amazon services

Selling on Amazon: Guide to Data Exchange

Table of Contents

Supported Data-Exchange Methods: AMTU and SOAP

Sellers and Amazon.com can exchange data through seller-customized Application Programming Interfaces (APIs). The APIs help sellers quickly develop applications for integration with their own point-of-sale and fulfillment systems. Amazon.com supports two data-exchange methods for sending feeds and retrieving reports (both also known as documents) as alternatives to using manual processes in Seller Central.

The first method, and the easiest to implement, is AMTU (Amazon Merchant Transport Utility). This utility is compatible with Windows, Mac, and Linux operating systems. Once AMTU is installed, a service runs in the background and automatically searches for documents waiting to be sent to or pulled from Amazon.com. In other words, it handles the bi-directional communication of documents between Amazon.com and the seller. The seller copies outbound feeds into a specified folder and AMTU posts them to Amazon.com. When processing is complete, the utility pulls a processing report from Amazon.com, providing the seller with detailed processing results.

The second method is SOAP (Simple Object Access Protocol). For this method, the seller develops applications that use predefined APIs to send or receive feeds, processing status reports, and acknowledgments for downloaded documents.

> Note:
>
> Both methods support XML feeds or text-file feeds as long as they adhere to the predefined formats.
>
> With AMTU you won't have to develop the SOAP transport layer, resulting in lower cost and faster deployment (in most cases).

This guide covers SOAP in depth. For more information about AMTU, go to http://amtu.sourceforge.net.

SOAP Prerequisites

Before you decide to implement SOAP, make sure you meet the prerequisites.

Ask yourself, do I have development resources who can:

1. write the SOAP transport layer using Java, PHP or a similar scripting language,
2. understand Web Services, and
3. interpret a WSDL?

Using SOAP to Exchange Data

HTTPS Transports for Exchanging Data

Data exchange is always initiated by the seller using a secure internet connection using HTTPS (Hyper Text Transfer Protocol Secure). This allows information between the seller and Amazon.com to be encrypted and secure.

When a feed is posted, the data is unencrypted, the session is authenticated and the seller is identified using credentials provided in the HTTPS header. An acknowledgment response is returned to the seller with a status code. The actual feed that was attached in the message is then processed. When processing is complete, a more detailed report is generated for the seller to retrieve.

All of the APIs are formed as XML messages and transmitted over the internet using HTTPS connections. For document exchange, the XML message contains two main parts--the header and the payload (or data package). The header includes seller-specific information (for example, username and password) and the payload is the actual document being exchanged.

Encoding Attachments using DIME or MIME

The payload is wrapped by the SOAP message, and is encoded using either DIME (Dual Independent Map Encoding) or MIME (Multipurpose Internet Mail Extensions) depending upon the seller's preferred programming environment. The Microsoft .NET platform encodes data using DIME and most other platforms use MIME. Amazon.com supports both encoding methods; however, your settings will differ based upon the chosen method.

WSDL

Amazon.com uses a WSDL (Web Services Definition Language) to describe how to communicate and lists the functions available through APIs.

The latest WSDLs are available here:

DIME: https://merchant-api.amazon.com/gateway/merchant-interface-dime/

DIME (.NET): https://sellercentral.amazon.com/help/merchant_documents/WSDL/merchant-interface-dime-dotnet.wsdl

MIME: https://merchant-api.amazon.com/gateway/merchant-interface-mime/

Basic Functions Supported

Every function is described as a remote procedure call with the following format:

ReturnObject method (**Argument** *argument1*, **Argument** *argument2*)

Method: method of execution (for example, *postDocument, which* is used to send Amazon.com a document for processing)

Argument: abstract data type that defines an input for the "Method" (for example, seller of record)

ReturnObject: abstract data type that defines the return "Argument" for the "Method" (for example, DocumentSubmissionResponse, which is an output argument that would, for example, return the condition "Success")

There are four basic functions supported by Amazon.com:

1. Posting documents to Amazon.com (for example, posting a product feed containing the seller's new products)
2. Getting processing status for posted documents (for example, retrieving the processing results of a posted product feed)
3. Getting documents from Amazon.com (for example, retrieving pending order reports for the last hour)
4. Posting acknowledgements for downloaded documents (for example, acknowledging the successful download of specific order reports so they will no longer display as pending retrieval)

Note

A complete transaction cycle requires two separate functions:

1) The action of posting or getting a document

2) The acknowledgment or confirmation of processing (getting the processing status or posting an acknowledgment of a successful download)

Posting Documents

Supported Document Types

Document Type Supported	Valid Message Type
Product data	_POST_PRODUCT_DATA_
Item data	_POST_ITEM_DATA_
Product inventory	_POST_INVENTORY_AVAILABILITY_DATA_
Product pricing	_POST_PRODUCT_PRICING_DATA_
Product images	_POST_PRODUCT_IMAGE_DATA_
Product relationships	_POST_PRODUCT_RELATIONSHIP_DATA_
Product shipping overrides	_POST_PRODUCT_OVERRIDES_DATA_
Order acknowledgement	_POST_ORDER_ACKNOWLEDGEMENT_DATA_
Order fulfillment	_POST_ORDER_FULFILLMENT_DATA_
Order adjustment	_POST_PAYMENT_ADJUSTMENT_DATA_

Applicable APIs

postDocument – This is used to post documents to Amazon.com.

Input Arguments

Parameter Name	Parameter Type	Required	Note
merchant	Merchant	Yes	Seller of record
messageType	MessageType	Yes	Accepted message types: _POST_PRODUCT_DATA_ _POST_ITEM_DATA_ _POST_INVENTORY_AVAILABILITY_DATA_ _POST_PRODUCT_PRICING_DATA_ _POST_PRODUCT_IMAGE_DATA_ _POST_PRODUCT_RELATIONSHIP_DATA_ _POST_PRODUCT_OVERRIDES_DATA_ _POST_ORDER_ACKNOWLEDGEMENT_DATA_ _POST_ORDER_FULFILLMENT_DATA_ _POST_PAYMENT_ADJUSTMENT_DATA_
attachedDocument	String	Yes	Actual document (encoded)

Output Arguments

Parameter Type	Return Condition	Note
DocumentSubmissionResponse	Success	Acknowledgment of successful receipt of this message.
Fault	Failure	Possible faults are: _INVALID_MESSAGE_TYPE_ _UNRECOGNIZED_MERCHANT_ _MISSING_OR_INVALID_DATA_ _INTERNAL_ERROR_

getDocumentProcessingStatus – This is used to retrieve the processing status of a posted document.

Input Arguments

Parameter Name	Parameter Type	Required	Note

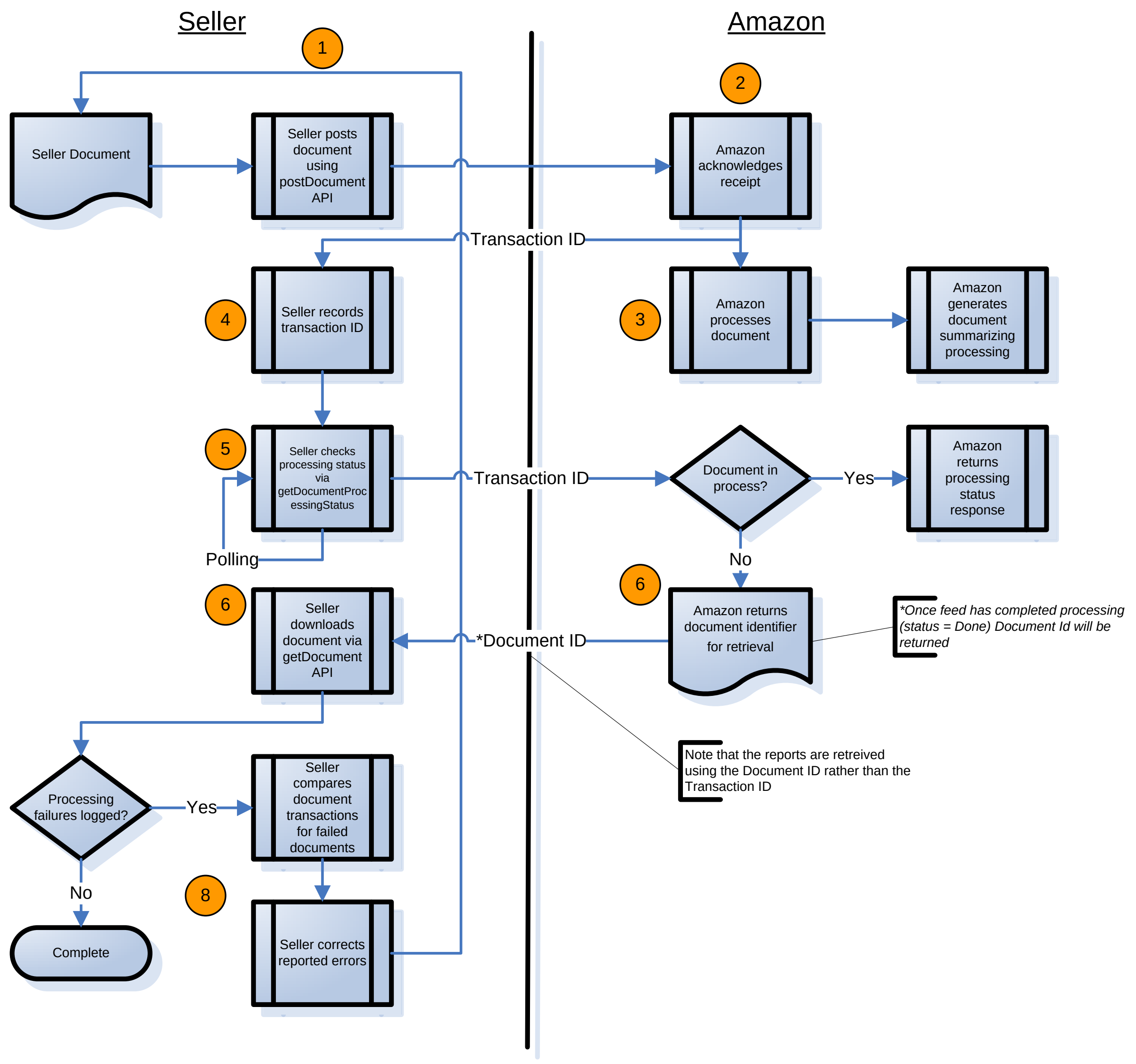
Parameter Name	Parameter Type	Required	Note
merchant	Merchant	Yes	Seller of record
documentTransactionIdentifier	DocumentTransactionID	Yes	A transaction reference ID is returned to the seller after Amazon.com successfully receives the inbound document for processing.

Output Arguments

Parameter Type	Return Condition	Note
DocumentProcessingInfo	Success	The Transaction ID is used for polling the processing report. When processing is done, the Document ID is returned. Possible values are: _PENDING_ _IN_PROGRESS_ _DONE_ _FAILED_DUE_TO_FATAL_ERRORS
Fault	Failure	Possible faults are: _INVALID_MESSAGE_TYPE_ _UNRECOGNIZED_MERCHANT_ _INVALID_DOCUMENT_TRANSACTION_IDENTIFIER_ _MISSING_OR_INVALID_DATA_ _INTERNAL_ERROR_

Choreography

When you post a document using SOAP, it is important to follow the correct choreography. To help simplify the process, we have outlined the recommended steps below.


Seller
Amazon
1
2
Seller Document
Seller posts document using postDocument API
Amazon acknowledges receipt
Transaction ID
4
Seller records transaction ID
3
Amazon processes document
Amazon generates document summarizing processing
5
Seller checks processing status via getDocumentProcessingStatus
Transaction ID
Document in process?
Yes
Amazon returns processing status response
Polling
No
6
Seller downloads document via getDocument API
*Document ID
6
Amazon returns document identifier for retrieval
*Once feed has completed processing (status = Done) Document Id will be returned
Note that the reports are retreived using the Document ID rather than the Transaction ID
Processing failures logged?
Yes
Seller compares document transactions for failed documents
No
8
Complete
Seller corrects reported errors

1. Seller posts the document (postDocument): Seller creates a document (such as an Order Fulfillment feed) and posts it using the postDocument API.

2. Amazon.com acknowledges the document: Amazon.com returns an acknowledgement receipt that the document was received and provides a transaction ID.

3. Amazon.com processes the document: Amazon.com processes the document and summarizes the result.

4. Seller records the transaction ID: The transaction ID is the ID for the transfer and receipt of the document. Note that this is not the same as the document ID returned when the posted document has completed processing.

5. Seller checks to see if there is a document processing report ready to download: Seller polls the document processing status by using the getDocumentProcessingStatus API and the transaction ID is returned as described in Step 2, above.

6. Amazon.com communicates the document ID status: If the document has not yet been fully processed, the status "Pending" is returned. When the document is completely processed, the status "Done" is returned along with the document ID for retrieving the processing results (also known as the processing report).

7. Seller retrieves the processing report: The seller uses the getDocument API to retrieve the document using the document ID, returned in Step 6, above.

8. Seller corrects any errors found: If there are errors listed in the processing report, the seller corrects the problems and re-sends the document (Step 1). If there are no errors, processing is complete.

Retrieving Documents

Supported Document Types

Document Type Supported	Valid Message Type
Order report	_GET_ORDERS_DATA_
Settlement report	_GET_PAYMENT_SETTLEMENT_DATA_

Handshake

When retrieving documents from Amazon.com, it is important to note the following:

1. Every document is uniquely identified by a document ID.
2. The seller chooses the document from the set (array) of pending documents.
3. The seller requests a document using a specified document ID.
4. Download failures are reported back using fault codes.

Applicable APIs

getAllPendingDocumentInfo – This is used to retrieve an array of pending documents.

Input Arguments

Parameter Name	Parameter Type	Required	Note
merchant	Merchant	Yes	Seller of record
messageType	MessageType	Yes	Accepted message types: _GET_ORDERS_DATA_ _GET_PAYMENT_SETTLEMENT_DATA_

Output Arguments

Parameter Type	Return Condition	Note
DocumentInfoArray	Success	An array of DocumentInfo objects. Provides information about the documents being requested by the seller.
Fault	Failure	Possible faults are: _INVALID_MESSAGE_TYPE_ _UNRECOGNIZED_MERCHANT_ _MISSING_OR_INVALID_DATA_ _INTERNAL_ERROR_

getDocument – This is used to retrieve a pending document.

Input Arguments

Parameter Name	Parameter Type	Required	Note
merchant	Merchant	Yes	Seller of record
documentIdentifier	DocumentID	Yes	Identifier returned as part of the DocumentInfo object

Output Arguments

Parameter Type	Return Condition	Note
Bytes	Success	AttachedDocument is the actual document requested. Note that the format is dependent on the binding that is used to send/receive messages.
Fault	Failure	Possible faults are: _INVALID_MESSAGE_TYPE_ _UNRECOGNIZED_MERCHANT_ _MISSING_OR_INVALID_DATA_ _INTERNAL_ERROR_

postDocumentDownloadAck – This is used to remove documents from the list of pending documents. (After you successfully retrieve an order report or settlement report, use this API to ensure that retrieved documents will no longer appear in the array of pending documents.)

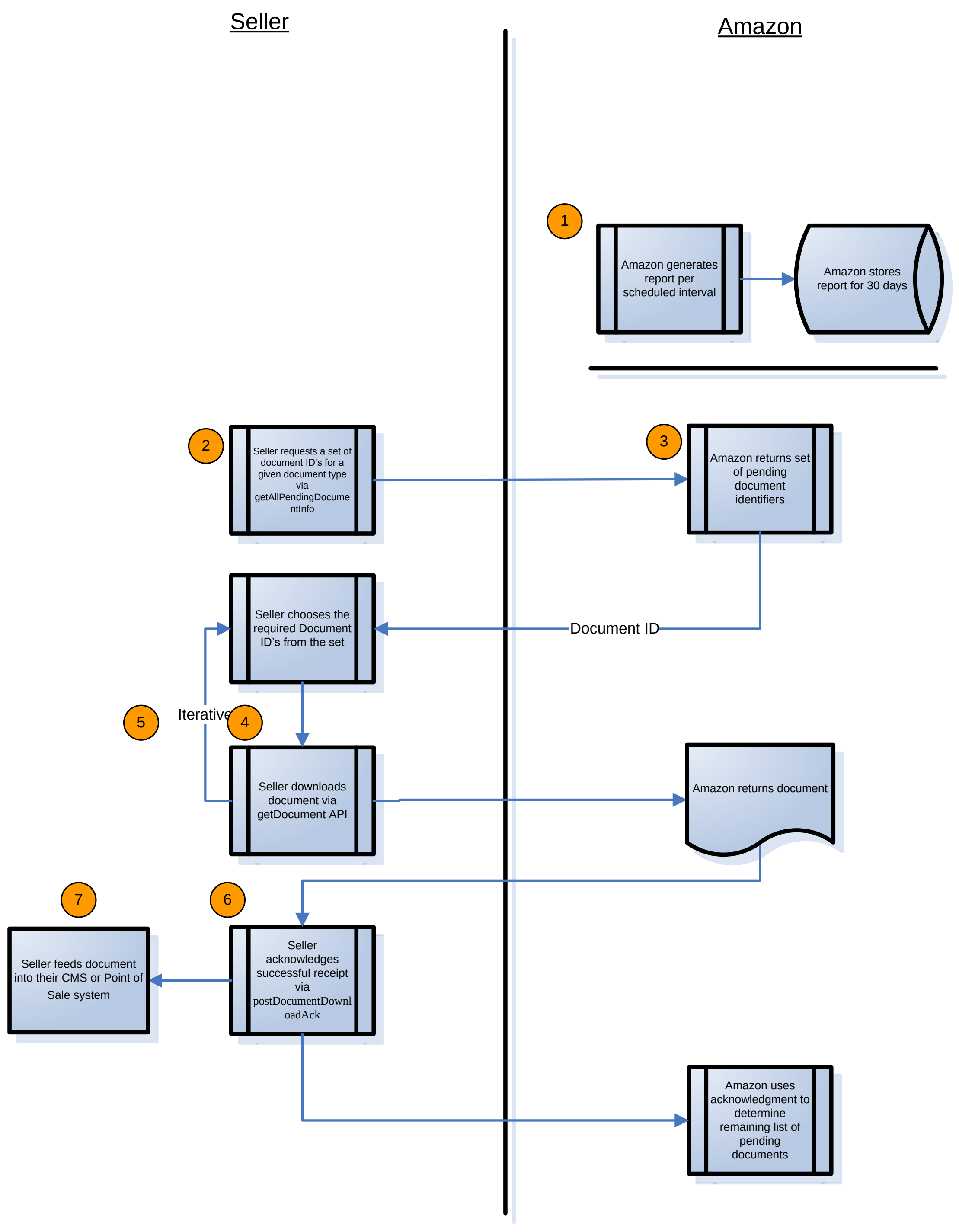
Input Arguments

Parameter Name	Parameter Type	Required	Note
merchant	Merchant	Yes	Seller of record
DocumentIdentifierArray	DocumentIDArray	Yes	The array of document IDs whose successful download is being acknowledged.

Output Arguments

Parameter Type	Return Condition	Note
DocumentDownloadAckStatusArray	Success	Provides acknowledgment processing status for each document. Possible values are: _SUCCESSFUL_ _ACCESS_TO_DOCUMENT_DENIED_ _INVALID_DOCUMENT_IDENTIFIER_ _INTERNAL_ERROR_
Fault	Failure	Possible faults are: _UNRECOGNIZED_MERCHANT_ _MISSING_OR_INVALID_DATA_ _INTERNAL_ERROR_

Choreography


Seller
Amazon
1
Amazon generates report per scheduled interval
Amazon stores report for 30 days
2
Seller requests a set of document ID's for a given document type via getAllPendingDocumentInfo
3
Amazon returns set of pending document identifiers
Document ID
Seller chooses the required Document ID's from the set
5
Iterative
4
Seller downloads document via getDocument API
Amazon returns document
7
6
Seller feeds document into their CMS or Point of Sale system
Seller acknowledges successful receipt via postDocumentDownloadAck
Amazon uses acknowledgment to determine remaining list of pending documents

1. Amazon.com generates reports: Amazon.com generates reports and stores them for 30 days.

2. Seller requests documents: Seller requests an array of document IDs waiting to be pulled for a given document type using the getAllPendingDocumentInfo API.

3. Amazon.com returns an array: Amazon.com returns an array of pending document IDs.

4. Seller chooses a document: Seller chooses a document to retrieve and requests the document using the getDocument API.

5. Iteration: Seller iterates until all desired documents are retrieved.

6. Seller acknowledges documents: Seller removes the documents from the pending retrieval list by acknowledging receipt using the postDocumentDownloadAck API.

7. Seller consumes document: Seller uses the document in their CMS or Point-of-Sale system.

Appendix

Fault Codes

Fault Code	Return Condition	Note
01	_UNRECOGNIZED_MERCHANT_	Cannot recognize the seller specified in the method call.
02	_INVALID_MESSAGE_TYPE_	The specified message type is not supported or is invalid for use with the method call.
03	_MISSING_OR_INVALID_DATA_	Data provided for the method call is invalid.
04	_INTERNAL_ERROR_	Unrecognizable internal error found while handling this method call.
05	_INVALID_DATE_RANGE_	One or both of the dates provided in the method call are invalid.
06	_INVALID_INTEGER_	The integer provided in the method call is invalid.
07	_ACCESS_TO_DOCUMENT_DENIED_	The requested document is protected and cannot be accessed in the method call.
08	_DOCUMENT_NO_LONGER_AVAILABLE_	The requested document is no longer available.
09	_INVALID_DOCUMENT_TRANSACTION_IDENTIFIER_	The Document ID provided in the method call is invalid.

Additional APIs

API	Use	Note
getLastNDocumentProcessingStatus	Returns array of SummaryInfo objects that contain the number of documents processed and the number of documents with errors.	Parameters: Merchant – Seller of record MessageType – type of document sought howMany – number of documents sought

Code Samples

Note

These code samples are for informational purposes only and might not be suitable for your particular environment and/or SOAP client. Amazon.com makes no representation as to their accuracy.

JAVA: https://images-na.ssl-images-amazon.com/images/G/01/cba/documents/cba-merchantAtAPIs-java.zip

PHP: https://images-na.ssl-images-amazon.com/images/G/01/cba/documents/cba-merchantAtAPIs-php.zip