

*IBM WebSphere Business Integration Collaborations
for Consumer Products Item Synchronization
Version 4.4*



Solution Development Guide

Note!

Before using this information and the product it supports, be sure to read the general information under “Notices and Trademarks” on page 31.

Eighth Edition (July 2004)

This edition applies to Version 4, Release 4, *IBM WebSphere Business Integration Collaborations for Consumer Products Item Synchronization, Version 4.4.0* (5724-H62)

IBM welcomes your comments. You can send them to the following address:

IBM Corporation
Department 6R4A
11501 Burnet Road
Austin, TX 78578

Include the title of this book, and the page number or topic related to your comment.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright International Business Machines Corporation 2002, 2004. All rights reserved.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Solution development guide 1

Introduction	1
Who should read the Solution development guide	1
Overview	1
Collaboration template processes	2
Collaboration templates.	3
CommandRouter collaboration template	3
UCCnet_ItemSync collaboration template.	4
UCCnet_requestWorklist collaboration template	4
UCCnet_processWorklist collaboration template	4
Notify_by_eMail collaboration template	5
CIN_CIP_Dispatcher collaboration template	5
CI_Sync collaboration template	5
UCCnet_Price collaboration template	6
Standard process logic	6
Setting up a collaboration object	6
Business objects	16
Retail_Item business object	16
Retail_Price business object	17
UCCnetGBO_envelope business object	17
UCCnetXSD_envelope business object	17
UCCnetTPIXSD_envelope business object	18
UCCnetJMSXSD_envelope business object	18
Maps and data handlers	18

RCIR maps	18
Catalogue Item (CI) maps	19
Catalogue Item Notification (CIN) maps.	19
Catalogue Item Publication (CIP) map	20
Data flow	20
Retrieving worklists from UCCnet.	23
Sending information from collaboration objects to UCCnet.	23
Receiving data for a collaboration object from UCCnet.	23
Using the PROCESSED_GTIN table	24
Using the audit_log table.	25
Sending e-mail	26
Alerting e-mail recipients of processing errors	26
Sending e-mail through collaboration objects	26
Logging to an Interchange Server log file	29
Tracing	29

Notices and Trademarks 31

Notices	31
Programming interface information	32
Trademarks and service marks	33

Solution development guide

The Item Synchronization for Suppliers solution workflow is composed of business objects, collaboration objects, connectors, and maps. These basic components work together to enable supply-side trading partners to automatically add items to, update or delist items within, or withdraw items from UCCnet when item updates are made in their Enterprise Resource Planning (ERP) applications. When an update is made in a supplier's ERP system, item data is automatically validated, reformatted, and sent to the UCCnet standard registry. Suppliers can also communicate new or updated item information to subscribing trading partners via UCCnet. Thus, enterprise data is synchronized with item data sent outside the enterprise.

Introduction

Who should read the Solution development guide

The Solution development guide describes the internal processing of the Item Synchronization for Suppliers solution. It is intended for programmers who design and implement workflows using the solution and who might participate in designing customizations to this solution. It assumes that users are experienced programmers and that they understand the following concepts and have experience with the software associated with them:

- Developing collaboration objects, business objects, maps, and other related components.
- Installing, configuring, and operating the Item Synchronization for Suppliers solution.

Programmers must also have experience with the operating systems on which their implementations are installed.

Overview

The IBM® WebSphere® Business Integration Collaborations for Consumer Products Item Synchronization synchronizes item information in compliance with UCCnet standards. It is used to maintain the consistency of a supplier's item information in UCCnet and to ensure that the information is kept current with the most recent data in the supplier's application. The supplier can also communicate new or changed item information to subscribing trading partners in UCCnet.

The IBM WebSphere Business Integration Collaboration for Consumer Products Item Synchronization does these tasks through a set of component collaboration templates.

The way you communicate with UCCnet determines which templates and connectors you use and how you configure them. Items to consider include the following:

- How you connect to UCCnet. The following options are available:
 - AS2/EDIINT interface using Trading Partner Interchange (TPI) server
 - AS2/EDIINT interface using iSoft Peer-to-Peer Agent
 - AS2/EDIINT interface using WebSphere Business Integration Connect
 - AS2/EDIINT interface using WebSphere Business Integration Connect Express

- UCCnet Command Line Utility (CLU)
- Who provides the source data pool:
 - You Implement your own source data pool
 - UCCnet provides your source data pool

Because the connector you use to communicate with UCCnet is dependent on your setup, the term *AS2 channel* connector is used throughout this documentation as a general term for the TPIConnector, iSoftConnector, JMSConnector, or JTextConnector. Details on making these choices and implementing them are in the Installation Guide. The term *AS2 channel server* is used to refer to the iSoft Peer-to-Peer Agent, the TPI server, and WebSphere Business Integration Connect, as appropriate for the type of connectivity you are using.

If information in the documentation pertains only to a specific set of options, the following terminology is used:

Catalogue Item Notification or CIN operation

The information pertains only to configurations in which you implement your own source data pool.

Catalogue Item Publication or CIP operation

The information pertains only to configurations in which you allow UCCnet to provide the source data pool.

Collaboration template processes

The following sections illustrate the process logic for the collaboration templates included with the IBM WebSphere Business Integration Collaboration for Consumer Products Item Synchronization:

1. The CommandRouter collaboration is triggered by a Retail_Item with a notificationTopic of RCIR_ADD.
2. The CommandRouter forwards the Retail_Item to the UCCnet_ItemSync collaboration. The item is logged in the PROCESSED_GTIN table by gtin+fromGLN+targetMarket. The item is sent to UCCnet via the AS/2 connector with the appropriate registerCommand type (ADD, CHANGE, or CORRECT). If an item hierarchy is sent, only the top level item in the Retail_Item is registered. The item hierarchy is traversed, therefore, each item to be registered must be sent to the collaboration individually. A row in the PROCESSED_GTIN table for the item is added for an RCIR_ADD and updated for an RCIR_CHANGE and RCIR_CORRECT.

A row will be added to the AUDIT_LOG table for the RCIR command with the unique identifier value of the message sent to UCCnet.

UCCnet_processWorklist handles RCIR_RESPONSEs by finding the matching gtin+fromGLN+targetMarket row in the PROCESSED_GTIN table and updating the status to "Success" or "Fail", update the state to "REGISTERED", and update the timestamp.

3. The supplier then triggers the CommandRouter with a Retail_Item containing a notificationTopic of CI_ADD. This Retail_Item must contain the fully formed hierarchy, for example, Pallet >case >each. The CI_Sync collaboration takes the Retail_Item GBO and passes it through to an AS2 connector which then calls the map to create a publication command for a CI. This collaboration creates a log entry in the AUDIT_LOG Table.

UCCnet_processWorklist handles a CI_RESPONSE by creating a log entry in the AUDIT_LOG table and forwarding the response to the supplier's back end application

4. UCCnet_processWorklist processes a worklist by creating a new row in the AUDIT_LOG table for every CIC (Catalogue Item Confirmation) in the worklist. The entire UCCnetGBO_envelope, containing up to 150 CICs, is then be sent to the To port. The port is bound to a connector which makes the response available to the supplier's application. Any notifications in the worklist, other than a CIC, are sent to the Unsolicited_Response port. The Catalogue Item Confirmation responses are:

REVIEW

The retailer is reviewing the item.

REJECTED

The retailer has rejected the item. No additional information is requested at this time.

ACCEPTED

The retailer has accepted the item, but has not yet synchronized it. This state is similar to the DTD-based PRE-AUTHORIZATION state.

SYNCHRONISED

The retailer has accepted the item and synchronized it. This state is similar to a DTD-based AUTHORIZE state.

Collaboration templates

This section describes the collaboration templates.

CommandRouter collaboration template

The CommandRouter collaboration reduces the need for configuring multiple instances of an adapter to trigger each collaboration. The UCCnet_ItemSync, CI_Sync, and CIN_CIP_Dispatcher all require the Retail_Item as the triggering BO. Because you would not want to trigger three collaborations at the same time, they would require three separate instances of the input adapter. This collaboration will alleviate that need.

This collaboration can be used by the supplier to route a Retail_Item BO to the appropriate collaboration. The collaboration uses the item.notificationTopic.topic attribute to determine whether to route the Retail_Item to the UCCnet_ItemSync, CI_Sync, or CIN_CIP_Dispatcher collaborations. Because these collaborations are each triggered by a Retail_Item BO, this collaboration prevents the supplier from having to create separate instances of an adapter to uniquely trigger each of these destination collaborations.

Table 1 shows how the CommandRouter triggers the other collaborations based upon the topic

Table 1. CommandRouter topics and triggered collaborations

Topic	Collaboration
RCIR_ADD RCIR_CHANGE RCIR_CORRECT	UCCnet_ItemSync
CI_ADD CI_CHANGE CI_CORRECT	CI_Sync

Table 1. CommandRouter topics and triggered collaborations (continued)

Topic	Collaboration
PUBLISH_ADD	CIN_CIP_Dispatcher
PUBLISH_CHANGE	
PUBLISH_CORRECT	
PUBLISH_INITIAL_LOAD	
PUBLISH_WITHDRAW	

UCCnet_ItemSync collaboration template

An object based on the UCCnet_ItemSync collaboration template performs processing to add, change, or correct item registrations in the Global Registry via UCCnet Services. The UCCnet_ItemSync collaboration receives the Retail_Item GBO. The Retail_Item should contain one item. UCCnet_ItemSync checks for duplicates, creates an entry for the item to be registered in the PROCESSED_GTIN table, and a log entry in the AUDIT_LOG table.

UCCnet_requestWorklist collaboration template

An object based on the UCCnet_requestWorklist collaboration template sends query commands to UCCnet via the AS2 channel connector. These requests obtain the notifications in the supplier's worklist which result from previous item sync UCCnet messages and responses from trading partner retailers.. A UCCnet_requestWorklist collaboration object is triggered when it receives a UCCnetGBO_envelope business object from the JTextRWLConnector. The JTextRWLConnector sends this triggering business object after it polls an input file folder and discovers an XML message that contains a UCCnet worklist query command.

The mechanism by which the UCCnet worklist query command is placed in the input file folder is independent of the collaboration template. You must create this mechanism.

UCCnet_processWorklist collaboration template

An object based on the UCCnet_processWorklist collaboration processes messages from UCCnet This collaboration object is triggered when it receives a UCCnetGBO_envelope business object from an AS2 channel connector. The UCCnetGBO_envelope should contain notifications and asynchronous responses which result from previous item sync commands. It may also contain Initial_Item_Load requests.

The UCCnet_processWorklist determines what is in the UCCnetGBO_envelope and forwards it to one of its ports. If the response is a failure response, the UCCnet_GBO_envelope is forwarded to the Failure port. Unsolicited messages, for example, Initial_Item_Load requests, are forwarded to the Unsolicited_Response port. You can bind this port to an instance of the Notify_by_eMail collaboration. All other responses are sent to the To port.

When confirmation notifications are received from UCCnet, the collaboration property BATCH_RESPONSE is set to true or false. If true, then the entire message containing up to 150 confirmations is sent to the To port. If the property is false, UCCnet_processWorklist breaks the message up and sends the confirmation messages one at a time to the To port.

UCCnet_processWorklist creates a log entry in the AUDIT_LOG table and updates the PROCESSED_GTIN table for RCIR responses.

Notify_by_eMail collaboration template

An object based on the Notify_by_eMail collaboration template can be used by a UCCnet_processWorklist collaboration object to pass communications to configured e-mail addresses. It can also be used with UCCnet_ItemSync, CI_Sync, and CIN_CIP_Dispatcher for validation error messages.

CIN_CIP_Dispatcher collaboration template

An object based on the CIN_CIP_Dispatcher collaboration object can be set up to dispatch Catalogue Item Notifications (CIN operation) or Catalogue Item Publications (CIP operation) to a list of trading partners subscribed to the item categories. The collaboration properties: TO_UCCNETGBO_PROCESSING_MAP and GLN_ATTRIBUTE determine the configuration.

CIN Configuration

For CIN operation the TO_UCCNETGBO_PROCESSING_MAP contains the value Retail_item_to_UCCnetGBO_envelope_notifyCommand_catalogueItem. The GLN_ATTRIBUTE contains the value:

```
ROOT.body[0].transaction.command[0].notifyCommand.notifyCommandOperand.catalogueItemNotification
.catalogueItem.dataRecipient
```

For sending notifications, the dataRecipient must be populated at every level of the item hierarchy. The collaboration uses the CatalogueItemUtility class to do the recursive population of the dataRecipient information in the item hierarchy

CIP Configuration

For CIP operation the TO_UCCNETGBO_PROCESSING_MAP contains the value Retail_Item_to_UCCnetGBO_envelope_publicationCommand_CIP. The GLN_ATTRIBUTE contains the value:

```
ROOT.body[0].transaction.command[0].publicationCommand.publicationCommandOperand
.catalogueItemPublication.publishToGLN
```

The supplier must supply the list of retailers for each publication. The list of Retailers who should receive publications and notifications must be supplied in the Retail_Item.glnList.toGln array. The CIN_CIP_Dispatcher creates a notification/publication message for each gln in that list. A UCCnetGBO_envelope, containing either the notify or publication command and the target GLN from the GLN list, is sent to the AS/2 connector where the appropriate UCCnet_GBO_envelope_to_UCCnetXSD_envelope map is called.

CI_Sync collaboration template

An object based on the CI_Sync collaboration template is used to send the fully formed item hierarchy to the UCCnet data pool. The Catalogue Item document contains a base of 151 attributes that describe trade item data. The CI_Sync collaboration receives the Retail_Item GBO. The Retail_Item should contain a fully formed hierarchy, for example, . Pallet > case > each. The collaboration generates an entry in the AUDIT_LOG table. The collaboration then passes the Retail_Item to a second instance of the AS/2 connector for transmission to UCCnet. The second instance of the AS/2 connector is configured to invoke the appropriate map convert the Retail_Item to the UCCnetXSD_envelope containing the catalogue item.

UCCnet_Price collaboration template

An object based on the UCCnet_Price collaboration template is triggered when it receives a Retail_Price business object from the ERP connector. This collaboration validates that the item exists in the PROCESSED_GTIN table and is in a REGISTERED state. The collaboration also validates that the gln is present. The collaboration creates an entry in the AUDIT_LOG table after successfully sending the Retail_Price business object to the To port. The AS/2 connector calls a map to convert the Retail_Price to a UCCnetXSD_envelope containing the Price or Price Bracket command.

Standard process logic

All collaboration templates use the following standard processes for collaboration templates:

- Retrieve process
- USE_RETRIEVE process
- Filtering process
- Additional Retrieve process
- Email process for error handling

The item sync collaborations modify the standard e-mail error handling process. The collaborations never automatically fail the flow when errors are detected. The collaborations log all errors which will result in an e-mail being sent if the collaboration object has an e-mail notification address configured. The SEND_EMAIL and SEND_EMAIL_TO collaboration properties control whether errors are e-mailed and to what e-mail address list. If the error code generated by the collaboration is listed in the FAILURE_EXCEPTIONS collaboration property then the collaboration fails the flow. By default, there are no error codes included in the FAILURE_EXCEPTIONS collaboration property.

Setting up a collaboration object

To set up any collaboration template supplied with the IBM WebSphere Business Integration Collaboration for Consumer Products Item Synchronization as a standalone collaboration object, complete the following steps:

1. Create the collaboration object.
2. Bind the ports for the collaboration object:
 - “UCCnet_ItemSync collaboration template” on page 8
 - “UCCnet_Price collaboration template” on page 9
 - “CI_Sync collaboration template” on page 11
 - “UCCnet_processWorklist collaboration template” on page 13
 - “UCCnet_requestWorklist collaboration template” on page 12
 - “Notify_by_eMail collaboration template” on page 14
 - “CIN_CIP_Dispatcher collaboration template” on page 15
 - “CommandRouter collaboration template” on page 7
3. Set the configuration properties for the collaboration object.

The following sections include information on each collaboration template’s ports and configuration properties. For general information on creating collaboration objects, refer to the Collaboration Development Guide.

CommandRouter collaboration template

Table 2 provides information about the UCCnet_ItemSync collaboration template's ports. Use the ItemBasic business object for all ports.

Note: Because every port must be bound, bind the unused port to the PortConnector. Doing so indicates that the port is unused without causing the collaboration object to provide additional functionality.

Table 2. CommandRouter collaboration template's ports

Port name	Bound to	Function
DestinationAppRetrieve	PortConnector	Unused.
From	Supplier's choice of adapter	Receives the triggering Retail_item business object.
ToCIN_CIP	CIN_CIP_Dispatcher collaboration object	Passes the processed Retail_item business object.
ToCI	CI_Sync collaboration object	Passes the processed Retail_item business object.
ToRICR	UCCnet_ItemSync collaboration object	Passes the processed Retail_item business object.
Error	Supplier's choice of adapter	Send validation errors to the supplier

CommandRouter standard properties: This collaboration templates uses the following standard configuration properties for collaboration templates:

- ADDITIONAL_RETRIEVE
- 1_FILTER_ATTRIBUTE
- INFORMATIONAL_EXCEPTIONS
- 1_INCLUDE_VALUES
- 1_FAIL_ON_INVALID_VALUE
- CONVERT_UPDATE
- USE_RETRIEVE
- 1_EXCLUDE_VALUES
- SEND_EMAIL
- CONVERT_CREATE

Note: In this collaboration template, the SEND_EMAIL property specifies whether or not e-mail is sent to the address defined by the SEND_EMAIL_TO property. If you want e-mail to be sent, set the value of SEND_EMAIL to all. Otherwise, set the value to none.

CommandRouter specific properties: In addition to its standard configuration properties, this collaboration template has the following configuration properties:

Table 3. *CommandRouter specific properties*

Property name	Description	Required
SEND_EMAIL_TO	Defines the e-mail address list to which problems detected during execution of collaboration object processing are sent. This entry is for InterChange Server administrators. Do not confuse this property with the Email notification address field, which is configured from the Collaboration General Properties tab when creating a collaboration object.	No. Requires the Email Connector to run if e-mail is entered. The SEND_EMAIL property must also be set.
SEND_EMAIL	Specifies whether e-mail is sent to the addresses set in the SEND_EMAIL_TO property.	Yes, if you want to send e-mail. To send e-mail, set the value to all. Otherwise, set the value to none.
FAILURE_EXCEPTIONS	A comma separated list of error codes which should cause the collaboration to fail the flow.	No. The default value is none.

UCCnet_ItemSync collaboration template

Table 4 describes the UCCnet_ItemSync collaboration template's ports.

Note: Because every port must be bound, bind the unused port to the PortConnector. Doing so indicates that the port is unused without causing the collaboration object to provide additional functionality.

Table 4. *UCCnet_ItemSync collaboration template's ports*

Port name	Bound to	Function
DestinationAppRetrieve	PortConnector	Unused.
From	CommandRouter collaboration object	Receives the triggering Retail_Item business object.
To	AS/2 connector	Passes the processed Retail_Item business object to the specific AS2 channel connector.
Error	Supplier's choice of adapter	Sends validation errors to the supplier.

UCCnet_ItemSync standard properties: This collaboration templates uses the following standard configuration properties for collaboration templates:

- ADDITIONAL_RETRIEVE
- 1_FILTER_ATTRIBUTE
- INFORMATIONAL_EXCEPTIONS
- 1_INCLUDE_VALUES
- 1_FAIL_ON_INVALID_VALUE
- CONVERT_UPDATE
- USE_RETRIEVE
- 1_EXCLUDE_VALUES
- SEND_EMAIL
- CONVERT_CREATE

Note: In this collaboration template, the SEND_EMAIL property specifies whether or not e-mail is sent to the address defined by the SEND_EMAIL_TO property. If you want e-mail to be sent, set the value of SEND_EMAIL to all. Otherwise, set the value to none.

UCCnet_ItemSync specific properties: In addition to its standard configuration properties, this collaboration template has the following configuration properties:

Table 5. UCCnet_ItemSync specific properties

Property name	Description	Required
DATABASE_VALIDATION	Check PROCESSED_GTIN table contents before sending commands to UCCnet. If performing an "ADD" and value is true, then the item is considered a duplicate. If performing a "CHANGE" or "CORRECT" and the value is true, then the item must exist in the PROCESSED_GTIN table to be processed.	Yes. The default value is true.
DB_CONN_POOL_NAME	The name of the connection data pool used by the collaboration to access the PROCESSED_GTIN and AUDIT_LOG database tables.. The default value is ITEMSYNC_DATA..	Yes
FAILURE_EXCEPTIONS	A comma separated list of error codes which should cause the collaboration to fail the flow.	No. The default value is none.
SEND_EMAIL_TO	Defines the e-mail address list to which problems detected during execution of collaboration object processing are sent. This entry is for InterChange Server administrators. Do not confuse this property with the Email notification address field, which is configured from the Collaboration General Properties tab when creating a collaboration object.	No. Requires the Email Connector to run if e-mail is entered. The SEND_EMAIL property must also be set.
SEND_EMAIL	Specifies whether e-mail is sent to the addresses set in the SEND_EMAIL_TO property.	Yes, if you want to send e-mail. To send e-mail, set the value to all. Otherwise, set the value to none.
SEND_EMAIL_TO	Defines the e-mail address list to which problems detected during execution of collaboration object processing are sent. This entry is for InterChange Server administrators. Do not confuse this property with the Email notification address field, which is configured from the Collaboration General Properties tab when creating a collaboration object.	No. Requires the Email Connector to run if e-mail is entered. The SEND_EMAIL property must also be set.

UCCnet_Price collaboration template

Table 6 on page 10 describes the UCCnet_Price collaboration template's ports.

Note: Because every port must be bound, bind the unused port to the PortConnector. Doing so indicates that the port is unused without causing the collaboration object to provide additional functionality.

Table 6. UCCnet_Price collaboration template's ports

Port name	Bound to	Function
DestinationAppRetrieve	PortConnector	Unused.
From	Supplier's choice of adapter	Receives the triggering Retail_Price business object.
To	AS/2 connector	Passes the processed Retail_Price business object to the specific AS2 channel connector.
Error	Supplier's choice of adapter	Sends validation errors to the supplier.

UCCnet_Price standard properties: This collaboration templates uses the following standard configuration properties for collaboration templates:

- ADDITIONAL_RETRIEVE
- 1_FILTER_ATTRIBUTE
- INFORMATIONAL_EXCEPTIONS
- 1_INCLUDE_VALUES
- 1_FAIL_ON_INVALID_VALUE
- CONVERT_UPDATE
- USE_RETRIEVE
- 1_EXCLUDE_VALUES
- SEND_EMAIL
- CONVERT_CREATE

Note: In this collaboration template, the SEND_EMAIL property specifies whether or not e-mail is sent to the address defined by the SEND_EMAIL_TO property. If you want e-mail to be sent, set the value of SEND_EMAIL to all. Otherwise, set the value to none.

UCCnet_Price specific properties: In addition to its standard configuration properties, this collaboration template has the following configuration properties:

Table 7. UCCnet_Price specific properties

Property name	Description	Required
SEND_EMAIL_TO	Defines the e-mail address list to which problems detected during execution of collaboration object processing are sent. This entry is for InterChange Server administrators. Do not confuse this property with the Email notification address field, which is configured from the Collaboration General Properties tab when creating a collaboration object.	No. Requires the Email Connector to run if e-mail is entered. The SEND_EMAIL property must also be set.

Table 7. UCCnet_Price specific properties (continued)

Property name	Description	Required
SEND_EMAIL	Specifies whether e-mail is sent to the addresses set in the SEND_EMAIL_TO property.	Yes, if you want to send e-mail. To send e-mail, set the value to all. Otherwise, set the value to none.
DB_CONN_POOL_NAME	The name of the connection data pool used by the collaboration to access the PROCESSED_GTIN and AUDIT_LOG database tables.. The default value is ITEMSYNC_DATA..	Yes
FAILURE_EXCEPTIONS	A comma separated list of error codes which should cause the collaboration to fail the flow.	No. The default value is none.
PERFORM_VALIDATION	Specifies whether to check the PROCESSED_GTIN table for GTIN, GLN, and Target Market before processing the commands.	Yes. The default value is true.

CI_Sync collaboration template

Table 8 describes the CI_Sync collaboration template's ports.

Note: Because every port must be bound, bind the unused port to the PortConnector. Doing so indicates that the port is unused without causing the collaboration object to provide additional functionality.

Table 8. CI_Sync collaboration template's ports

Port name	Bound to	Function
DestinationAppRetrieve	PortConnector	Unused.
From	CommandRouter collaboration object	Receives the triggering Retail_Price business object.
To	AS/2 connector	Passes the processed Retail_Price business object to the specific AS2 channel connector.
Error	Supplier's choice of adapter	Sends validation errors to the supplier.

CI_Sync standard properties: This collaboration templates uses the following standard configuration properties for collaboration templates:

- ADDITIONAL_RETRIEVE
- 1_FILTER_ATTRIBUTE
- INFORMATIONAL_EXCEPTIONS
- 1_INCLUDE_VALUES
- 1_FAIL_ON_INVALID_VALUE
- CONVERT_UPDATE
- USE_RETRIEVE
- 1_EXCLUDE_VALUES
- SEND_EMAIL

- CONVERT_CREATE

Note: In this collaboration template, the SEND_EMAIL property specifies whether or not e-mail is sent to the address defined by the SEND_EMAIL_TO property. If you want e-mail to be sent, set the value of SEND_EMAIL to all. Otherwise, set the value to none.

CI_Sync specific properties: In addition to its standard configuration properties, this collaboration template has the following configuration properties:

Table 9. CI_Sync specific properties

Property name	Description	Required
SEND_EMAIL_TO	Defines the e-mail address list to which problems detected during execution of collaboration object processing are sent. This entry is for InterChange Server administrators. Do not confuse this property with the Email notification address field, which is configured from the Collaboration General Properties tab when creating a collaboration object.	No. Requires the Email Connector to run if e-mail is entered. The SEND_EMAIL property must also be set.
SEND_EMAIL	Specifies whether e-mail is sent to the addresses set in the SEND_EMAIL_TO property.	Yes, if you want to send e-mail. To send e-mail, set the value to all. Otherwise, set the value to none.
DB_CONN_POOL_NAME	The name of the connection data pool used by the collaboration to access the PROCESSED_GTIN and AUDIT_LOG database tables.. The default value is ITEMSYNC_DATA..	Yes
FAILURE_EXCEPTIONS	A comma separated list of error codes which should cause the collaboration to fail the flow.	No. The default value is none.

UCCnet_requestWorklist collaboration template

Table 10 describes the UCCnet_requestWorklist collaboration template's ports. Use the UCCnetGBO_envelope business object for all ports.

Note: Because every port must be bound, bind the unused port to the PortConnector. Doing so indicates that the port is unused without causing the collaboration object to provide additional functionality.

Table 10. UCCnet_requestWorklist collaboration template's ports

Port name	Bound to	Function
From	JTestRWLConnector	Receives the input UCCnetGBO_envelope business object.
To	AS/2 channel connector	Passes the processed UCCnetGBO_envelope business object to the AS2 channel connector, which transmits the data to UCCnet as a command by way of the AS2 channel server.connector.

UCCnet_requestWorklist standard properties: This collaboration template does not use standard configuration properties

UCCnet_requestWorklist specific properties: In addition to its standard configuration properties, this collaboration template has the following configuration properties:

Table 11. UCCnet_requestWorklist specific properties

Property name	Description	Required
SET_UNIQUE_IDS	Controls whether unique IDs (messageIdentifier and uniqueCreateorIdentification) are set in the output XML messages. Possible values for this property are: <ul style="list-style-type: none"> • ALL (the default value — set all three unique IDs) • NONE • BLANK (set unique ID only if it is blank in the input) 	No, unless the default value needs to be changed.
DTD_URL	Value to be copied into DTD DocType messages	No, not used

UCCnet_processWorklist collaboration template

Table 12 describes the UCCnet_processWorklist collaboration template's ports.

Table 12. UCCnet_processWorklist collaboration template's ports

Port name	Bound to	Function
From	AS/2 channel connector	Receive data from UCCnet
To	Supplier's connector of choice	Send responses to the supplier
Failure	Supplier's connector of choice	Send failed responses to the supplier
UnsolicitedResponse	Supplier's connector of choice	Send unsolicited responses to the supplier

UCCnet_processWorklist standard properties: This collaboration templates does not use standard configuration properties.

UCCnet_processWorklist specific properties: In addition to its standard configuration properties, this collaboration template has the following configuration properties:

Table 13. UCCnet_processWorklist specific properties

Property name	Description	Required
SEND_EMAIL_TO	Defines the e-mail address list to which problems detected during execution of collaboration object processing are sent. This entry is for InterChange Server administrators. Do not confuse this property with the Email notification address field, which is configured from the Collaboration General Properties tab when creating a collaboration object.	No. Requires the Email Connector to run if e-mail is entered. The SEND_EMAIL property must also be set.

Table 13. UCCnet_processWorklist specific properties (continued)

Property name	Description	Required
SEND_EMAIL	Specifies whether e-mail is sent to the addresses set in the SEND_EMAIL_TO property.	Yes, if you want to send e-mail. To send e-mail, set the value to all. Otherwise, set the value to none.
DB_CONN_POOL_NAME	The name of the connection data pool used by the collaboration to access the AUDIT_LOG database table.. The default value is ITEMSYNC_DATA..	Yes
FAILURE_EXCEPTIONS	A comma separated list of error codes which should cause the collaboration to fail the flow.	No. The default value is none.
BATCH_RESPONSE	Indicates whether the collaboration should send all notification responses in one batch or individually.	Yes. The default value is true.

Notify_by_eMail collaboration template

Table 14 describes the Notify_by_eMail collaboration template's ports.

Table 14. Notify_by_eMail collaboration template's ports

Port name	Bound to	Function
From	UCCnet_processWorklist collaboration object	A UCCnetGBO_envelope business object is passed in on this port. The business object is incorporated into an email message and sent out to configured email recipients.
FromItem	UCCnet_processWorklist collaboration object	A Retail_Item business object is passed in on this port. The business object is incorporated into an e-mail message and sent out to configured email recipients.

Notify_by_eMail standard properties: This collaboration templates does not use standard configuration properties.

Notify_by_eMail specific properties: In addition to its standard configuration properties, this collaboration template has the following configuration properties:

Table 15. Configuration properties specific to the Notify_by_eMail collaboration template.

Property name	Property description	Required
EMAIL_MESSAGE	Body of the e-mail. The processing of this field uses the same variables as the EMAIL_SUBJECT property. It also processes values beginning with the @ sign in the same manner. For example, in the following, the first character of the string is an @ sign, so the collaboration object loads the text from the filename following the @ sign: @c:\IBM\WebSphereICS\UCCnet\collaborations\eMail\ \UCCnet_processWorklist_AUTHORIZATION_RESPONSES.mail	Yes
EMAIL_NOTIFICATION_RCPTS	E-mail address of the recipients.	Yes

Table 15. Configuration properties specific to the *Notify_by_eMail* collaboration template (continued).

Property name	Property description	Required
EMAIL_SUBJECT	<p>Subject line of the e-mail. This value can contain variables in the form <code>\${variable_name}</code> into which the collaboration object substitutes data from the business object dynamically. Type text using the following escape codes:</p> <ul style="list-style-type: none"> • <code>\${getRoot}</code> — substitutes the entire triggering business object. • <code>\${getDate}</code> — substitutes the current date and time. • <code>\${getName}</code> — substitutes the name of the triggering business object. • <code>\${getVerb}</code> — substitutes the verb of the triggering business object. • <code>\${attribute}</code> — substitutes the value of the named attribute from the triggering business object. If the value for the <i>variable_name</i> does not match one of the specific values above, the collaboration object interprets it as the name of a business object attribute. <p>If the first character of the string for this value is an @ sign, the collaboration object loads the text from a filename following the @ sign.</p> <p>See “Sending e-mail” on page 26.</p>	Yes

CIN_CIP_Dispatcher collaboration template

Table 16 describes the CIN_CIP_Dispatcher collaboration template’s ports.

Note: Because every port must be bound, bind the unused port to the PortConnector. Doing so indicates that the port is unused without causing the collaboration object to provide additional functionality.

Table 16. CIN_CIP_Dispatcher collaboration template’s ports

Port name	Bound to	Function
DestinationAppRetrieve	PortConnector	Unused.
From	CommandRouter	Receives Retail_item business object on this port.
To	AS/2 channel connector	A UCCnetGBO_envelope business object is passed out on this port.
Error	Supplier’s connector of choice	Send validation errors.

CIN_CIP_Dispatcher standard properties: This collaboration templates uses the following standard configuration properties for collaboration templates:

- ADDITIONAL_RETRIEVE
- 1_FILTER_ATTRIBUTE
- INFORMATIONAL_EXCEPTIONS
- 1_INCLUDE_VALUES
- 1_FAIL_ON_INVALID_VALUE
- CONVERT_UPDATE
- USE_RETRIEVE
- 1_EXCLUDE_VALUES

- SEND_EMAIL
- CONVERT_CREATE

CIN_CIP_Dispatcher specific properties: In addition to its standard configuration properties, this collaboration template has the following configuration properties:

Table 17. Configuration properties specific to the CIN_CIP_Dispatcher collaboration template.

Property name	Property description	Required
DB_CONN_POOL_NAME	The name of the connection data pool used by the collaboration to access the AUDIT_LOG database table. The default value is ITEMSYNC_DATA.	Yes
GLN_ATTRIBUTE	Identifies where in the UCCnetGBO business object the GLN retrieved from the Dispatcher GLN file should be placed. Values to use: CIN operation Default. CIP operation	Yes
TO_UCCNETGBO_PROCESSING_MAP	Called by the CIN_CIP_Dispatcher object to map incoming item basic objects to the outgoing Catalogue Item Notification or Catalogue Item Publication message. Value: CIN operation Default. CIP operation	Yes
FAILURE_EXCEPTIONS	Specifies exception message numbers that would cause the collaboration object to fail flow	No. The default value is none.

Business objects

Retail_Item business object

This business object contains all the information required to describe the item including the necessary linking attributes to define fully formed item hierarchies. It also contains a fromGln attribute and a toGln attribute. This business object triggers the UCCnet_ItemSync, CommandRouter, CI_Sync, and CIN_CIP_Dispatcher collaboration objects. It contains data attributes that are common across all the logical organizations that use the item data.

The CommandRouter collaboration uses the Retail_Item.notification.topic to determine which collaboration to trigger.

A UCCnet_ItemSync collaboration object uses the Retail_Item business object to register items in the Global. Registry. At that point the Retail_Item business object contains only the item to be registered.

A CI_Sync collaboration object uses the Retail_Item to publish the fully formed hierarchy to the UCCnet data pool.

The CIN_CIP_Dispatcher collaboration object uses the Retail_Item business object to build CatalogueItemNotification messages or CatalogueItemPublication messages to send to trading partners.

When the Retail_Item is used for the CI_Sync collaboration and CIN_CIP_Dispatcher collaboration is must contain a fully formed item hierarchy.

When using any of the xsd extensions to describe the item, for example hardlines, produce, or fmcg, the Retail_Item's tradeItem.tradeItemExtensionType attribute must be specified

Retail_Price business object

This business object is used to trigger UCCnet_Price collaboration objects. It contains price or price bracket specific data that is used for issuing price commands to UCCnet.

UCCnetGBO_envelope business object

When a collaboration object exchanges data with an AS2 channel connector, the data must be mapped between the business object used by the connector and the UCCnetGBO_envelope business object used by the collaboration object. This business object is created and used as follows:

- A UCCnet_requestWorklist collaboration object uses this business object to exchange messages between the JTextRWLConnector and an AS2 channel connector. The JTextRWLConnector passes a UCCnetGBO_envelope business object to the UCCnet_requestWorklist collaboration object, which in turn passes it to the AS2 channel connector.
- A UCCnet_processWorklist collaboration object uses this business object to exchange messages with the AS2 channel connector. The UCCnetGBO_envelope business object is sent to any user defined connector for receiving responses. In addition, the UCCnetGBO_envelope can be used to trigger the Notify_by_eMail collaboration.

The UCCnet_ItemSync,UCCnet_Price,CI_Sync,CIN_CIP_Dispatcher, and UCCnet_processWorklist collaboration objects use this business object to communicate error information. The UCCnetGBO_envelope can then be forwarded to a connector of choice.

UCCnetXSD_envelope business object

This application specific business object is used by the AS2 channel connector when the XSD XML definition type and iSoft connectivity are used. Unlike the UCCnetGBO_envelope business object, it does not contain the TPIRouteInfo child business object.

When a message is received from UCCnet, the AS2 channel connector and the XML DataHandler create this business object from the received XML file. Before it is processed by the collaboration objects included with the solution, it must first be transformed into a UCCnetGBO_envelope business object by passing through the UCCnetXSD_envelope_to_UCCnetGBO_envelope map.

When a message is sent to UCCnet, the UCCnetXSD_envelope business object is the result of the conversion of either a Retail_Item or a UCCnetGBO_envelope business object by one of the associated maps configured in the AS2 channel connector.

UCCnetTPIXSD_envelope business object

This application specific business object is used by the AS2 channel connector when the XSD XML definition type and TPI connectivity are used.

When a message is received from UCCnet, the AS2 channel connector and the XML DataHandler create this business object from the received XML file. Before it is processed by the collaboration objects included with the solution, it must first be transformed into a UCCnetGBO_envelope business object by passing through the UCCnetTPIXSD_envelope_to_UCCnetGBO_envelope map.

When a message is sent to UCCnet, the UCCnetTPIXSD_envelope business object is the result of the conversion of either a Retail_Item or a UCCnetGBO_envelope business object by one of the associated maps configured in the AS2 channel connector.

The UCCnetTPIXSD_envelope business object contains the TPIRouteInfo child object. This object contains the senderId and receiverId required for a TPI server. The senderId is mapped from the Retail_Item.glnList.fromGln and the receiverId is hardcoded in the maps.

UCCnetJMSXSD_envelope business object

This application specific business object is used by the AS2 channel connector when the XSD XML definition type and WebSphere Business Integration Connect Advanced/Enterprise edition connectivity are used.

When a message is received from UCCnet, the AS2 channel connector and the XML DataHandler create this business object from the received XML file. Before it is processed by the collaboration objects included with the solution, it must first be transformed into a UCCnetGBO_envelope business object by passing through the UCCnetJMSXSD_envelope_to_UCCnetGBO_envelope map.

When a message is sent to UCCnet, the UCCnetJMSXSD_envelope business object is the result of the conversion of either a Retail_Item or a UCCnetGBO_envelope business object by one of the associated maps configured in the AS2 channel connector.

The UCCnetJMSXSD_envelope business object contains the JMSProperties child object. This object contains the senderId, receiverId, and other properties required for a WebSphere Business Integration Connect server. The senderId is mapped from the Retail_Item.glnList.fromGln and the receiverId is hardcoded in the maps.

Maps and data handlers

This section describes the maps and data handlers provided in the solution.

RCIR maps

The following maps are used to create the registerCommand:

Retail_Item_to_UCCnetXSD_envelope_registerCommand_itemAddChange
Retail_Item_to_UCCnetJMSXSD_envelope_registerCommand_itemAddChange
Retail_Item_to_UCCnetTPIXSD_envelope_registerCommand_itemAddChange

The following attributes are custom coded in the map:

RegisterCommand attribute	Custom value
registerCommandHeader.type	If Retail_Item.notificationTopic.topic RCIR_ADD = ADD RCIR_CHANGE = CHANGE RCIR_CORRECT = CORRECT
MessageHeader.messageIdentifier.value	MSGID+date/timestamp
classificationCategoryCode	additionalClassificationAgencyName + "." + additionalClassificationCategoryCode
registerCommandHeader.entityIdentification.uniqueCreatorIdentification	GTIN + TargetMarket
registryCatalogueItem.dateInformation.effectiveDate	Current date
.registryCatalogueItemIdentification.uniqueCreatorIdentification	UID3+date/timestamp

Also the following are mapped:

Item Mapped from

eanuccType

tradeItemIdentification.additionalItemIdentification.eanuccType

eanuccCode

tradeItemIdentification.additionalItemIdentification.eanuccCode

deliveryMethodIndicator

tradeItemIdentification.additionalItemIdentification.deliveryMethodIndicator

Catalogue Item (CI) maps

The following maps are used to create the publicationCommand containing a catalogueItem:

Retail_Item_to_UCCnetXSD_envelope_publicationCommand_catalogueItem
Retail_Item_to_UCCnetJMSXSD_envelope_publicationCommand_catalogueItem
Retail_Item_to_UCCnetTPIXSD_envelope_publicationCommand_catalogueItem

The following attributes are custom coded in the map:

PublicationCommand attribute	Custom value
publicationCommandHeader.type	If Retail_Item.notificationTopic.topic CI_ADD = ADD CI_CHANGE = CHANGE CI_CORRECT = CORRECT
MessageHeader.messageIdentifier.value	MSGID+date/timestamp
publicationCommandHeader.entityIdentification.uniqueCreatorIdentification	GTIN + TargetMarket

Catalogue Item Notification (CIN) maps

The following map is used to create the notifyCommand

Retail_Item_to_UCCnetGBO_envelope_notifyCommand_catalogueItem

The following attributes are custom coded in the map:

notifyCommand attribute	Custom value
notifyCommandOperand.catalogueItemNotification.notificationTopic.topic	If Retail_Item.notificationTopic.topic PUBLISH_ADD = NEW_ITEM PUBLISH_CHANGE = DATA_CHANGE PUBLISH_CORRECT = CORRECTION PUBLISH_WITHDRAW = WITHDRAW
MessageHeader.messageIdentifier.value	MSGID+date/timestamp
catalogueItemNotification.isReload	Set to true if topic is PUBLISH_INITIAL_ITEM_LOAD else set to false
notifyCommandHeader.entityIdentification.uniqueCreatorIdentification	GTIN + TargetMarket
catalogueItemNotification.creationDate	Current date
CatalogueItemNotificationIdentification.uniqueCreatorIdentification	UID1+date/timestamp

Catalogue Item Publication (CIP) map

The following map is used to create the publicationCommand

Retail_Item_to_UCCnetGBO_envelope_publicationCommand_CIP

The following attributes are custom coded in the map:

publicationCommand attribute	Custom value
publicationCommandHeader.type	If Retail_Item.notificationTopic.topic PUBLISH_ADD = ADD PUBLISH_INITIAL_LOAD = IS_RELOAD PUBLISH_WITHDRAW = DELETE
MessageHeader.messageIdentifier.value	MSGID+date/timestamp
catalogueItemPublicationIdentification.uniqueCreatorIdentification	UID2+date/timestamp
messageHeader.creationDate	Current date
publicationCommandHeader.entityIdentification.uniqueCreatorIdentification	UID1+date/timestamp
catalogueItemPublication.creationDate	Current date

Data flow

As the data flows through the workflows, the form of the information changes. Sometimes it is stored in a generic business object, sometimes in an application specific business object, and sometimes, the data is contained in an XML message. As part of the processing, data passes through a variety of transformations. Normally, these transformations are done to prepare the data for the next step in the workflow, whether it be a collaboration object, a connector, or the AS2 channel server.

The following table shows how the format of the information changes as it moves between the connectors, collaboration objects, and other parts that make up a workflow. The **Transform initiator** and **Time initiated** columns indicate the connector or collaboration object that initiated the data transform, and the point in the workflow that it does this. The **Input** and **Output** columns respectively indicate the format of the data before and after the transformation takes place. The **Transformer** column indicates the map or data handler used to carry out the

transformation. This example illustrates TPI connectivity.

Table 18. Transformers used for XSD support using TPI connectivity as an example. This table indicates how the format of information changes as it moves between parts of a workflow. It also indicates what transformers make these changes. Information for the TPI Connector controller and TPI Connector agent is also applicable for the JTextTPI Connector controller and JTextTPI Connector agent.

Transform initiator	Time initiated	Input	Output	Transformer
TPI Connector controller	Sending information to UCCnet to register item in Global Registry	Retail_Item generic business object (GBO)	UCCnetTPIXSD_envelope application specific business object (ASBO)	Retail_Item_to_UCCnetTPIXSD_envelope_registerCommand_itemAddChange
TPI Connector controller (second instance)	Sending information to UCCnet to Create the Catalogue	Retail_Item GBO	UCCnetTPIXSD_envelope ASBO	Retail_Item_to_UCCnetTPIXSD_envelope_publicationCommand_catalogueItem
TPI Connector controller	Receiving information from UCCnet	UCCnetTPIXSD_envelope ASBO	UCCnetGBO_envelope GBO	UCCnetTPIXSD_envelope_to_UCCnetGBO_envelope map
TPI Connector controller	Sending information to UCCnet	UCCnetGBO_envelope GBO	UCCnetTPIXSD_envelope ASBO	UCCnetGBO_envelope_to_UCCnetTPIXSD_envelope map
TPI Connector agent	Sending information to UCCnet	UCCnetTPIXSD_envelope ASBO	XML message in UCCnet format	IBM WebSphere Business Integration DataHandler for XML
TPI Connector agent	Receiving information from UCCnet	XML message in UCCnet format	UCCnetTPIXSD_envelope ASBO	IBM WebSphere Business Integration DataHandler for XML
JTextRWLConnector Connector agent	Initiating request for UCCnet to return a workload	XML message in UCCnet format	UCCnetTPIXSD_envelope ASBO	IBM WebSphere Business Integration Data Handler for XML
JTextRWL Connector controller	Initiating request for UCCnet to return a workload	UCCnetTPIXSD_envelope ASBO	UCCnetGBO_envelope GBO	UCCnetTPIXSD_envelope_to_UCCnetGBO_envelope map
CIN_CIP_Dispatcher collaboration object	Sending CIN or CIP requests to UCCnet	Retail_Item GBO	UCCnetGBO_envelope GBO	CIN operation map Retail_Item_to_UCCnetGBO_envelope_notifyCommand_catalogueItem CIP operation map
UCCnet_Price collaboration object	Sending Price/Price Bracket requests to UCCnet	Retail_Price GBO	UCCnetGBO_envelope GBO	Retail_Item_to_UCCnetGBO_env_publicationCommand_catalogueItemPublication Retail_Price_to_UCCnetTPIXSD_envelope map

Retrieving worklists from UCCnet

In many of the workflows, UCCnet generates, but does not automatically return, a worklist containing a response to a message received from the supplier. In these cases, the supplier must request the worklist from UCCnet.

Retrieval of a worklist from UCCnet happens as follows:

1. The supplier uses a time-triggered process to move a worklist query command to the event directory of the JTextRWLConnector: This movement process is not a part of the solution, and must be customized by the supplier.
2. The JTextRWLConnector polls its event directory for new query commands at user-defined time intervals.
3. When the JTextRWLConnector finds a query command in the event directory, it sends it to the IBM WebSphere Business Integration Data Handler for XML, which converts it to an application specific business object (ASBO).
4. The JTextRWLConnector passes the business object through a map to convert it to a UCCnetGBO_envelope GBO and then passes it to a UCCnet_requestWorklist collaboration object.
5. The UCCnet_requestWorklist collaboration object, in turn sends the business object to the AS2 channel connector for delivery to UCCnet. See “Sending information from collaboration objects to UCCnet” for details.

Sending information from collaboration objects to UCCnet

Sending messages from a collaboration object to UCCnet:

In many of the workflows, a collaboration object needs to send information to UCCnet. The following set of events accomplishes this action:

1. The collaboration object sends a generic business object (GBO) to the AS2 channel connector.
2. The connector controller calls the associated map configured for this GBO to convert it into an application specific business object (ASBO).
3. The connector controller passes the ASBO to the connector agent.
4. The connector agent calls the IBM WebSphere Business Integration Data Handler for XML which converts the ASBO into a UCCnet formatted XML message.
5. The AS2 channel connector agent then passes the XML message to the AS2 channel server, which creates the digest, encrypts, and transmits the message to UCCnet.

Receiving data for a collaboration object from UCCnet

Receiving messages back from UCCnet:

In many of the workflows, UCCnet will send a message to the AS2 channel server to be passed on to a collaboration object. The following set of events accomplishes this action:

1. UCCnet sends an XML message to the AS2 channel server.
2. The AS2 channel connector agent retrieves the message from the AS2 channel server.

3. The connector agent calls the IBM WebSphere Business Integration Data Handler for XML to convert the message into an application specific business object (ASBO).
4. The connector agent passes the ASBO to the connector controller.
5. The connector controller calls the associated map configured for this ASBO to convert it into a generic business object (GBO).
6. The AS2 channel connector then passes the GBO to all collaboration objects that subscribe to the GBO.

Using the PROCESSED_GTIN table

The PROCESSED_GTIN table is a persistent record of which gtins have been registered with the Global Registry. This table is used to detect duplicate registration attempts and attempts to change or correct unregistered items. The key to this table is the GTIN+GLN + TargetMarket.

During the installation process, you create the ITEMSYNC database. The PROCESSED_GTIN table is created by running the supplied Create_ItemSync_Tables.sql file for the database type (DB2[®], Oracle, or Microsoft[®]) SQL Server). See the Installation guide for installation instructions. The UCCnet_ItemSync,UCCnet_processWorklist, and UCCnet_Price collaborations objects connect to the database through the collaboration property, DB_CONN_POOL_NAME.

Table 19. Fields in the PROCESSED_GTIN table

Field	Definition	Example value
GTIN	Global Trade Item Number	00005743116018
FROMGLN	Supplier's GLN	7789333000026
TARGMRKT	Target Market	840
DTCREATE	Creation date	2004060921433EDT
DTUPDATE	Update date	2004060921605EDT
COMMAND	The last command processed for the gtin	RCIR_ADD
STATE	The state of the gtin (REGISTERED)	REGISTERED
STATUS	The status of the command (SENT, Success, Failure)	Success
ERRCODE	Error code if status is Failure	
ERRDATA	Error description if status is Failure	

The UCCnet_ItemSync writes to the Processed_Gtin table for any add, change, or correct action. The collaboration checks the table to determine existence and status of the gtin before performing any action

Add

- A row is added if the GTIN does not exist
- A row is added if the GTIN exists and the state is REGISTERED and the DatabaseValidation collaboration property is false.

Change/Correct

- The row is updated if the GTIN exists

- A row is added if the GTIN does not exist and the DatabaseValidation collaboration property is false. (The command stays as a Change or Correct)

The UCCnet_processWorklist writes to the Processed_Gtin table when receiving a response from UCCnet. The collaboration uses the gtin+gln+targetMarket to find the row matching the response and changes the dtupdate, status, state, errcode and errdata.

The UCCnet_Price collaboration checks the Processed_Gtin table to determine the state of the gtin as a validation check before generating a price document. This collaboration does not update any fields for that gtin.

Using the audit_log table

The AUDIT_LOG table is a history log of every operation performed for each gtin and the status of that operation. It includes every one of the following commands and their results:

- RCIR command
- CI command
- CIP/CIN command
- Price command

An application can use this table to determine the history of operations performed on each gtin. If the table is pruned it might not contain an accurate record of registered items.

The rows in this table have no key because multiple rows can exist which contain operations on the same items. The MSGID field can be used to match a command to its response. Responses to CIN and CIP commands in the form of CIC worklist notifications result in a separate row logged in the table.

The AUDIT_LOG table does not contain the message payload or processing errors. You can use the trace feature to obtain this information.

During the installation process, you create the ITEMSYNC database. The AUDIT_LOG table is created by running the supplied Create_ItemSync_Tables.sql file for the database type (DB2, Oracle, or Microsoft SQL Server). See the *Installation guide* for installation instructions.

Each collaboration object connects to the database through the collaboration property, DB_CONN_POOL_NAME.

Table 20. Fields in the AUDIT_LOG table

Field	Definition	Example value
MSGID	message ID assigned to the transaction	10868152632320
GTIN	Global Trade Item Number	00005743116018
FROMGLN	Supplier's GLN	7789333000026
TARGMRKT	Target Market	840
TOGLN	The publish to gln (filled in on publication and notifications)	1270257111116
DTCREATE	Creation date	2004060921433EDT

Table 20. Fields in the AUDIT_LOG table (continued)

Field	Definition	Example value
DTUPDATE	Update date	
COMMAND	The last command processed for the gtn	PUBLISH_ADD
STATE	The state of the gtn (REGISTERED)	
STATUS	The status of the command (SENT, Success, Failure)	SENT
ERRCODE	Error code if status is Failure	
ERRDATA	Error description if status is Failure	
PRICEID	ID to correlate price and price bracket	

Sending e-mail

E-mail can be generated by two processes:

- Collaboration objects can be configured to send e-mail to alert if a processing error occur. See “Alerting e-mail recipients of processing errors.”
- Collaborations can use the the Notify_by_eMail collaboration template on any of the collaboration Error ports, or any port of the UCCnet_processWorklist. See “Sending e-mail through collaboration objects.”

Alerting e-mail recipients of processing errors

Collaboration objects can be configured to send e-mail to alert recipients when processing errors occur. The following configuration properties control whether e-mail is sent and to specify the recipients:

SEND_EMAIL

Controls whether e-mail is sent to the e-mail address specified in the SEND_EMAIL_TO configuration property. Set the property value to all to send e-mail or to none to not send e-mail. If the value is left empty, no e-mail is sent even if recipients exist in the SEND_EMAIL_TO property.

SEND_EMAIL_TO

Defines the e-mail addresses to which error messages are sent. Multiple addresses can be provided in a comma-delimited list. You must define these addresses.

Sending e-mail through collaboration objects

The collaboration object contains an Error port which is used to deliver validation errors. E-mails can be sent to a set of configured addresses by instantiating a collaboration object based on the Notify_by_eMail collaboration template..

The Notify_by_eMail collaboration template can be configured to contain the e-mail message, subject, and recipients specific to its processing situation through its EMAIL_MESSAGE, EMAIL_SUBJECT, and EMAIL_NOTIFICATION_RCPTS configuration properties, respectively. These properties can also contain the names of files, which permits messages, subjects, and recipients to be shared among multiple collaboration objects. Also, more than one recipient can be specified to receive e-mail through use of a comma-delimited list. Plus, e-mail message and

subject text can be constants that contain variables. The Notify_by_eMail collaboration object substitutes data from the business object into these variables dynamically. See the following sections for more information on these features:

- “Specifying message text, subjects, and recipients in external files”
- “Specifying changing individual or multiple message recipients”
- “Using substitution variables in message and subject text” on page 28

Specifying message text, subjects, and recipients in external files

A Notify_by_eMail collaboration object allows the contents of its properties that specify e-mail message text, subject text, and recipients to contain the names of files. These files contain the actual e-mail message text, subject text, and addresses, and can be easily modified without modifying the using collaboration objects. This feature permits messages, subjects, and recipients to be shared among multiple collaboration objects. A solution’s messages, subjects, and recipients can all be contained in one easily modifiable directory.

A Notify_by_eMail collaboration object uses the following configuration properties to identify the e-mail message text, subject text, and recipients:

EMAIL_MESSAGE

Identifies the message text.

EMAIL_SUBJECT

Identifies the subject text.

EMAIL_NOTIFICATION_RCPTS

Identifies the recipient or list of recipients.

The collaboration object distinguishes whether the content of a property is an actual value or filename based on whether the value is prefixed by the character @. If the value of the property is prefixed with the character @, the Notify_by_eMail collaboration object interprets the rest of the value as a filename. The collaboration object reads the value of the file into a String variable in preparation for further processing. Files must be identified by their fully qualified names.

For instance, if the filename containing the e-mail recipients is `c:\Email_Files\CategoryManagerRole.txt`, set the value of the EMAIL_NOTIFICATION_RCPTS property, as follows:

```
@c:\Email_Files\CategoryManagerRole.txt
```

If the value of a property does not start with the character @, the Notify_by_eMail collaboration object obtains the e-mail value directly from the attribute.

Specifying changing individual or multiple message recipients

A Notify_by_eMail collaboration object allows all e-mail messages to be routed to an administrator or to a specific role in an organization (like a Category Manager), without the need to maintain the e-mail recipient’s fully qualified e-mail address in every collaboration object that might send e-mail. By placing the e-mail address in an external file, if the address changes, the file can be modified without having to reconfigure the using collaboration objects. More than one recipient can be specified to receive the e-mail through use of a comma-delimited list. The comma-delimited list can be specified in the business object attribute or in the external file pointed to by the attribute.

Using substitution variables in message and subject text

Email message and subject text can be constants that contain variables. A collaboration object based on the Notify_by_eMail template substitutes data from the business object into these variables dynamically. Variables to be substituted must be enclosed in the prefix characters `${` and the suffix character `}`. As a result, the substitution variables in the e-mail message and subject text must appear as:

`${variable_name}`

Note: These characters might have to be changed to meet National Language requirements.

The supported values for *variable_name*, along with the values that the collaboration object actually inserts in the text, are as follows:

getRoot

Substitutes the entire triggering business object.

getDate

Substitutes the current date and time.

getName

Substitutes the name of the triggering business object.

getVerb

Substitutes the verb of the triggering business object.

Any attribute name

Substitutes the value of the named attribute from the triggering business object.

If the value for *variable_name* does not match one of the specific values above, the collaboration object interprets it as the name of a business object attribute. For instance, in the following sample message:

```
UCCnet_processWorklist_Failure_RESPONSES.mail:      \
Date:  ${getDate}
BusinessObject:  ${getName}.${getVerb}
Topic:
${ROOT.body[0].response.acknowledge.acknowledgement.      \
subdocumentValid[0].subdocumentValid[0].resultList[0].      \
notification.topic}
GLN:
${ROOT.body[0].response.acknowledge.acknowledgement.      \
subdocumentValid[0].subdocumentValid[0].resultList[0].      \
notification.notificationDetail.transactionInformation.      \
entityIdentification.globalLocationNumber.gln}
GTIN:
${TLO.body.body_Wrapper1[0].response.acknowledge.      \
acknowledgement.subdocumentValid[0].subdocumentValid[0].      \
resultList.resultList_Wrapper1[0].notification.      \
notificationDetail.authorizationNotification.publication.      \
item.itemInformation.globalTradeItemNumber.gtin}

${getRoot}
```

The following variables are filled in automatically during the generation of the message, as follows:

- `${getDate}`, with the current date and time.
- `${getName}`, with the name of the triggering business object.
- `${getVerb}`, with the verb of the triggering business object.

- All variables beginning with `${ROOT.body[0]. . .}`, with the values for those attributes.
- `${getRoot}` with the entire triggering business object.

Logging to an Interchange Server log file

If a collaboration object encounters error situations during any stage of processing, it does the following:

- Logs the error in the configured log destination.
- Returns the object to the calling collaboration object through the From port.

Note: For Interchange Server error logging to occur, tracing must be enabled. Also, use separate files for tracing and logging. Use logging files to maintain persistent records of processed data. Use tracing files to diagnose problems and to show the flow of an item through the Item Synchronization for Suppliers solution. The Log Viewer tool has log and trace file filters that enable users to view the log or trace records for a particular business object or collaboration object.

Tracing

All collaboration objects based on collaboration templates included in the Item Synchronization for Suppliers solution provide tracing capabilities to record logical flows and data processed. Users can enable tracing for a particular collaboration object by selecting the collaboration object in the System Manager, displaying its properties, and, on the **Collaboration General Properties** tab, selecting a trace level greater than 0 from the **System trace level** field.

Enable tracing for one or more collaboration objects when a reproducible problem occurs. If a problem occurs only once during processing, leave the tracing function enabled continually so that the first occurrence of the failure is captured. However, leaving the tracing function enabled continually can degrade performance. Clear the trace file periodically to simplify viewing and filtering it.

Note: Use separate files for tracing and logging. Use tracing files to diagnose problems and to show the flow of an item through the Item Synchronization for Suppliers solution. Use logging files to maintain persistent records of processed data. The Log Viewer tool has trace and log file filters that enable users to view the trace or log records for a particular business object or collaboration object.

Notices and Trademarks

Proprietary Information

US Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Notices

IBM may not offer the products, services, or features discussed in this document in all countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created

programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Burlingame Laboratory Director
IBM Burlingame Laboratory
577 Airport Blvd., Suite 800
Burlingame, CA 94010
U.S.A

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not necessarily tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information may contain examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples may include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

COPYRIGHT LICENSE

This information may contain sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Programming interface information

Programming interface information, if provided, is intended to help you create application software using this program.

General-use programming interfaces allow you to write application software that obtain the services of this program's tools.

However, this information may also contain diagnosis, modification, and tuning information. Diagnosis, modification and tuning information is provided to help you debug your application software.

Warning: Do not use this diagnosis, modification, and tuning information as a programming interface because it is subject to change.

Trademarks and service marks

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States or other countries, or both:

IBM
the IBM logo
DB2
WebSphere

Microsoft is a trademark of Microsoft Corporation in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.