignored if you do not specify a value in the **conversationID** field. **Note:** If you specify `false` here when supplying the conversationID of an ongoing sequenced conversation in the **conversationID** field, and then this message cannot be processed correctly.
timeToLive	Integer	No	The maximum length of time for this message to be delivered. This field is used to calculate the value of the `<TimeToLive>` element in the SOAP header of the public message. The value in this field overrides the one in the **Lifetime of Message** field of **request action** tab in the Operations Editor panel.
closure	String	No	A unique string for BusinessConnect Container Edition - ebXML Protocol

Field	Type	Required	Description
			to keep track of this transaction. It is copied to all response messages of this request. If you do not supply a value here, it is automatically generated by BusinessConnect Container Edition - ebXML Protocol.
trackingID	String	No	An identifier that the private process can keep track of messages that are otherwise related to each other. It is copied to all response messages to this request. Contrary to the **closure** and **conversationID** fields, there is no requirement on the value of this field, nor is it used to process public messages.

Request Acknowledgment Format

BusinessConnect Container Edition - ebXML Protocol uses the following message class to indicate to the private process whether the request or notify message is sent to the trading partner successfully.



Class

```
ae/ebXML/InitiatorAck
```

Subject

Example: `AX.BC.ACME_SERVER.ebXML.INITIATOR.ACK`

prefix.installation`.INITIATOR.ACK` (JMS queue)

Message fields

See the following table for a description of each message field.

ae/ebXML/InitiatorAck Message Fields

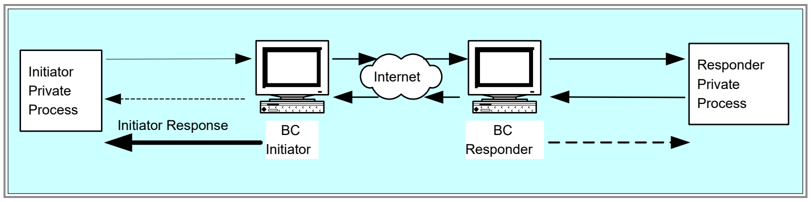
	Type	Required	Description
standardID	String	Yes	The string "ebXML".
tpName	String	Yes	The name of the trading partner, as specified in the **tpName** field in the corresponding `ae/ebXML/InitiatorRequest` message.
tradingPartner	trading partner	Yes	The trading partner. See ae/BC/TradingPartner Fields for formatting information.
host	trading partner	Yes	The trading host. See ae/BC/TradingPartner Fields for formatting information.
operationID	String	Yes	The identifier of the transaction. For example: `rosettanet.org/3A4/2.0/Request Purchase Order`.
organization	String	Yes	The organization name for the transaction.
businessProcess	String	Yes	The name of the business process for the transaction.

	Type	Required	Description
businessProcessVersion	String	Yes	The business process version for the transaction.
transactionName	String	Yes	The name of the transaction.
cpaID	String	No	The value of the `<CPAId>` element in the SOAP header of the public message. This is configured in the **Specification ID** field of the Participants panel for the trading partner. See Setting Up Trading Hosts and Partners.
conversationID	String	Yes	The value of the `<ConversationId>` element in the SOAP header of the public message.
messageID	String	Yes	The value of the `MessageData/MessageId` element in the SOAP header of the public message.
sequenceNo	Integer	No	The value of the `MessageOrder/SequenceNumber` element in the SOAP header of the public message. This field is present only for sequenced messages.
closure	String	No	A unique string for BusinessConnect Container Edition - ebXML Protocol to keep track of this transaction. It is either generated by BusinessConnect Container Edition - ebXML Protocol or copied from the **closure** field in the corresponding `ae/ebXML/InitiatorRequest` message.

	Type	Required	Description
trackingID	String	No	The value of the **trackingID** field in the corresponding `ae/ebXML/InitiatorRequest` message, which can be used by the private process to keep track of messages that are otherwise related to each other.

Inbound Response Format

BusinessConnect Container Edition - ebXML Protocol uses the following message class to forward inbound response messages to the initiator private process.



Class

`ae/ebXML/InitiatorResponse`

Subject

Example: `AX.BC.ACME_SERVER.ebXML.INITIATOR.RESPONSE`

prefix.installation`.INITIATOR.RESPONSE` (JMS queue)

Message fields

See the following table for a description of each message field.

ae/ebXML/InitiatorResponse Message Fields

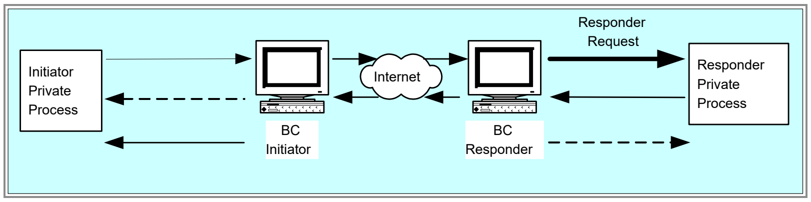
	Type	Required	Description
standardID	String	Yes	The string "ebXML".
fromTP	trading partner	Yes	The trading partner. See ae/BC/TradingPartner Fields for formatting information.
host	trading partner	Yes	The trading host. See ae/BC/TradingPartner Fields for formatting information.
fromRole	String	No	The value of the `From/Role` element in the SOAP header of the inbound public message.
toRole	String	No	The value of the `To/Role` element in the SOAP header of the inbound public message.
operationID	String	Yes	The identifier of the transaction. For example: `rosettanet.org/3A4/2.0/Request Purchase Order`.
organization	String	Yes	The organization name for the transaction.
businessProcess	String	Yes	The name of the business process for the transaction.
businessProcess Version	String	Yes	The business process version for the transaction.
transactionName	String	Yes	The name of the transaction.
statusCode	Integer	Yes	A code indicating the status of the transaction. See Status Codes.

	Type	Required	Description
statusMsg	String	Yes	A brief description of the code in the **statusCode** field.
binaryResponse	base64Binary	No	Response is sent to this field if the message contains binary data.
response	String	No	The response payload, which is usually an XML string. See Request and Response Fields for formatting information.
attachment	sequence of `ebXML/Info/Attachment`	No	A sequence of attachments. See ae/BC/Attachment Fields for formatting information.
cpaID	String	No	The value of the `<CPAId>` element in the SOAP header of the inbound public message.
conversationID	String	Yes	The value of the `<ConversationId>` element in the SOAP header of the inbound public message.
messageID	String	Yes	The value of the `MessageData/MessageId` element in the SOAP header of the inbound public message.
sequenceNo	Integer	No	The value of the `MessageOrder/SequenceNumber` element in the SOAP header of the inbound public message.
businessDocument Name	String	No	The name of the root XML element in the **response** field. It is copied from the

	Type	Required	Description
			`Manifest/Reference/xlink:role` attribute in the SOAP header of the inbound public message. **Note:** This field is not included if the document name cannot be extracted from the previous attribute.
timeToLive	Integer	No	The maximum length of time for the response to be processed. It is calculated by subtracting the value of the `<TimeToLive>` element in the inbound public message from the time the private message is delivered.
closure	String	No	A unique string for BusinessConnect Container Edition - ebXML Protocol to keep track of this transaction. It is either generated by BusinessConnect Container Edition - ebXML Protocol or copied from the **closure** field in the corresponding `ae/ebXML/InitiatorRequest` message.
trackingID	String	No	The value of the **trackingID** field in the corresponding `ae/ebXML/InitiatorRequest` message, which can be used by the private process to keep track of messages that are otherwise related to each other.
messageHeaders	String	No	The HTTP headers of the received message displayed in their general format.

Inbound Request Format

BusinessConnect Container Edition - ebXML Protocol uses the following message class to forward inbound request or notify messages to the responder private process.



Class

`ae/ebXML/ResponderRequest`

Subject

Example: `AX.BC.ACME_SERVER.ebXML.RESPONDER.REQUEST`

prefix.installation`.RESPONDER.REQUEST` (JMS queue)

Message fields

See the following table for description of each message field.

ae/ebXML/ResponderRequest Message Fields

Field	Type	Required	Description
standardID	String	Yes	The string "ebXML".
fromTP	trading partner	No	The trading partner. See ae/BC/TradingPartner Fields for formatting information.
host	trading partner	Yes	The trading host. See ae/BC/TradingPartner Fields for formatting information.
fromRole	String	No	The role of the trading partner,

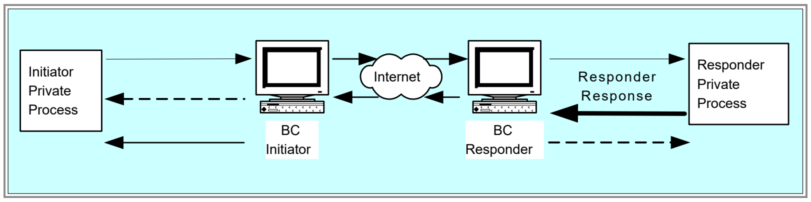
Field	Type	Required	Description
			which is copied from the `From/Role` element in the SOAP header of the inbound public message.
toRole	String	No	The role of the trading host, which is copied from the `To/Role` element in the SOAP header of the inbound public message.
operationID	String	Yes	The identifier of the transaction. For example: `rosettanet.org/3A4/2.0/Request Purchase Order`.
organization	String	Yes	The organization name for the transaction.
businessProcess	String	Yes	The name of the business process for the transaction.
businessProcess Version	String	Yes	The business process version for the transaction.
transactionName	String	Yes	The name of the transaction.
service	String	Yes	The service that acts on the message, which is copied from the `<Service>` element in the SOAP header of the inbound public message.
serviceType	String	No	The service type, which is copied from the `Service/type` attribute in the SOAP header of the inbound public message.
action	String	Yes	The action used to process the

Field	Type	Required	Description
			inbound message, which is copied from the `<Action>` element in the SOAP header of the inbound public message.
binaryRequest	base64Binary	No	Used by BusinessConnect Container Edition to send data when binary data is being passed to a partner.
request	String	No	The request payload, which is usually an XML string. See Request and Response Fields for formatting information.
attachment	sequence of `ebXML/Info/Attachment`	No	A sequence of attachments. See ae/BC/Attachment Fields for formatting information.
cpaID	String	No	The identifier of the CPA, copied from the `<CPAId>` element in the SOAP header of the inbound public message.
conversationID	String	Yes	The identifier for the current conversation, copied from the `<ConversationId>` element in the SOAP header of the inbound public message.
messageID	String	Yes	The unique identifier for the inbound request, copied from the `MessageData/MessageId` element in the SOAP header of the inbound public message.
sequenceNo	Integer	No	The sequence number of the request,

Field	Type	Required	Description
			copied from the `MessageOrder/SequenceNumber` element in the SOAP header of the inbound public message.
timeToLive	Integer	No	The maximum length of time for the request to be processed. It is calculated by subtracting the value of the `<TimeToLive>` element in the inbound public message from the time the private message is delivered.
closure	String	Yes	A unique string generated by BusinessConnect Container Edition - ebXML Protocol to keep track of this transaction. You must copy the value here to the **closure** field in the corresponding `ae/ebXML/ResponderResponse` message.
messageHeaders	String	No	The HTTP headers of the received message displayed in their general format.

Outbound Response Format

The responder private process uses the following message class to send outbound response messages and business-level exceptions to BusinessConnect Container Edition - ebXML Protocol.



Class

`ae/ebXML/ResponderResponse`

Subject name

Example: `AX.BC.ACME_SERVER.ebXML.RESPONDER.RESPONSE`

prefix.installation`.RESPONDER.RESPONSE` (JMS queue)

Message fields

See the following table for a description of each message field.

ae/ebXML/ResponderResponse Message Fields

Field	Type	Required	Description
standardID	String	Yes	The string "ebXML".
statusCode	Integer	Yes	The status of this transaction. Specify one of the following codes: • **200** The request is processed successfully. • **3652** The request is processed successfully but rejected. In this case, BusinessConnect Container Edition - ebXML Protocol sends a response message to the trading partner with `AcceptanceException` as the value of the `<ExceptionType>` element. • **3650** The request is not processed successfully. In this case, BusinessConnect Container Edition - ebXML Protocol sends a response message to the trading partner

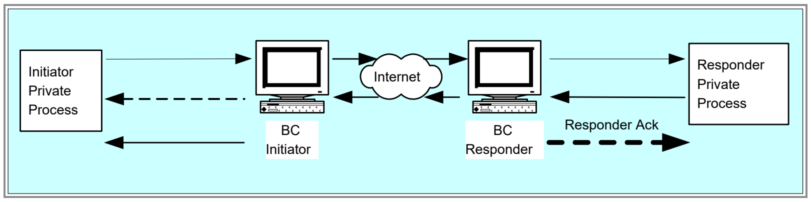
Field	Type	Required	Description
			with `GeneralException` as the value of the `<ExceptionType>` element. See Business-Level Signals.
statusMsg	String	Yes	A description of the code in the **statusCode** field. See Status Codes.
response	String	No	The response payload, which is usually an XML string. See Request and Response Fields for formatting information. **Note:** Text-based payloads and attachments must be encoded in UTF-8.
binaryResponse	base64Binary	No	Response is sent to this field if the message contains binary data.
attachment	sequence of `ebXML/Info/Attachment`	No	A sequence of attachments. See ae/BC/Attachment Fields for formatting information.
messageID	String	No	A unique identifier for the outbound ebXML message. It must conform to the format described in RFC 2822. Otherwise, your trading partner cannot process the message correctly. If you do not supply a value here, it is automatically generated for the public message.
closure	String	No	A unique string for BusinessConnect Container Edition - ebXML Protocol to

Field	Type	Required	Description
			keep track of this transaction. You must copy the value from the **closure** field in the corresponding `ae/ebXML/ResponderRequest` message.
businessDocumentName	String	No	The name of the root XML element in the **response** field.
dupElimination	Boolean	No	Request that the trading partner (initiator) eliminate duplicate messages. This field is `false` by default.
ackRequested	Boolean	No	Request for a receipt acknowledgment from the Message Service Handler of the trading partner (initiator). This field is `false` by default.
ackSignatureRequested	Boolean	No	Request that the Message Service Handler of the trading partner (initiator) digitally sign the receipt acknowledgment. This field is `false` by default.
timeToLive	Integer	No	The maximum length of time for this response to be delivered. This field is used to calculate the value of the `<TimeToLive>` element in the SOAP header of the public message. The value in this field overrides the one in the **Lifetime of Message** field of the **Response Action** tab in the Operations Editor panel.

Field	Type	Required	Description
trackingID	String	No	An identifier that the private process can keep track of messages that are otherwise related to each other. It is copied to the corresponding `ae/ebXML/ResponderAck` message. Contrary to the **closure** and **conversationID** fields, there is no requirement on the value of this field, nor is it used to process public messages.

Response Acknowledgment Format

BusinessConnect Container Edition - ebXML Protocol uses the following message class to indicate to the private process whether the response message is sent to the trading partner successfully.



Class

`ae/ebXML/ResponderAck`

Subject

Example: `AX.BC.ACME_SERVER.ebXML.RESPONDER.ACK`

prefix.installation`.RESPONDER.ACK` (JMS queue)

Message fields

See the following table for a description of each message field.

ae/ebXML/ResponderAck Message Fields

	Type	Required	Description
standardID	String	Yes	The string "ebXML".
tradingPartner	trading partner	Yes	The trading partner. See ae/BC/TradingPartner Fields for formatting information.
host	trading partner	Yes	The trading host. See ae/BC/TradingPartner Fields for formatting information.
statusCode	Integer	Yes	A code indicating the status of the transaction. See Status Codes.
statusMsg	String	Yes	A brief description of the code in the **statusCode** field.
operationType	String	No	The identifier of the transaction. Example: `rosettanet.org/3A4/2.0/Request Purchase Order`.
closure	String	Yes	A unique string for BusinessConnect Container Edition - ebXML Protocol to keep track of this transaction. It is copied from the **closure** field in the corresponding `ae/ebXML/ResponderRequest` message.
operationID	String	Yes	The identifier of the transaction. For example: `rosettanet.org/3A4/2.0/Request Purchase Order`.
organization	String	Yes	The organization name for the transaction.
businessProcess	String	Yes	The name of the business process for the transaction.

	Type	Required	Description
businessProcessVersion	String	Yes	The business process version for the transaction.
transactionName	String	Yes	The name of the transaction.
cpaID	String	No	The value of the `<CPAId>` element in the SOAP header of the public message. This is configured in the **Specification ID** field of the Participants panel for the trading partner. See Setting Up Trading Hosts and Partners.
conversationID	String	Yes	The value of the `<ConversationId>` element in the SOAP header of the public message.
messageID	String	No	The value of the `MessageData/MessageId` element in the SOAP header of the public message.
sequenceNo	Integer	No	The value of the `MessageOrder/SequenceNumber` element in the SOAP header of the public message. This field is present only for sequenced messages.
trackingID	String	No	The value of the **trackingID** field in the corresponding `ae/ebXML/ResponderResponse` message, which can be used by the private process to keep track of messages that are otherwise related to each other.

Advisory Signal Message Format

BusinessConnect Container Edition - ebXML Protocol uses the `ae/ebXML/Advisory` class to update private processes on the status of ebXML transactions. BusinessConnect Container

Edition - ebXML Protocol publishes `ae/ebXML/Advisory` messages on the advisory signals subject when it receives business-level signals or MSH acknowledgments from the trading partner.

For more information about business-level signals, see Processing Business-Level Signals.

For more information about the MSH acknowledgment, see Delivery Semantics.

The field names map to (but are not the same as) column names in the audit database, and their values match the values that are written to the audit database.

Note: The `ae/ebXML/Advisory` message class is used for both advisory and error signals, but they are sent to separate private message subjects, and some of the message fields can hold different data for advisory and error signals.

Class

`ae/ebXML/Advisory`

Subject

Example: `AX.BC.ACME_SERVER.ebXML.ADVISORY.SIGNALS`

prefix.installation`.ADVISORY.SIGNALS` (JMS topic)

Message fields

See the following table for description of each message field.

ae/ebXML/Advisory Signal Fields

	Type	Required	Description
standardID	String	Yes	The string "ebXML".
tpName	String	No	The name of the trading partner in this transaction, as defined in the Participants panel in the BusinessConnect Container Edition console.

	Type	Required	Description
tradingPartner	trading partner	Yes	The trading partner. See ae/BC/TradingPartner Fields for formatting information.
host	trading partner	Yes	The trading host. See ae/BC/TradingPartner Fields for formatting information.
operationID	String	No	The identifier of the transaction. Example: `rosettanet.org/3A4/2.0/Request Purchase Order`.
transactionID	String	No	Currently not used.
statusCode	Integer	No	The status of this transaction. See the following possible codes: • **1651** Received a receipt acknowledgment from the trading partner. • **1653** Received an MSH acknowledgment from the trading partner. • **3650** Received an exception message from the trading partner with an `ExceptionType/GeneralException` element in the message payload, indicating that the private process of the trading partner failed to process the request or response. • **3652** Received an exception message from the trading partner with an `ExceptionType/AcceptanceException` element in the message payload, indicating that the private process of the trading partner rejected the request or response.

	Type	Required	Description
			See Status Codes.
statusMsg	String	No	A brief description of the code in the **statusCode** field.
details	String	No	A more detailed description of the transaction status. This can be an analysis of the error that occurred.
extraInfo	String	No	The XML-formatted signal message that is received.
closure	String	No	A unique string for BusinessConnect Container Edition - ebXML Protocol to keep track of this transaction. It is copied from the **closure** field in the corresponding `ae/ebXML/InitiatorRequest` or `ae/ebXML/ResponderRequest` message.
organization	String	No	The organization name for the transaction.
businessProcess	String	No	The name of the business process for the transaction.
businessProcess Version	String	No	The business process version for the transaction.
transactionName	String	No	The name of the transaction.
service	String	No	The service that actions on the message, which is copied from the `<Service>` element in the SOAP header of the public message.
serviceType	String	No	The service type, which is copied from the `Service/type` attribute in the SOAP header of the public message.

	Type	Required	Description
action	String	No	The action used to process the inbound message, which is copied from the `<Action>` element in the SOAP header of the public message. If the value of the `<Action>` element is `Exception`, and then the value here is `GeneralException` or `AcceptanceException`, depending on the kind of exception.
cpaID	String	No	The value of the `<CPAId>` element in the SOAP header of the public message. This is configured in the **Specification ID** field of the Participants panel for the trading partner. See Setting Up Trading Hosts and Partners.
conversationID	String	No	The value of the `<ConversationId>` element in the SOAP header of the public message.
messageID	String	No	The value of the `MessageData/MessageId` element in the SOAP header of the public message.
sequenceNo	Integer	No	The value of the `MessageOrder/SequenceNumber` element in the SOAP header of the public message. This field is present only for sequenced messages.
trackingID	String	No	The value of the **trackingID** field in the corresponding `ae/ebXML/InitiatorRequest` or `ae/ebXML/ResponderResponse` message, which can be used by the private process to keep track of messages that are otherwise related to each other.
messageHeaders	String	No	The HTTP headers of the received message displayed in their general format.

Error Message Format

BusinessConnect Container Edition - ebXML Protocol also uses the `ae/ebXML/Advisory` class to send transaction errors to private processes.

The field names map to (but are not the same as) column names in the audit database, and their values match the values that are written to the audit database.

> **Note:** The `ae/ebXML/Advisory` message class is used for both advisory and error signals, but they are sent to separate private message subjects, and some of the message fields might hold different data for advisory and error signals.

Class

`ae/ebXML/Advisory`

Subject

Example: `AX.BC.ACME_SERVER.ebXML.ERROR`

prefix.installation`.ERROR` (JMS topic)

Message fields

See the following table for a description of each message field.

ae/ebXML/Advisory Error Fields

	Type	Required	Description
standardID	String	Yes	The string "ebXML".
tpName	String	No	The name of the trading partner in this transaction defined in the Participants panel in the BusinessConnect Container Edition console.
tradingPartner	trading partner	Yes	The trading partner. See ae/BC/TradingPartner Fields for formatting

	Type	Required	Description
			information.
host	trading partner	Yes	The trading host. See ae/BC/TradingPartner Fields for formatting information.
operationID	String	No	The identifier of the transaction. For example: `rosettanet.org/3A4/2.0/Request Purchase Order`.
transactionID	String	No	N/A
statusCode	Integer	No	A code indicating the status of the transaction. See Status Codes.
statusMsg	String	No	A brief description of the code in the **statusCode** field.
details	String	No	A more detailed description of the transaction status. This can be an analysis of an error that occurred.
extraInfo	String	No	N/A
closure	String	No	A unique string for BusinessConnect Container Edition - ebXML Protocol to keep track of this transaction. It is copied from the **closure** field in the corresponding `ae/ebXML/InitiatorRequest` or `ae/ebXML/ResponderRequest` message.
organization	String	No	The organization name for the transaction.
businessProcess	String	No	The name of the business process for the transaction.

	Type	Required	Description
businessProcess Version	String	No	The business process version for the transaction.
transactionName	String	No	The name of the transaction.
service	String	No	The service that actions on the message, which is copied from the `<Service>` element in the SOAP header of the public message.
serviceType	String	No	The service type, which is copied from the `Service/type` attribute in the SOAP header of the public message.
action	String	No	The action used to process the inbound message, which is copied from the `<Action>` element in the SOAP header of the public message.
cpaID	String	No	The value of the `<CPAId>` element in the SOAP header of the public message. This is configured in the **Specification ID** field of the Participants panel for the trading partner. See Setting Up Trading Hosts and Partners.
conversationID	String	No	The value of the `<ConversationId>` element in the SOAP header of the public message.
messageID	String	No	The value of the `MessageData/MessageId` element in the SOAP header of the public message.
sequenceNo	Integer	No	The value of the `MessageOrder/SequenceNumber` element in the SOAP header of the public message.

	Type	Required	Description
			This field is present only for sequenced messages.
trackingID	String	No	The value of the **trackingID** field in the corresponding `ae/ebXML/InitiatorRequest` or `ae/ebXML/ResponderResponse` message, which can be used by the private process to keep track of messages that are otherwise related to each other.
messageHeaders	String	No	The HTTP headers of the received message displayed in their general format.

Additional Data Classes

Two additional data classes are commonly used by some of the private process message classes:

- `ae/BC/Attachment`. See the following table for a description of each field.

ae/BC/Attachment Fields

Field	Type	Description
name	String	The name of the attached data.
content-type	String	The attached data type. For example: `application/binary`.
content-id	String	A unique identifier for the attachment. For outbound attachments with an empty **name** field and inbound attachments, BusinessConnect Container Edition - ebXML Protocol automatically generates a random string for the content-id element of an attachment.
content	Binary	The content of the attachment. This field takes precedence

Field	Type	Description
		over the **fileReference** field. **Note:** All text-based attachments must be encoded in UTF-8.
fileReference	String	The absolute path to the file that contains the attachment. This field is ignored if the **content** field is not empty. For inbound messages, BusinessConnect Container Edition - ebXML Protocol uses this field to pass the reference of a large attachment.

- `ae/BC/TradingPartner`. See the following table for a description of each field.

ae/BC/TradingPartner Fields

Message Field	Type	Description
name	String	The name of the trading participant.
domain	String	The domain of the trading participant.
id	String	The identity of the trading participant.

Request and Response Fields

The **request** or **response** field in a private process message contains the XML-formatted business document, without the SOAP header and attachments.

The content in the **request** or **response** field is eventually composed with other information by BusinessConnect Container Edition - ebXML Protocol into a valid ebXML document, which conforms to ebXML Message Service. It contains an XML declaration, followed by a request or response document as the single root element. The example is a RosettaNet PIP 3A4 Purchase Order Request document in the **request** field:

```
<?xml version="1.0">
<Pip3A4PurchaseOrderRequest>
```

```
...
</Pip3A4PurchaseOrderRequest>
```

Configuring Private Processes with TIBCO ActiveMatrix BusinessWorks

To define a TIBCO ActiveMatrix BusinessWorks private process to interact with BusinessConnect Container Edition requires the following resources:

- BusinessConnect Connection is a shared resource that connects to the BusinessConnect Container Edition configuration store for the settings in BusinessConnect Container Edition - ebXML Protocol.
- Activities of TIBCO ActiveMatrix BusinessWorks™ Plug-in for BusinessConnect™ are used to handle outbound and inbound BusinessConnect Container Edition messages.

Configuring BCServerConfig Resource

For a private process to communicate with your BusinessConnect Container Edition server, you must include exactly one BCServerConfig resource in the root project view.

To configure the BCServerConfig resource, perform the following steps:

Procedure

1. Start TIBCO Business Studio.
2. Go to **File** > **Import**.
3. On the Import page, expand the **General** folder and select **Existing Studio Projects into Workspace**. Click **Next**.
4. Click **Browse** next to the **Select archive file** field to navigate to the `bcce-ebxml-1.0.0/samples/ebMS2/tutorial/bw` directory, and select the `ebXMLDemo_for_bw6.zip` file. Click **Open** and then click **Finish**.
5. Expand **ebXML** > **Module Descriptors** in the Project Explorer view.
6. Double-click **Module Properties**.

7. Change the default value of the **BCHOME** property according to your environment.
8. In the Project Explorer view, expand **Resources** and double-click **BCConnection.bcResource**.
9. Click the **Server Access** tab.
10. Specify the values in the **JDBC Driver**, **JDBC URL**, **DB User**, and **DB Password** fields.

    > ℹ **Note:** JDBC settings must be the same as the settings in your BusinessConnect Container Edition installation. To see the JDBC settings, review the `deployment.properties` file of BusinessConnect Container Edition.

11. Click the **Configuration** tab, and click **Update from Configuration Store**.
12. Select **ebXML** from the **Protocol Name** list.
13. Click **Import Selected Business Protocol**.

    When you import the protocol, TIBCO ActiveMatrix BusinessWorks retrieves information from the BusinessConnect Container Edition configuration store and puts it in the project folder.

    > ⚠ **Warning:** Do not move, modify, or delete the BCSchemas folder or the contents therein.

14. Click **Save**.

    For more information on the BusinessConnect Container Edition Connection resource, see the *TIBCO ActiveMatrix BusinessWorks™ Plug-in for BusinessConnect™* documentation.

Configuring Send and Receive Activities

You can use BusinessConnect Container Edition Send activities for sending requests or receipts, responses, and miscellaneous messages. You can also use the BusinessConnect Container Edition Receive activities for receiving requests or receipts, response, and miscellaneous messages.

When configuring a Send activity, you must ensure that all required fields are supplied to the activity before the Send activity can be started. When a Send activity is started, the resource sends a request of the selected operation to BusinessConnect Container Edition.

Unlike Send activities, Receive activities are *process starters*. Therefore, there can be only one Receive activity in a process and the Receive activity acts to start the process. A process with the Receive start activity is invoked when BusinessConnect Container Edition receives a message from a trading partner and sends it to TIBCO ActiveMatrix BusinessWorks. You can configure the process to use the output of the Receive activity to perform different tasks, such as formulating a response message and invoking a Send activity.

For more information on configuring each of the Send and Receive activities, see TIBCO ActiveMatrix BusinessWorks™ Plug-in for BusinessConnect™. You can also step through the Tutorial — TIBCO ActiveMatrix BusinessWorks Private Processes to learn how to configure a TIBCO ActiveMatrix BusinessWorks project to send and receive ebXML messages.

The following subsections give you a quick glance at the resources you must configure different kinds of private processes.

Configuring Initiators

For your TIBCO ActiveMatrix BusinessWorks processes to initiate request-response or notify transactions, use the following resources:

- BCServerConfig — for connecting to your BusinessConnect Container Edition server.
- Send Request/Notification — for sending request or notify messages to trading partners.
- Receive Response — for receiving responses from trading partners. This is only for request-response.
- Receive Misc. Msg — for receiving acknowledgments, advisory signals, or error signals from BusinessConnect Container Edition.

Configuring Responders

For your TIBCO ActiveMatrix BusinessWorks processes to receive messages, use the following resources:

- BCServerConfig — for connecting to your BusinessConnect Container Edition server.
- Receive Request/Notification — for receiving request messages from trading partners.

- Send Response — for sending response messages to BusinessConnect Container Edition.
- Receive Misc. Msg — for receiving acknowledgments, advisory signals, or error signals from BusinessConnect Container Edition.

Configuring Notification Receivers

For your TIBCO ActiveMatrix BusinessWorks processes to receive notify messages, use the following resources:

- BCServerConfig — for connecting to your BusinessConnect Container Edition server.
- Receive Request/Notification — for receiving notify messages from trading partners.
- Receive Misc. Msg — for receiving advisory and error signals from BusinessConnect Container Edition.

Viewing Logs

This section explains how to view the transaction logs in Audit Trail of BusinessConnect Container Edition - ebXML Protocol.

TIBCO® AuditSafe stores information about the messages and documents processed by BusinessConnect Container Edition.

To view the logs, on the Data Viewer tile, click **Audit Trail** > **ebXML**.

All the transactions are displayed on Transactions Screen.

Transactions Screen

From the Transactions screen, you can do the following:

- **Sort** transactions
- **Search Transactions** by using filters
- View **Transaction Details**
- Change the columns that you view by selecting the categories from the **Settings** dialog.

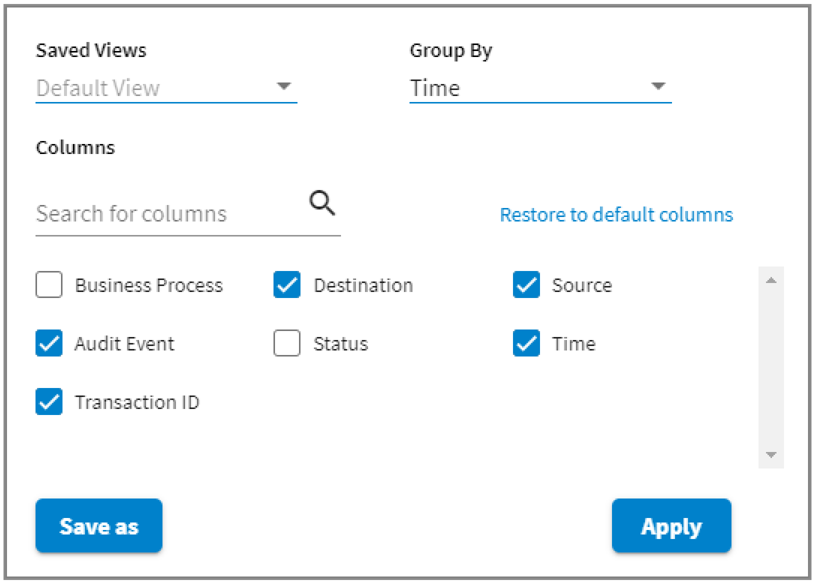
Customizing Transactions Screen

Procedure

1. Click the **Gear** ⚙ icon.
2. In the **Settings** dialog:
   - Select the checkboxes for the columns that you want to view on the Transaction screen.
   - Clear the checkboxes for the columns that you do not want to view on the Transaction screen.

3. Click **Apply** to save your transaction screen settings.

Figure 17: Settings



Saved Views: Default View

Group By: Time

Columns

Search for columns

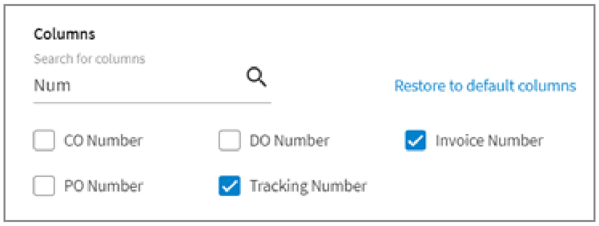
Restore to default columns

☐ Business Process ☑ Destination ☑ Source

☑ Audit Event ☐ Status ☑ Time

☑ Transaction ID

4. You can also use the following options:

Option	Description
Group By	Use this option to group results by category. To do this, select a category from the dropdown list.
Columns	Check the box next to the category to display the columns you want to view. You can also search for a category by entering a partial word to display all categories containing that string as shown in the following example.  Columns Search for columns: Num Restore to default columns ☐ CO Number ☐ DO Number ☑ Invoice Number ☐ PO Number ☑ Tracking Number

5. Each page displays up to `20` records with the newest transactions listed first as default. Page through the display to see the older records. You can sort the

transactions by category, or search for records of a specific type. The number of records returned are limited to `10,000` audit events or a maximum of `500` pages.

> **Note:** A "`No records found`" error is generated if your selection exceeds `500` pages.

Sort Transactions

Click the arrows on a column (category) heading to sort transactions.

- ↑ : Sort first to last.
- ↓ : Sort last to first.

To reorder the columns, click the category name to select it, and drag it to a different location on the page.

Transactions are sorted according to date and time by default, so the newest transactions appear at the top of the list.

Search for Transactions

The search feature of TIBCO AuditSafe is powerful and easy-to-use.

You can search for values in multiple categories and find matching records even if a category (or column) is not displayed.

While searching for values, use the following options:

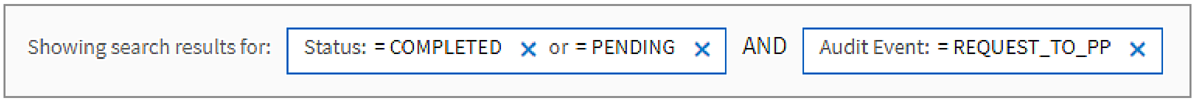
- **Keyword**: To search for a term in all categories.
- **Audit Event**: To search for an audit event in all categories.
- **OR**: To search for values or keywords within the same category.

  Example: Search **Status** for in `PROCESSING` **OR** `COMPLETED` to find all the transactions that contain those values.
- **AND**: To search for values between two or more categories.

  Example: Search **Status** for `COMPLETED` **OR** `PENDING`, **AND** search **Audit Event** for

`Request_TO_PP`.

The system automatically segments your query into **OR** and **AND** searches as shown in the following example:



Showing search results for: Status: = COMPLETED ✕ or = PENDING ✕ AND Audit Event: = REQUEST_TO_PP ✕

Using Search

Search contains the following options:

- **Time**: To define the start and end range for your search.
- **Keyword**: To search for the term in all categories.

Search Option	Description
=	**Equals**: Finds an exact match for a term. Entering the first few letters list matching options. Select one and press ENTER. If an exact match is not available, no items are listed.
!=	**Not Equal to**: Use this option to exclude terms from your search.
~	**Like**: Enter a few letters to list items containing a matching string.

1. Click the **Select** dropdown to choose a category.
2. In the **Search Transaction** field, select a Search Option , and enter a word.
3. Select an option from the ones that are displayed when you enter a word, and add it to your query.
4. Continue adding options to narrow your search results.
5. After creating your query, click **Find** to list the results.
6. Click the **Remove** icon ✕ to remove a search term.

Transaction Details

Click a transaction in the list to see more details.

You can view the transactions using either a Diagram view or a List view.

Switch between the two views by clicking the icon on the top right.

The diagram view also displays related transactions.

Tutorial — TIBCO ActiveMatrix BusinessWorks Private Processes

This section provides an overview of how to use TIBCO ActiveMatrix BusinessWorks with BusinessConnect Container Edition - ebXML Protocol.

You can use the BusinessConnect palette in TIBCO Business Studio to integrate the processes in TIBCO ActiveMatrix BusinessWorks with BusinessConnect Container Edition. This tutorial demonstrates the configuration of TIBCO ActiveMatrix BusinessWorks private processes by using the BusinessConnect palette in TIBCO Business Studio.

For more information on each resource in the BusinessConnect palette, see the *TIBCO ActiveMatrix BusinessWorks™ Plug-in for BusinessConnect™ User Guide*.

You can find the TIBCO ActiveMatrix BusinessWorks project files in the `bcce-ebxml-1.0.0/samples/ebMS2/tutorial/interfaces` directory. This example project demonstrates how to send and receive messages from BusinessConnect Container Edition - ebXML Protocol with TIBCO ActiveMatrix BusinessWorks private processes.

The following sample file is included in this directory:

- The `ebXMLDemo_for_bw6.zip` file is used with TIBCO ActiveMatrix BusinessWorks 6.

Prerequisites

Before starting the tutorial, install the following TIBCO software packages:

1. TIBCO BusinessConnect™ Container Edition

   For more information on setting up and running TIBCO BusinessConnect™ Container Edition, see the *TIBCO BusinessConnect™ Container Edition Administration* and *TIBCO BusinessConnect™ Container Edition Trading Partner Management* guides.

2. TIBCO ActiveMatrix BusinessWorks.
3. TIBCO ActiveMatrix BusinessWorks™ Plug-in for BusinessConnect™
4. TIBCO BusinessConnect™ Container Edition - ebXML Protocol

Setting Up the Tutorial

Completed the tasks in Setting Up Buyer on Machine 1 and Setting Up Seller on Machine 2. Use the same setup on two machines to run the tutorial.

Configuring Private Processes

To configure the tutorial in TIBCO Business Studio, complete the following tasks:

Opening the TIBCO ActiveMatrix BusinessWorks Project

To set up the example project on the initiator and the responder machines, perform the following steps:

Procedure

1. Start TIBCO Business Studio.
2. Go to **File** > **Import**.
3. On the Import page, expand the **General** folder and select **Existing Studio Projects into Workspace**. Click **Next**.
4. Click **Browse** next to the **Select archive file** field to navigate to the `bcce-ebxml-1.0.0/samples/ebMS2/tutorial/interfaces` directory.
5. Select the `ebXMLDemo_for_bw6.zip` file and click **Open**.
6. Click **Finish**.
7. Expand **ebXMLDemo** > **Module Descriptors** in the Project Explorer view.
8. Double-click **Module Properties**.
9. Change the default value of the **BCHOME** property according to your environment.

Configuring Connections to BusinessConnect Container Edition

To configure connections to BusinessConnect Container Edition, perform the following steps:

Procedure

1. In the Project Explorer view, expand **Resources** and double-click **BCConnection.bcResource**.
2. Click the **Server Access** tab.
3. Specify the values in the **JDBC Driver Type**, **JDBC Driver**, **JDBC URL**, **DB User**, and **DB Password** fields.

   Note: JDBC settings must be the same as the settings in your BusinessConnect Container Edition installation. To see the JDBC settings, review the `deployment.properties` file of BusinessConnect Container Edition.

4. Click the **Configuration** tab, and click **Update from Configuration Store**.
5. Select **ebXML** from the **Protocol Name** list.

   If you select the **Select Operations** checkbox, you can select any of the configured and imported operations. For this tutorial, select all operations and click **OK**.

6. Click **Import Selected Business Protocol**.

   When you import the protocol, TIBCO ActiveMatrix BusinessWorks retrieves information from the BusinessConnect Container Edition configuration store and puts it in the project folder.

7. Click **Save**.

Running the Tutorial

The following sample transactions are described in this tutorial:

- Send Request
- Send Response

Creating a Process Invoker

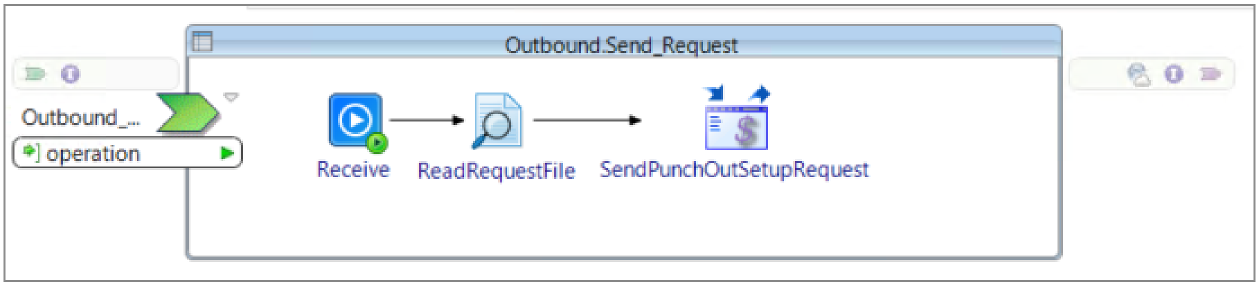
Procedure

1. In the Project Explorer view, expand **ebXMLDemo** > **Processes**.

2. Right-click on Processes, from the context menu, select **New** > **BusinessWorks Process**.
3. In the BusinessWorks Process Creation dialog, in the **Process Name** field, enter `InvokerProcess` and click **Finish**.
4. From the **Palette Library** > **General Activities**, add the **Timer** activity to the process.
5. From the **Palette Library** > **Basic Activities**, add the **Invoke** activity to the process.
6. Drag the icon to create a transition between the added activities.
7. Select the Invoke activity on the **Properties** tab, click the ✚ icon next to the **Service** field.
8. From the Select a Service dialog, select the process that you want to invoke and click **OK**.

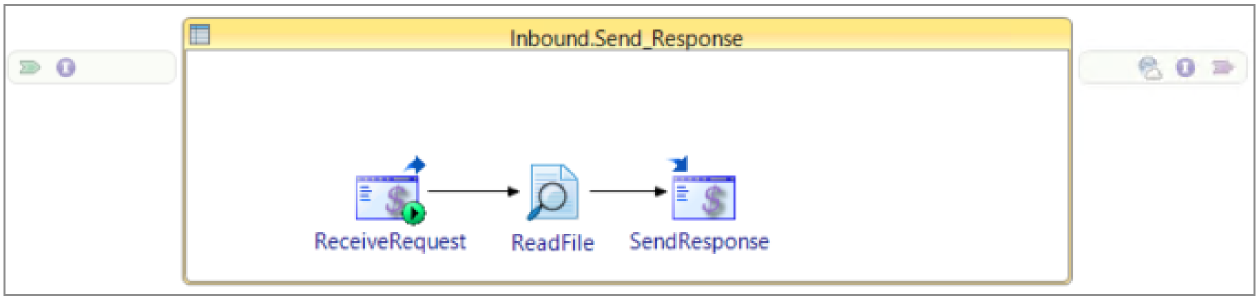
Send Request

Use the Send Request process on Machine 1 to initiate a PunchOutSetupRequest transaction. You can access this process from the TIBCO ActiveMatrix BusinessWorks project panel in the **Outbound** folder. The private process reads a sample cXML PunchOutSetupRequest from a local directory and sends the document to the BusinessConnect Container Edition server by Send PunchOutSetupRequest that is a Send Request/Notification resource, and waits for a response message. The process ends when SendRequest receives the response message to the request.



Send Response

Use the Send Response process on Machine 2 to respond to the PunchOutSetupRequest message. You can access this process from the TIBCO ActiveMatrix BusinessWorks project panel in the **Inbound** folder. When the private process receives a request document from the BusinessConnect Container Edition server, the private process reads a sample cXML PunchOutSetupResponse document from a local directory and sends the document to the BusinessConnect Container Edition server by Send PunchOutSetupResponse that is a Send Response resource.



Running Send Response on Machine 2

To start the **Send Response** process on Machine 2, perform the following steps:

Procedure

1. In the opened TIBCO ActiveMatrix BusinessWorks project on Machine 2, expand the **Inbound** folder.
2. Click the **Send Response** process in Project Explorer.
3. Click the **Receive_3A4_Request** activity.
4. On the **General** tab, select **cxml.org/PunchOut/1.2/PunchOutSetupRequest** from the **Operation Name** list.
5. Click **Apply** to save the changes.
6. On the **Tester** tab, click the dropdown icon next to the **BWApplication** icon ▶.
7. Select **Run Configuration** from the dropdown list.

   Note: If you want to run the process in the debug mode, click **Debug BWApplication** icon > **Debug Configurations**.

8. In the **Run Configurations** dialog, expand **BusinessWorks Application** > **BWApplication**.
9. From the list of the processes displayed, you can select one or multiple items. In this

tutorial, select **Inbound** > **Send Response**.

10. Click **Apply** > **Run**.

    You can see BusinessWorks logs on the **Console** tab.

Running Send Request on Machine 1

You must start the **`Send Response`** process on Machine 2 before proceeding with this section.

To start the **`Send Request`** process on Machine 1, perform the following steps:

Procedure

1. In the open TIBCO ActiveMatrix BusinessWorks project on Machine 1, expand the **Outbound** folder.
2. Click the **`Send Request`** process in Project Explorer.
3. Click the **Send PunchOutSetupRequest** activity.
4. On the **General** tab, select **cxml.org/PunchOut/1.2/PunchOutSetupRequest** from the **Operation Name** list.
5. Click **Apply** to save the changes.
6. On the **Tester** tab, click the dropdown icon next to the **BWApplication** icon ▶.
7. Select **Run Configuration** from the dropdown list.

    > **Note:** If you want to run the process in the debug mode, click **Debug BWApplication** icon > **Debug Configurations**.

8. In the **Run Configurations** dialog, expand **BusinessWorks Application** > **BWApplication**.
9. From the list of the processes displayed, you can select one or multiple items. In this tutorial, select **Outbound** > **Send Request**.
10. Click **Apply** > **Run**.

    You can see BusinessWorks logs on the **Console** tab.

Expected Results

If everything works as expected, the following sequence of events occurs:

1. The tester on Machine 1 pauses at the Send PunchOutSetupRequest resource indicating that this activity is waiting for a response from its BusinessConnect Container Edition server.
2. The tester on Machine 2 is triggered, and then steps through the events and sends a response to its BusinessConnect Container Edition server.
3. The tester on Machine 1 resumes its activity at the Send PunchOutSetupRequest resource and runs to completion.

Status Codes

This section describes the status codes used by BusinessConnect Container Edition - ebXML Protocol. 200~699 are standard HTTP codes and BusinessConnect Container Edition-defined codes, when 1100~4999 are codes defined by BusinessConnect Container Edition - ebXML Protocol.

These status codes and their descriptions are displayed in the following locations:

- **Private messages** The following private message classes include the **statusCode** and **statusMsg** fields. See Private Process Message Formats:
  - `ae/ebXML/InitiatorRequest`
  - `ae/ebXML/InitiatorResponse`
  - `ae/ebXML/Advisory` (This is for both error and advisory signal messages.)
- **Audit log** The **Description** field in the transaction detail view might display a status code with an accompanying description. See Transaction Details View.

Status Codes

Code (statusCode field)	Description (statusMsg field)	Role	Comment
`200`	OK	HTTP(S)	
`201~299`	HTTP(S) OK codes	HTTP(S)	
`300~503`	HTTP(S) error codes	HTTP(S)	
`510`	No valid HTTP response.	HTTP(S)	The most likely causes are that the trading partner sent a trading partner identity that was not

Code (statusCode field)	Description (statusMsg field)	Role	Comment
			recognized by BusinessConnect Container Edition, or that the trading partner did not send a correct HTTP response.
`699`	Failed to connect to trading partner.	BusinessConnect Container Edition	
`1100`	`Received [request|response|error] message from TP.`	Information	
`1101`	`Received [signed|] [encrypted|] [sync/async|] [action|] from TP.`	Information	
`1102`	`Ignoring [Request/response|Receipt Acknowledgment|MSH Acknowledgment] message. Reason:` *detailed_description*`.`	Information	
`1651`	`Receipt Acknowledgment received for message ID:` *message_id*`.`	Information	The trading partner has received a Receipt Acknowledgment for the indicated message.
`1652`	`Acceptance Acknowledgment received for message ID:` *message_id*`.`	Information	
`1653`	`MSH Acknowledgment received`	Information	The trading partner

Code (statusCode field)	Description (statusMsg field)	Role	Comment
	`for message ID:` *message_id*.		has sent an MSH Acknowledgment for the indicated message.
1999	`Info:` *detailed_description*.	Information	
2999	`Warning:` *detailed_description*.	Warning	
3101	`Missing either operation ID or To TP name.`	Error	
3102	`Configuration read error. Reason:` *detailed_description*.	Error	
3103	`Operation not allowed. Reason:` *detailed_description*.	Error	
3104	`Host Identity for Authentication not configured - message cannot be signed.`	Error	
3105	`To or From Role not specified - receipt acknowledgment cannot be constructed.`	Error	
3106	`Partner Identity for` *trading_partner_name* `not configured - [Request|Response|Receipt|MshAck|Error|Exception] cannot be [signed|encrypted].`	Error	
3107	`Config read failed for remote TP. Reason:` *detailed_description*.	Error	

Code (statusCode field)	Description (statusMsg field)	Role	Comment
3108	Config read failed for response action from TP.	Error	
3109	Inconsistency between configuration and message. Reason: *detailed_description*.	Error	
3110	Unknown Service Action. Service=*service* Service type=*service_type* Action=*action*.	Error	
3111	To party Domain *domain* and Identity *identity* does not match with configuration.	Error	
3112	Host not allowed to send the response to TP *trading_partner_name*. Reason: *detailed_description*.	Error	
3113	Failed to determine the transport. Reason: *detailed_description*.	Error	
3114	Private process supplied 'businessDocumentName' of *businessDocumentName* does not match configured names of *name_in_GUI*.	Error	
3115	Message ordering is only available for transactions with Once-And-Only-Once semantics and SyncReplyMode of 'none'.	Error	
3116	'MessageOrder' element must not be present with the 'SyncReply' element.	Error	

Code (statusCode field)	Description (statusMsg field)	Role	Comment
3117	`Time To Live expired`	Error	
3118	`Failed to get payload against the manifest. Reason:` *detailed_description*`.`	Error	
3119	`CPAId` *cpa_id* `does not match with configuration.`	Error	
3120	`Version` *business_process_version* `is not supported.`	Error	
3121	`From party Domain` *domain* `and Identity` *identity* `does not match with configuration.`	Error	
3201	`Post aborted - exhausted maximum number of retries (`*max_retries*`).`	Error	*max_retries* is the maximum number of retries configured in the GUI.
3202	`Unknown Error while sending [Request/response|Receipt Acknowledgment|MSH Acknowledgment].`	Error	
3203	`Posting to` *trading_partner_name* `failed with code` *http_return_code*`. Reason:` *detailed_description*`.`	Error	
3301	`Timed out waiting for [sync|async] [Acknowledgment|Receipt Acknowledgment|Acceptance Acknowledgment].`	Error	

Code (statusCode field)	Description (statusMsg field)	Role	Comment
3302	`Unexpected error while waiting for async response. Reason:` *detailed_description*.	Error	
3303	`No MSH Acknowledgment received after exhausting sending max. number of retries (`*max_retries*`).`	Error	*max_retries* is the maximum number of retries configured in the GUI.
3304	`No response data was received from trading partner.`	Error	
3401	`MIME formatting error. Reason:` *detailed_description*.	Error	
3402	`ebXML SOAP envelope generation error. Reason:` *detailed_description*.	Error	
3403	`Failed to create MSH Ack Envelope. Reason:` *detailed_description*.	Error	
3404	`Failed to create Receipt acknowledgment. Reason:` *detailed_description*.	Error	
3405	`Failed to create Attachment Sequence. Reason:` *detailed_description*.	Error	
3501	`Client authentication failed for Service=`*service* `Action=`*action*`. Reason:` *detailed_description*.	Error	
3502	`[S/MIME|XML] Encryption`	Error	

Code (statusCode field)	Description (statusMsg field)	Role	Comment

	`failed for Service=`*service* `Action=`*action*`. Reason:` *detailed_description*`.`
	`[S/MIME|XML] Encryption failed. Reason:` *detailed_ description*`.`		

3503	`[S/MIME|XML] Decryption failed for Service=`*service* `Action=`*action*`. Reason:` *detailed_description*`.`
	`[S/MIME|XML] Decryption failed. Reason:` *detailed_ description*`.`
		Error	

3504	`ebXML SOAP envelope signing failed for Service=`*service* `Action=`*action*`. Reason:` *detailed_description*`.`
	`ebXML SOAP envelope signing failed. Reason:` *detailed_ description*`.`
		Error	

3505	`ebXML SOAP envelope signature verification failed for Service=`*service* `Action=`*action*`. Reason:` *detailed_description*`.`
	`ebXML SOAP envelope signature verification failed. Reason:` *detailed_ description*`.`
		Error	

3506	`MSH Acknowledgment signature references verification failed.`	Error	
3507	`Receipt Acknowledgment NR`	Error	

Code (statusCode field)	Description (statusMsg field)	Role	Comment
	digests verification failed. Reason: *detailed_ description*.		
3601	MIME message parsing error. Reason: *detailed_description*.	Error	
3602	ebXML SOAP envelope parsing error. Reason: *detailed_ description*.	Error	
3603	Error while processing response from TP. Reason: *detailed_description*.	Error	
3604	Unable to compute the operation ID from message.	Error	
3605	Failed to validate ebXML envelope parameters against configuration. Reason: *detailed_description*.	Error	
3606	Inbound Payload/Attachments extraction error. Reason: *detailed_description*.	Error	
3607	Incorect RefToMessageId for [Request/response|Receipt Acknowledgment|MSH Acknowledgment] message.	Error	
3608	Sync Response Initialization failed.	Error	
3609	Failed to process Synchronous [response|MSH Acknowledgment].	Error	

Code (statusCode field)	Description (statusMsg field)	Role	Comment
3610	Error while sending request to private process. Reason: *detailed_description*.	Error	
3611	Timeout waiting for [response|acknowledgment] from private process.	Error	
3612	SOAP Fault (code: [server|client], text: *brief_description*, actor: *actor*, details: *detailed_description*).	Error	
3613	MSH Error (severity: [Warning|Error]).	Error	*msh_error_code* is one of the MSH error codes defined by the ebMS specifications.
3614	Application supplied a duplicate message ID.	Error	
3615	Could not find expected SOAP Fault message.	Error	
3616	Unexpected error while processing sequenced messages. Reason: *detailed_description*.	Error	
3617	Received message missing expected SequenceNumber.	Error	
3618	Sequenced message timed out while queued.	Error	
3619	Application supplied a duplicate message ID - not allowed for Once-And-Only-	Error	

Code (statusCode field)	Description (statusMsg field)	Role	Comment
	`Once semantics.`		
`3620`	`Sequenced message timed out while received.`	Error	
`3650`	`Generic Exception received for messageID:` *message_id*`. Reason:` *detailed_description*`.`	Error	Used by the responder private process to indicate that an exception occurred when processing an inbound request. See Outbound Response Format. See Exception Signals.
`3651`	`Receipt Exception received for messageID:` *message_id*`. Reason:` *detailed_description*`.`	Error	
`3652`	`Acceptance Exception received for messageID:` *message_id*`. Reason:` *detailed_description*`.`	Error	Used by the responder private process to indicate that an exception occurred when processing an inbound request. See Outbound Response Format. See Exception Signals.
`3691`	`Processing failed for`	Error	

Code (statusCode field)	Description (statusMsg field)	Role	Comment
	`Service=`*service* `Action=`*action*`.` `Reason:` *detailed_description*`.`		
3692	`Failed to process [MSH|Receipt] Acknowledgment for [request|response|Receipt Acknowledgment]. Reason:` *detailed_description*`.`	Error	
3699	`General processing error:` *detailed_description*`.`	Error	
3701	`Reliable storage insert failed. Reason:` *detailed_description*`.`	Error	
3702	`Reliable storage update failed. Reason:` *detailed_description*`.`	Error	
3703	`Reliable storage query failed. Reason:` *detailed_description*`.`	Error	
3801	`ebXML SOAP envelope validation error. Reason:` *detailed_description*`.`	Error	
3802	`[Incoming|Outgoing] request payload failed schema validation. Reason:` *detailed_description*`.`	Error	
3803	`[Incoming|Outgoing] response payload failed schema validation. Reason:` *detailed_description*`.`	Error	
3804	`Receipt acknowledgment`	Error	

Code (statusCode field)	Description (statusMsg field)	Role	Comment
	failed schema validation.		
3805	Missing schema URI for validation.	Error	
3806	Message validation error. Reason: *detailed_description*.	Error	
3807	Exception message failed schema validation. Reason: *detailed_description*.	Error	
3901	Reply to DMZ failed. Reason: *detailed_description*.	Error	
3991	Unkown Internal Engine Error. Reason: *detailed_description*.	Error	
3992	Missing error details resource. Reason for original error: *detailed_description*.	Error	Failed to find the error description resource. This is most likely because of packaging error.
4999	Debug: *detailed_description*.	Error	

Property Reference

The following table lists the properties that you can set in the **BusinessConnect Container Edition > System Settings > Others > Activate Protocol Plugins > ebXML** in the BusinessConnect Container Edition console.

Property Reference

Property	Description
ebxml.sequence.pollInterval	(In milliseconds) The interval at which BusinessConnect Container Edition - ebXML Protocol polls for queued messages that are ready to be dispatched and messages that have expired. The default value is `300000` milliseconds.
ebxml.sequence.iterateWait	(In milliseconds) The interval at which BusinessConnect Container Edition - ebXML Protocol dispatches queued messages to private processes. The default value is `5000` milliseconds.
ebxml.duns.check	Enforces the correct identity format in the Participants configuration panel when the DUNS domain is used. Select **duns** to enforce a nine-digit DUNS number, **duns+4** to enforce a thirteen-digit DUNS number, and **none** to turn off this function.
ebxml .ib.servicetype.check.relaxed	If selected, the type attribute of the `<Service>` element is ignored for inbound requests and responses.
ebxml .ob.reference.role.populate	If selected, the role attribute of the `<Reference>` element in an outbound message is populated with a valid URI based on the operation ID and the business document name. If cleared, a fixed string value "simple" is populated.

Property	Description
ebxml.ib.ack.dupElim.waived	If selected, consistency check of duplicate elimination is ignored for inbound acknowledgments.
ebxml.ob.ack.dupElim.waived	If selected, the `<DuplicateElimination>` element is ignored for outbound acknowledgments, and the consistency check of duplicate elimination is also ignored.
bc.protoc ol.ebxml.binaryEnabled	If selected, binary data documents can be sent in private process communication.
ebxml .ib.manifest.id.relax.validation	If selected, the manifest id validation is disabled when the manifest id starts with a numeric character.
ebxml.allow.generated.Content-Disposition.filename.param	If not selected, the Content-Disposition header in an ebMS2 outbound message is disabled when the attachment name is not specified in the InitiatorRequest of a private process. The default value is `enabled` or `true`.
ebxml.allow.nil.payload	If selected, an empty string can be used in the **stringData** field of InitiatorRequest in the TIBCO BusinessWorks and the application payload can be entered as an attachment with an appropriate value in the **content-type** field. **Note: Validate Outgoing Document** will not work against the schema as the primary payload fields accept nil or empty values. Therefore, it is recommended to disable **Validate Outgoing Document** at the operation level. The default value is `disabled` or `false`.
ebxml.fileref.threshold	When the content or payload size of an input message exceeds the configured threshold value, it is passed as file reference to the private process. Also, the path of the file can be displayed in the private process of the responder.

Property	Description
	The default value is 5 MB (5000000).
ebxml.bo.include.http.headers	If selected, the HTTP headers are sent to the TIBCO ActiveMatrix BusinessWorks. **Note:** IS restart is not required.

Smart Routing

You can use BusinessConnect Container Edition to define business rules to route messages to specific private processes. This function is called smart routing, which requires configuration of the BusinessConnect Container Edition server through the BusinessConnect Container Edition console, and configuration of the BusinessConnect palette resources in TIBCO Business Studio. In the BusinessConnect Container Edition console, you can set up the business rules and specify the smart ID to be assigned to messages that meet the conditions of the rule. In the BusinessConnect palette resources, you can configure which private processes receive messages that include specific smart IDs.

To configure the smart routing fields, go to **BusinessConnect Container Edition > System Settings > Others > Smart Routing > Private Process Smart Routing > Add New Smart Route**.

Smart Routing Fields for ebXML

Field	Description
Enabled	Select this checkbox to use the private process smart routing.
Protocol	The business protocol for the message. Use the asterisk character (*) to match all protocols. This is a required field.
From	The domain identity for the trading partner that sends the original message. If Host (your company) sends a request to Partner and Partner sends a response, you might want to use smart routing for the response. In this case, the **From** field is matched by Host because Host is the originator of the business transaction. Use the asterisk character (*) to match all hosts and partners, but do not use the asterisk character with a string. For example, do not use `TIB*`. This is a required field.

Field	Description
To	The domain identity for the trading partner that receives the original message. If a partner sends a request to Host (your company), you might want to use smart routing for the request. In this case, the **To** field is matched by Host because Host is the recipient of the request. Use the asterisk character (*) to match all hosts and partners, but do not use the asterisk character with a string. For example, do not use `TIB*`. This is a required field.
Direction	The *business* direction of the message: inbound or outbound. For example, if a partner sends a request to Host (your company), both the business direction and the message direction are inbound. However, if Host sends a request to Partner and Partner sends a response, the *message* direction of the response is inbound, but the *business* direction of the response is outbound because the original message was outbound. The asterisk character (*) matches both directions. This is a required field.

Operation ID	The location and identifier of the operation. This takes the form of a series of nodes.
	Use one asterisk character (*) to match all operations directly under a specific node. For example:
	`BCCE/*/*` matches `BCCE/MyNotify/Test` but not `BCCE/MyNotify/Test/notify1`
	Use two asterisk characters (**) to match operations recursively. Use double asterisks alone or use them as the last node. For example:
	`BCCE/MyNotify/**` matches `BCCE/MyNotify/1.3/Test`
	`BCCE/MyNotify/**/notify1` is the same as `BCCE/MyNotify/**`. The software ignores any nodes after a double asterisk.
	You can use both a single asterisk and a double asterisk, for example:
	`BCCE/*/1.0/**` matches `BCCE/Test-01/1.0/A/B`
	This is a required field.

Smart ID	An identifier that indicates which smart routing rules the message satisfied.

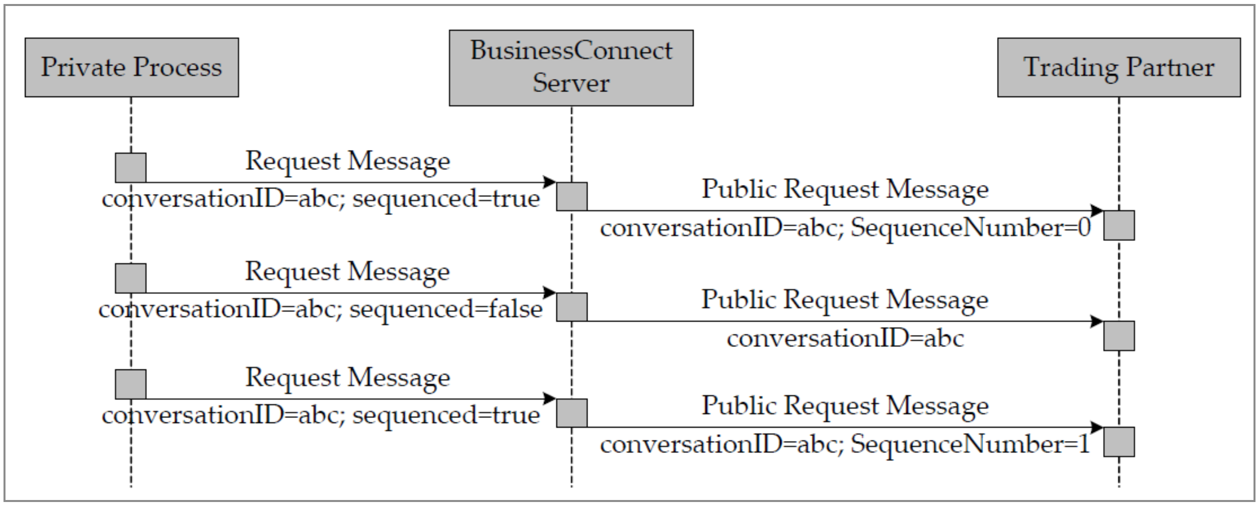
Field	Description
	Any combination of alphanumeric characters can be used, with a minimum of one character and a maximum of 25 characters. This is a required field.

Notes on Sequenced Messages

This section describes the caveats that you must be aware of in dealing with sequenced messages.

Unsequenced Messages in Active Conversations

In an active conversation (sequence number is not zero), BusinessConnect Container Edition - ebXML Protocol does not determine whether the **sequenced** field is set to `true` in an `ae/ebXML/InitiatorRequest` message. See the figure for an example:



In the example, the second request message does not require sequencing, but is part of an on-going sequenced conversation. BusinessConnect Container Edition - ebXML Protocol does not detect such an anomaly and sends this request message to the trading partner as a sequenced message. Therefore, it is the responsibility of the private process to make sure sequenced conversations contain messages that make sense for the business workflow.

Reusing Sequence Numbers

Since sequence numbers are unique to a conversation, you cannot reuse sequence numbers with BusinessConnect Container Edition - ebXML Protocol. However, it is possible for BusinessConnect Container Edition - ebXML Protocol to encounter an error after allocating a sequence number to an outbound request message. In this scenario, the sequence number is considered used and cannot be recovered for reuse. You must recover the sequence number manually by updating the database entry `NEXT_TO_TP_SEQ_NO` in a table called `EBXML_CONVERSATION_QUEUE`. The `NEXT_TO_TP_SEQ_NO` table entry must reflect the correct sequence number the next message in the conversation. To do so, run the query for your database, substituting *sequence_no* and *conversation_id* with the desired values.

For MySql use:

```
update ebxml_conversation_queue set NEXT_TO_TP_SEQ_NO=sequence_no,
LASTMODIFIED=CURRENT_TIMESTAMP where conversation_id='conversation_id';
```

For MSSQL use:

```
update ebxml_conversation_queue set NEXT_TO_TP_SEQ_NO=sequence_no,
LASTMODIFIED=CURRENT_TIMESTAMP where conversation_id='conversation_id';
```

For Oracle use:

```
update ebxml_conversation_queue set NEXT_TO_TP_SEQ_NO=sequence_no,
LASTMODIFIED=SYSTIMESTAMP where conversation_id='conversation_id';
```

For db2 use:

```
update ebxml_conversation_queue set NEXT_TO_TP_SEQ_NO=sequence_no,
LASTMODIFIED= DEFAULT where conversation_id='conversation_id';
```

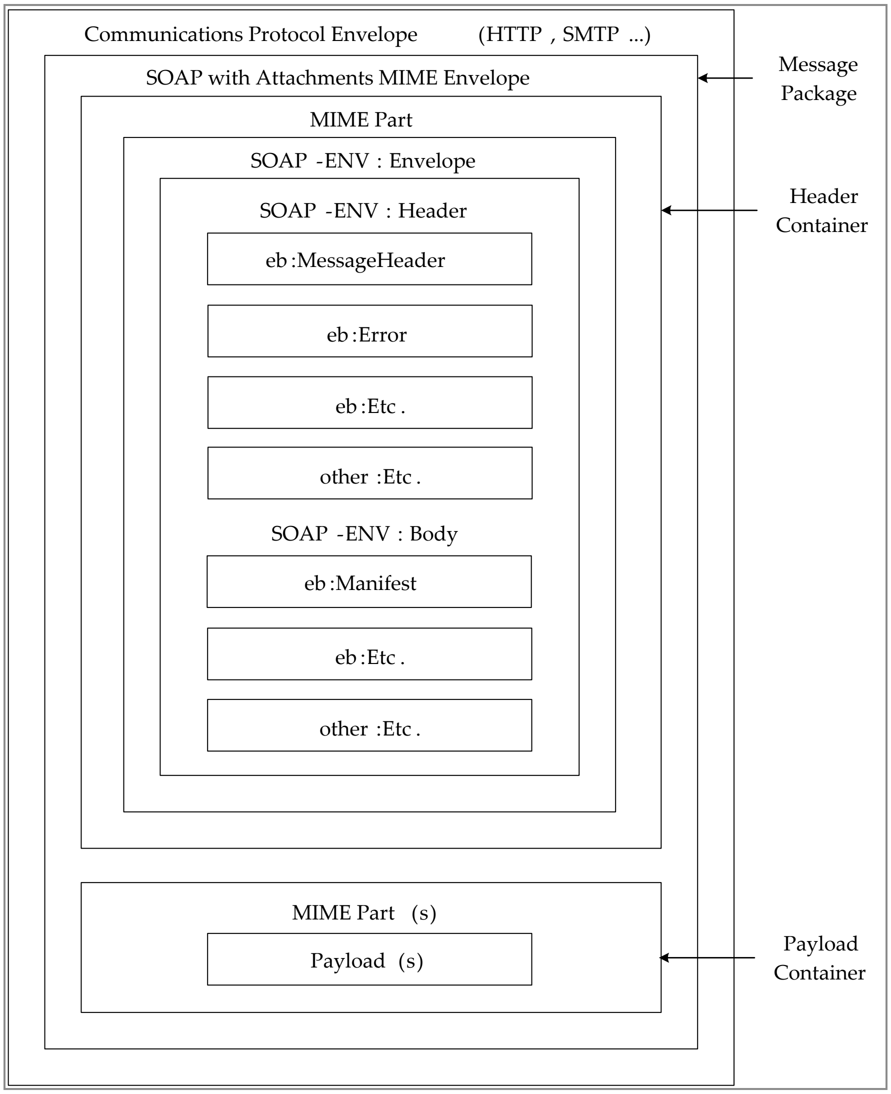
Public Messages

This section describes the elements and attributes in the ebXML public messages and how they correspond to GUI elements in the BusinessConnect Container Edition console or to private message fields.

ebXML Public Message Structure

The structure of an ebXML public message is shown in the following diagram:

Figure 18: ebXML Public Message Structure

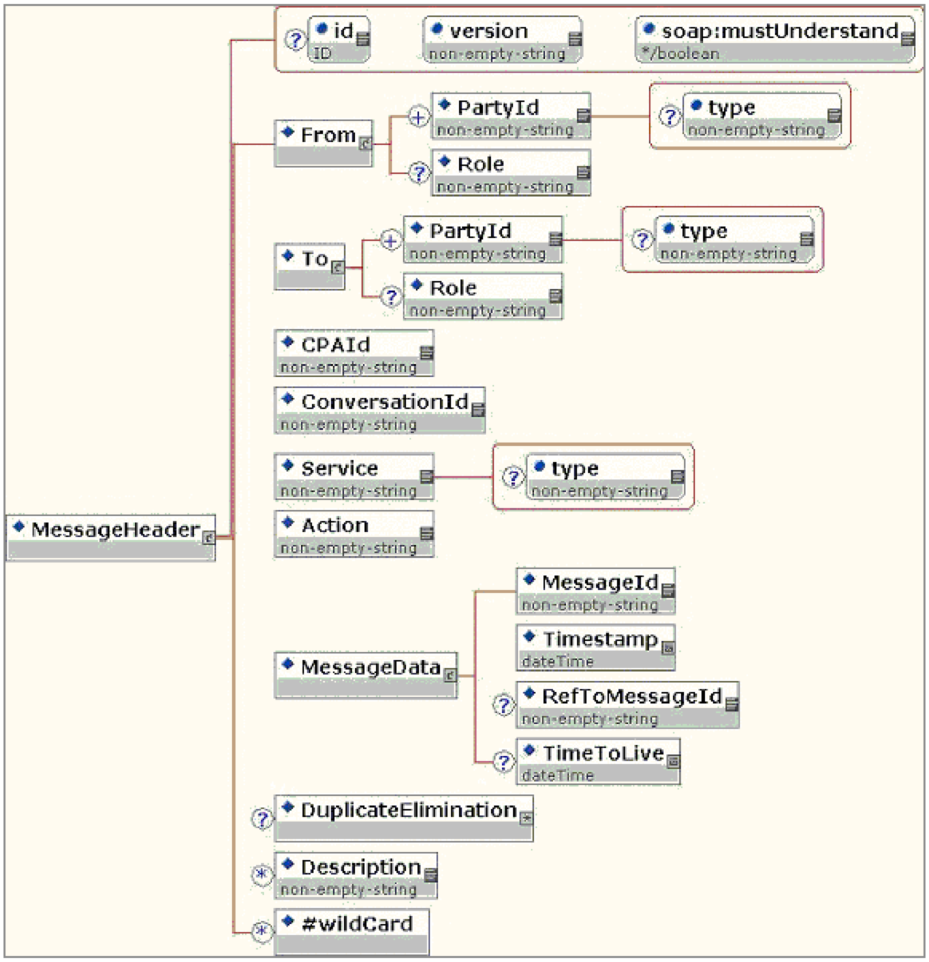


SOAP Envelope

This section describes the elements and attributes in the ebXML SOAP envelope and how they correspond to GUI elements in the BusinessConnect Container Edition console or to private message fields.

MessageHeader Element

The public message MessageHeader element, as defined by the ebXML Message Service Specification version 2.0, is shown in the following diagram.



See the following table for the corresponding GUI element in the BusinessConnect Container Edition console or the corresponding private message field of each element or attribute.

MessageHeader Element in Public Messages

Element/Attribute	Source of Value
id	Not included.

Element/Attribute	Source of Value
version	The string "2.0".
soap:mustUnderstand	The string "1".
From/PartyId	The ID string (without the domain string) in **BusinessConnect Container Edition > Partner Management > Partners > partner > Protocols > ebXML > Edit Configurations > General > Default Domain Identity**.
From/PartyId/type	The domain string (without the ID string) in **BusinessConnect Container Edition > Partner Management > Partners > partner > Protocols > ebXML > Edit Configurations > General > Default Domain Identity**. **Note:** For outbound messages, this attribute is not included if the domain string is `URI`.
From/Role	**BusinessConnect Container Edition > B2B Administration > Operations Editor > ebXML > transaction > name Transaction > Roles > Initiating Role** (or **Responding Role**). Or, in the case of override in operation binding: **BusinessConnect Container Edition > Partner Management > Business Agreements > agreement > ebXML > Edit Configurations > Operations > transaction > Initiating Role**. **Note:** If this field is empty, the `From/Role` element is not included. See Operation Settings Tab.
To/PartyId	The ID string (without the domain string) in **BusinessConnect Container Edition > Partner**

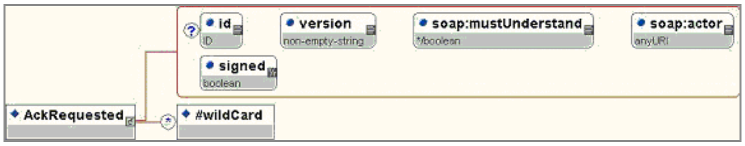
Element/Attribute	Source of Value
	Management > **Partners** > **partner** > **Protocols** > **ebXML** > **Edit Configurations** > **General** > **Default Domain Identity**.
To/PartyId/type	The domain string (without the ID string) in **BusinessConnect Container Edition** > **Partner Management** > **Partners** > **partner** > **Protocols** > **ebXML** > **Edit Configurations** > **General** > **Default Domain Identity**. **Note:** For outbound messages, this attribute is not included if the domain string is `URI`.
To/Role	**BusinessConnect Container Edition** > **B2B Administration** > **Operations Editor** > **ebXML** > **transaction** > **name Transaction** > **Roles** > **Initiating Role** (or **Responding Role**). Or, in the case of override in operation binding: **BusinessConnect Container Edition** > **Partner Management** > **Business Agreements** > **agreement** > **ebXML** > **Edit Configurations** > **Operations** > **transaction** > **Initiating Role**. **Note:** If this field is empty, the `To/Role` element is not included. See Operation Settings Tab.
CPAId	**BusinessConnect Container Edition** > **Partner Management** > **Partners** > **partner** > **Protocols** > **ebXML** > **Edit Configurations** > **General** > **Specification ID**. Or, in the case of override in business agreement: **BusinessConnect Container Edition** > **Partner Management** > **Business Agreements** > **agreement** > **ebXML** > **Edit Configurations** > **Override Configuration** >

Element/Attribute	Source of Value
	Specification ID. See Overriding Participant Settings.
ConversationId	This is for outbound message only. The **conversationID** field in the `ae/ebXML/InitiatorRequest` message class. If a value is not specified there, BusinessConnect Container Edition - ebXML Protocol randomly generates one in the outbound sequenced message. See Outbound Request Format.
Service	**BusinessConnect Container Edition > B2B Administration > Operations Editor > ebXML > transaction > name Transaction > Service Information > Service**.
Service/type	**BusinessConnect Container Edition > B2B Administration > Operations Editor > ebXML > transaction > name Transaction > Service Information > Service Type**.
Action	**BusinessConnect Container Edition > B2B Administration > Operations Editor > ebXML > transaction > [Request|Notify Request|Response] Action > General > Name**.
MessageData/MessageId	This is for outbound message only. The **messageID** field in the `ae/ebXML/InitiatorRequest` message class. If a value is not specified there, BusinessConnect Container Edition - ebXML Protocol randomly generates one. See Outbound Request Format.
MessageData/Timestamp	This is for outbound message only. Automatically generated by BusinessConnect Container Edition - ebXML Protocol.
MessageData/RefToMessageId	The `MessageData/MessageId` element in the related

Element/Attribute	Source of Value
	inbound message. This element is not included in request messages.
MessageData/TimeToLive	This is for outbound message only. Calculated from adding the value of **BusinessConnect Container Edition > B2B Administration > Operations Editor > ebXML > transaction > Action > General > Lifetime of Message** to the timestamp in `MessageData/Timestamp`. **Note:** If Lifetime of Message has the value of `0`, this element is not included.
DuplicateElimination	**BusinessConnect Container Edition > B2B Administration > Operations Editor > ebXML > transaction > Action > General > Eliminate Duplicate Message**. See General Tab.
Description	Not included.

AckRequested Element

The public message AckRequested element, as defined by the ebXML Message Service Specification version 2.0, is shown in the following diagram.



This element corresponds to the following GUI element in the BusinessConnect Container Edition console: **BusinessConnect Container Edition > B2B Administration > Operations Editor > ebXML > transaction > Action > Signal > Require MSH Acknowledgment Signal**.

See Signal Tab.

Note: This element does not exist in MSH Acknowledgment messages or MSH Error messages, regardless of the configuration.

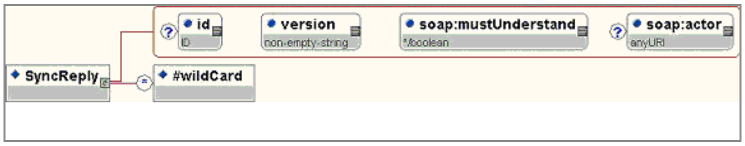
See the following table for the corresponding GUI element in the BusinessConnect Container Edition console or the corresponding private message field of each element or attribute.

AckRequested Element in Public Messages

Element/Attribute	Source of Value
id	Not included.
version	The string "2.0".
soap:mustUnderstand	The string "1".
soap:actor	The string "`urn:oasis:names:tc:ebxml-msg:actor:toPartyMSH`".
signed	**BusinessConnect Container Edition > B2B Administration > Operations Editor > ebXML > transaction > Action > Signal > Require Non-repudiation of MSH Acknowledgment**. See Signal Tab.

SyncReply Element

The public message SyncReply element, as defined by the ebXML Message Service Specification version 2.0, is shown in the following diagram.



This element corresponds to the following GUI element in the BusinessConnect Container Edition console: **BusinessConnect Container Edition > B2B Administration > Operations Editor > ebXML > transaction > Action > General > Synchronous Reply Mode**.

See General Tab.

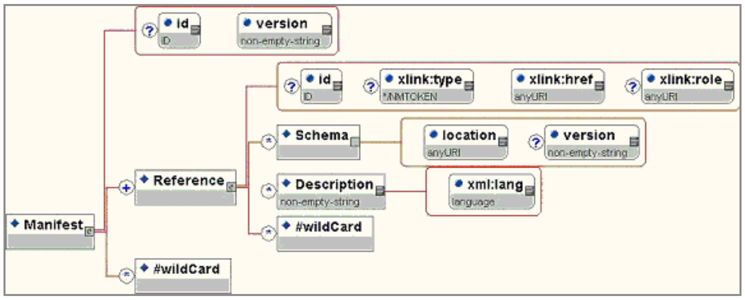
See the following table for the corresponding GUI element in the BusinessConnect Container Edition console or the corresponding private message field of each element or attribute.

SyncReply Element in Public Messages

Element/Attribute	Source of Value
id	Not included.
version	The string "2.0".
soap:mustUnderstand	The string "1".
soap:actor	The string "`http://schemas.xmlsoap.org/soap/actor/next`"

Manifest Element

The public message Manifest element, as defined by the ebXML Message Service Specification version 2.0, is shown in the following diagram.



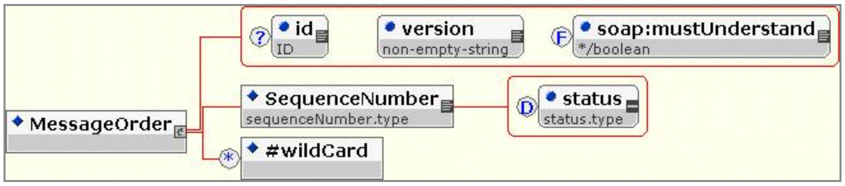
See the following table for the corresponding GUI element in the BusinessConnect Container Edition console or the corresponding private message field of each element or attribute.

Manifest Element in Public Messages

Element/Attribute	Source of Value
id	Not included.
version	The string "`2.0`".
Reference/id	Not included.
Reference/xlink:type	The string "`simple`".
Reference/xlink:href	The URI of the payload document. For outbound messages, it is automatically generated by BusinessConnect Container Edition - ebXML Protocol.
Reference/xlink:role	The string "simple".
Reference/Schema	Not included.
Reference/Description	Not included.

MessageOrder Element

The public message MessageOrder element, as defined by the ebXML Message Service Specification version 2.0, is shown in the following diagram.



The `<MessageOrder>` element is included in the outbound request messages when the following conditions are met:

- The public message is being delivered asynchronously with once-and-only-once delivery semantics. See Delivery Semantics.
- The outbound `ae/ebXML/InitiatorRequest` private message contains a non-empty

conversationID field, and the **sequenced** field set to `true`.

When an inbound request message contains the `<MessageOrder>` element, and then the element is included in the corresponding outbound response message.

If an inbound request message contains a `<MessageOrder>` element, and the transaction is not configured with asynchronous mode or the message also contains a `<SyncReply>` element, and then BusinessConnect Container Edition - ebXML Protocol ends the transaction and sends an MSH error to the trading partner with an error code of `Inconsistent` and a severity of `Error`.

See the following table for the corresponding GUI element in the BusinessConnect Container Edition console or the corresponding private message field of each element or attribute.

MessageOrder Element in Public Messages

Element/Attribute	Source of Value
id	Not included.
version	The string "2.0".
soap:mustUnderstand	The string "1".
SequenceNumber	Automatically generated by BusinessConnect Container Edition - ebXML Protocol. See Sequenced Messages. **Note:** This value can be found in the BusinessConnect Container Edition database, in the `NEXT_TO_TP_SEQ_NO` column of the `EBXML_CONVERSATION_QUEUE` table.
SequenceNumber/status	The string "Continue".

Business-Level Signals

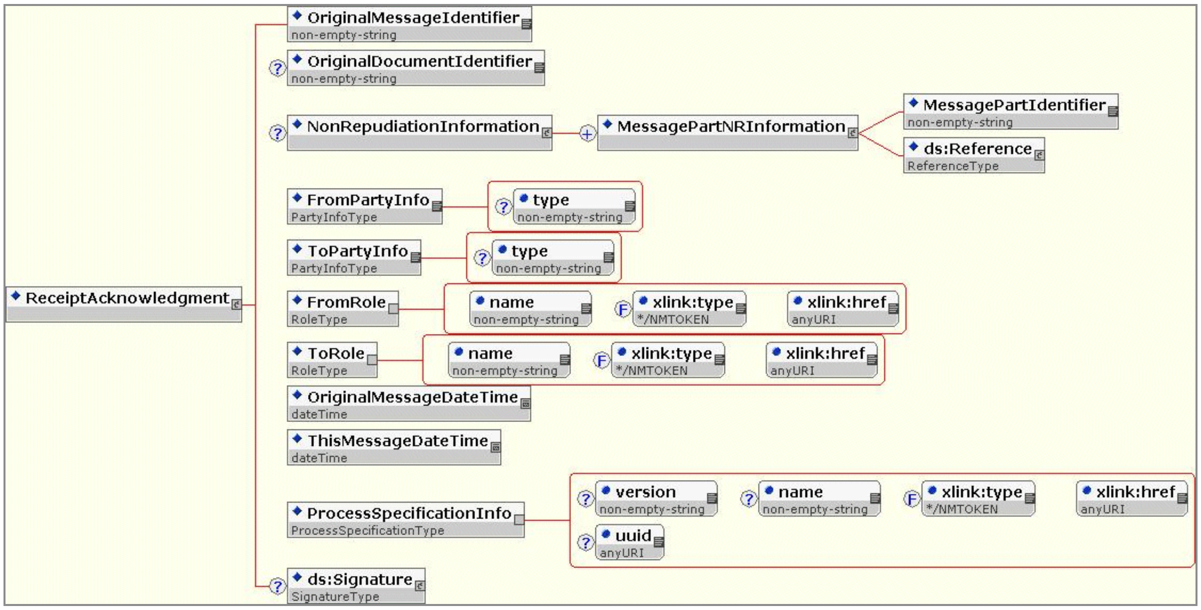
This section describes the elements and attributes in the ebXML business-level signal messages and how they correspond to GUI elements in the BusinessConnect Container Edition console or to private message fields.

Receipt Acknowledgment Signals

If an action message requires a receipt acknowledgment, BusinessConnect Container Edition - ebXML Protocol sends the receipt acknowledgment after it validates the payload of the action message. You can configure an action to require a receipt acknowledgment at **BusinessConnect Container Edition > B2B Administration > Operations Editor > ebXML > `transaction` > Action > Signal > Require Receipt Acknowledgment Signal**. See Signal Tab.

When the BusinessConnect Container Edition server receives this public ebXML message, it publishes an advisory signal message on Rendezvous or JMS to update the private process. See Advisory Signal Message Format.

The ReceiptAcknowledgment signal message, as defined by ebXML BPSS 1.05, is shown in the following diagram.



See the following table for the corresponding GUI element in the BusinessConnect Container Edition console or the corresponding private message field of each element or attribute, or elements or attributes in the action message.

ReceiptAcknowledgment Public Message

Element/Attribute	Source of Value
OriginalMessageIdentifier	The `MessageData/MessageId` element in the SOAP header of the action message.
OriginalDocumentIdentifier	Not included.
NonRepudiationInformation/ MessagePartNRInformation	If the inbound action message is signed, this contains the signature of the inbound message. If the inbound action message is not signed, this contains the MIME `content-id` of the payload business document. If the action message also includes attachments, the `<NonRepudiationInformation>` element contains multiple `<MessagePartNRInformation>` elements, each containing the MIME `content-id` of a MIME part in the action message.
FromPartyInfo	The `From/PartyId` element in the SOAP header of the action message.
FromPartyInfo/type	The `From/PartyId/type` attribute in the SOAP header of the action message.
ToPartyInfo	The `To/PartyId` element in the SOAP header of the action message.
ToPartyInfo/type	The `To/PartyId/type` attribute in the SOAP header of the action message.
FromRole	The `From/Role` element in the SOAP header of the action message.
ToRole	The `To/Role` element in the SOAP header of the action message.

Element/Attribute	Source of Value
OriginalMessageDateTime	The `MessageData/Timestamp` element in the SOAP header of the action message.
ThisMessageDateTime	Automatically generated by BusinessConnect Container Edition - ebXML Protocol.
ProcessSpecificationInfo/ version	**BusinessConnect Container Edition > B2B Administration > Operations Editor > ebXML > business process version > Name**.
ProcessSpecificationInfo/ name	**BusinessConnect Container Edition > B2B Administration > Operations Editor > ebXML > business process version > Name**.
ProcessSpecificationInfo/ xlink:type	The string "simple".
ProcessSpecificationInfo/ xlink:href	Automatically generated by BusinessConnect Container Edition - ebXML Protocol in the following format: `urn:organization/business_process/`*business_process_version* See the following contents for the corresponding GUI element in the BusinessConnect Container Edition console for each variable. • `organization` — **BusinessConnect Container Edition > B2B Administration > Operations Editor > ebXML > organization > Name**. • `business_process` — **BusinessConnect Container Edition > B2B Administration > Operations Editor > ebXML > business process > Name**. • `business_process_version` — **BusinessConnect Container Edition > B2B Administration > Operations Editor > ebXML > business process version > Name**.

Element/Attribute	Source of Value
ProcessSpecificationInfo/uuid	See the description of `ProcessSpecificationInfo/xlink:href`.
ds:Signature	Not included.

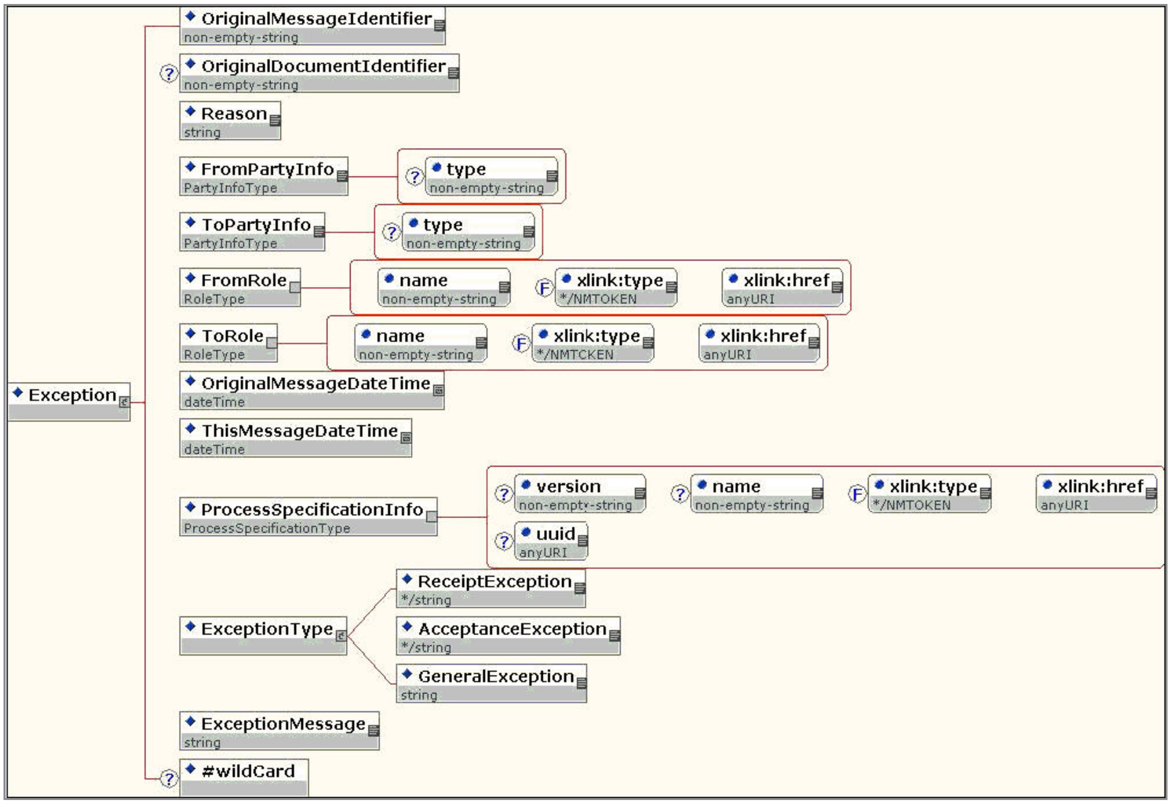
Exception Signals

The exception signal message indicates that a business-level exception has occurred. It is a public response message with an `<Exception>` element as the payload. See Processing Business-Level Signals.

When BusinessConnect Container Edition - ebXML Protocol receives an `ae/ebXML/ResponderResponse` message from the private process with `3650` or `3652` in the **statusCode** field, it sends an exception message to the trading partner. See Outbound Response Format.

When BusinessConnect Container Edition - ebXML Protocol receives an exception message from the trading partner, it updates the private process with both an advisory signal message and an error message. See Advisory Signal Message Format and Error Message Format.

The Exception message, as defined by ebXML BPSS 1.05, is shown in the following diagram:



See the following table for the corresponding GUI element in the BusinessConnect Container Edition console or the corresponding private message field of each element or attribute, or elements or attributes in the request message.

Exception Public Message

Element/Attribute	Source of Value
OriginalMessageIdentifier	The `MessageData/MessageId` element in the SOAP header of the request message.
OriginalDocumentIdentifier	Not included.
Reason	From the **statusMsg** field of the `ae/ebXML/ResponderResponse` message class.
FromPartyInfo	The `From/PartyId` element in the SOAP header of the request message.
FromPartyInfo/type	The `From/PartyId/type` attribute in the SOAP header of

Element/Attribute	Source of Value
	the request message.
ToPartyInfo	The `From/PartyId` element in the SOAP header of the request message.
ToPartyInfo/type	The `From/PartyId/type` attribute in the SOAP header of the request message.
FromRole	The `From/Role` element in the SOAP header of the request message.
ToRole	The `To/Role` element in the SOAP header of the request message.
OriginalMessageDateTime	The `MessageData/Timestamp` element in the SOAP header of the request message.
ThisMessageDateTime	Automatically generated by BusinessConnect Container Edition - ebXML Protocol.
ProcessSpecificationInfo/version	**BusinessConnect Container Edition > B2B Administration > Operations Editor > ebXML > business process version > Name**.
ProcessSpecificationInfo/name	**BusinessConnect Container Edition > B2B Administration > Operations Editor > ebXML > business process > Name**.
ProcessSpecificationInfo/xlink:type	The string "simple".
ProcessSpecificationInfo/xlink:href	Automatically generated by BusinessConnect Container Edition - ebXML Protocol in the following format: `urn:organization/business_process/business_process_version` See the following contents for the corresponding GUI element in the BusinessConnect Container Edition console

Element/Attribute	Source of Value
	for each variable. - organization — **BusinessConnect Container Edition > B2B Administration > Operations Editor > ebXML > organization > Name**. - business_process — **BusinessConnect Container Edition > B2B Administration > Operations Editor > ebXML > business process > Name**. - business_process_version — **BusinessConnect Container Edition > B2B Administration > Operations Editor > ebXML > business process version > Name**.
ProcessSpecificationInfo/uuid	See the description of ProcessSpecificationInfo/xlink:href.
ExceptionType	In an outbound response, the value of this element is formulated as follows: - **AcceptanceException** if the **statusCode** field is set to 3652 in the ae/ebXML/ResponderResponse message. - **GeneralException** if the **statusCode** field is set to any code other than 200 and 3652 in the ae/ebXML/ResponderResponse message. - **ReceiptException** if BusinessConnect Container Edition - ebXML Protocol fails to validate the inbound message payload.
ExceptionMessage	From the **response** field of the ae/ebXML/ResponderResponse message class.

TIBCO Documentation and Support Services

For information about this product, you can read the documentation, contact TIBCO Support, and join TIBCO Community.

How to Access TIBCO Documentation

Documentation for TIBCO products is available on the Product Documentation website, mainly in HTML and PDF formats.

The Product Documentation website is updated frequently and is more current than any other documentation included with the product.

Product-Specific Documentation

The following documentation for this product is available on the TIBCO BusinessConnect™ Container Edition - ebXML Protocol Product Documentation page:

- *TIBCO BusinessConnect™ Container Edition - ebXML Protocol Release Notes*
- *TIBCO BusinessConnect™ Container Edition - ebXML Protocol Installation*
- *TIBCO BusinessConnect™ Container Edition - ebXML Protocol User Guide - ebMS2 Standard*
- *TIBCO BusinessConnect™ Container Edition - ebXML Protocol User Guide - ebMS3/AS4 Standard*

Other TIBCO Product Documentation

When working with TIBCO BusinessConnect™ Container Edition - ebXML Protocol, you may find it useful to read the documentation of the following TIBCO products:

- TIBCO BusinessConnect™ Container Edition
- TIBCO Enterprise Message Service™
- TIBCO ActiveMatrix BusinessWorks™

- TIBCO ActiveMatrix BusinessWorks™ Plug-in for BusinessConnect

How to Contact Support for TIBCO Products

You can contact the Support team in the following ways:

- To access the Support Knowledge Base and getting personalized content about products you are interested in, visit our product Support website.
- To create a Support case, you must have a valid maintenance or support contract with a Cloud Software Group entity. You also need a username and password to log in to the product Support website. If you do not have a username, you can request one by clicking **Register** on the website.

How to Join TIBCO Community

TIBCO Community is the official channel for TIBCO customers, partners, and employee subject matter experts to share and access their collective experience. TIBCO Community offers access to Q&A forums, product wikis, and best practices. It also offers access to extensions, adapters, solution accelerators, and tools that extend and enable customers to gain full value from TIBCO products. In addition, users can submit and vote on feature requests from within the TIBCO Ideas Portal. For a free registration, go to TIBCO Community.

Legal and Third-Party Notices

Cloud SG software may be available on multiple operating systems. However, not all operating system platforms for a specific software version are released at the same time. See the “readme” file for the availability of a specific version of Cloud SG software on a specific operating system platform.

THIS DOCUMENT IS PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS DOCUMENT COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THIS DOCUMENT. CLOUD SG MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S), THE PROGRAM(S), AND/OR THE SERVICES DESCRIBED IN THIS DOCUMENT AT ANY TIME WITHOUT NOTICE.

THE CONTENTS OF THIS DOCUMENT MAY BE MODIFIED AND/OR QUALIFIED, DIRECTLY OR INDIRECTLY, BY OTHER DOCUMENTATION WHICH ACCOMPANIES THIS SOFTWARE, INCLUDING BUT NOT LIMITED TO ANY RELEASE NOTES AND "README" FILES.

This and other products of Cloud SG may be covered by registered patents. For details, please refer to the Virtual Patent Marking document located at https://www.tibco.com/patents.