PUBLIC

Document Version: 2605 – 2026-05

Invoicing Data Import and Administration Guide

SAP Ariba Buying and Invoicing
SAP Ariba Invoice Management
SAP Ariba Contract Invoicing

© 2026 SAP SE or an SAP affiliate company. All rights reserved.

Content

Invoicing Data Import and Administration Guide. 7

Getting Started with Invoicing Data Import and Administration 8

About Invoice Exceptions and Reconciliation. 9

About Approvers for Invoice Exceptions. 9

Rejection Reason Codes for Invoices. 10

Enabling and Configuring Rejection Reason Codes for Invoices. 10

Rejection Reason Codes in the User Interface and Email Notifications. 12

About invoice exception types and conditions. 13

Default invoice exception types. 14

Applying invoice exception types to specific types of invoices. 20

Restricting dispute on invoice exceptions. 21

How to restrict users from disputing an invoice exception. 22

Defining invoice exception types for specific invoice source documents. 23

Invoice exception type overrides. 28

Tolerance calculations. 29

About validation behavior. 32

Skipping validation for an invoice exception type or override. 32

About invoice exception ranking. 34

About standard line categories. 35

Procurement line types. 35

Maintaining invoice exception types in Ariba Administrator. 36

Adding or modifying invoice exception types. 37

About configuring invoice exception type overrides. 41

Applying an override to an invoice exception type. 42

How to configure an invoice exception type to be skipped. 43

How to view or delete invoice exception types. 44

Data import tasks for invoice exception types. 45

About import invoice exception types. 46

About the **Import Invoice Exception Type Validation Data** task. 47

About import invoice exception overrides. 48

Forcing reconciliation to change the status of an invoice. 50

How to re-reconcile an invoice reconciliation against updated invoice exception types. 51

Holding line-item credit and debit memos until the ERP system receives the invoice. 51

How to configure exception types for price-adjustment debit and credit memos. 53

Configuring exceptions that check for pending credit memos. 54

Configuring goods receipt-based invoice verification. 55
How to delay invoice reconciliation for goods receipt-based invoices. 59
Invoice exception types for goods receipt-based invoices. 61
Setting goods receipt-based invoice verification for items based on supplier location. 62
Integration support for goods receipt-based invoice verification. 63

Managing Payment Methods. 68
About Maintaining Payment Methods in Ariba Administrator. 68
Viewing Payment Methods. 68
Adding or Modifying Payment Methods. 69
Deleting Payment Methods. 70
About Importing Payment Methods. 70
About Translations for Payment Methods. 72

Managing Supplier Payment Information. 73
Remittance Locations and Payment Models. 73
External System Payment Model. 75
Data Relationships of Supplier Payment Information. 76
Supplier Location Matching on Invoices. 80
Examples of Remittance Information. 80
Maintaining Supplier Payment Information in Ariba Administrator. 82
Viewing Supplier Payment Information. 82
Adding or Modifying Supplier Remittance Locations. 82
Adding or Modifying Supplier Payment Methods. 83
Adding or Modify Supplier Payment Models for Suppliers. 84
Adding or Modify a Payment Model for a Supplier Location. 85
About Importing Remit To Information. 86
Import Remittance Location Information. 86
Import Supplier Location Remittance Information. 87

Managing Funds Management Accounting Information. 88
Funds Management Accounting Functionality. 88
Enabling the Funds Management Accounting Feature. 88
Prerequisites for the Funds Management Accounting Feature. 89
Configuring Funds Management Accounting. 89
Enabling Web Services Integration Tasks for Funds Management Account Assignment Fields. 90
Importing Flexible Master Data Templates for Funds Management Account Assignment Fields 90

Configuring Messaging. 95
Import Messaging Topics Task. 95
Import Conversation Context Rules for Messaging Task. 96
Configuring the Messaging Setup. 104

About Importing Service Sheets. 106
Importing Service Sheets Using Various Integration Channels. 107
Importing Service Sheets from SAP ERP-Integrated Sites Using SAP Integration Suite, managed gateway for spend management and SAP Business Network 109

How to Import External cXML Invoices to SAP Ariba Procurement solutions 110
cXML Guidelines for Importing External Invoices to SAP Ariba Procurement solutions. 113
cXML Requirements for Importing Invoices. 113
cXML Invoice Header for Importing External Invoices to SAP Ariba Procurement solutions 114
cXML Invoice Request Element. 116
Invoice Submission Method for Importing External cXML Invoices. 117
Tax Information cXML. 117
Required and Optional Calculated Totals cXML. 119
Attaching the Digital Image of a Paper Invoice. 119
Invoice Data to Include When Importing External cXML Invoices. 122
Invoice-to-cXML Data Mapping. 124
Sample cXML for a PO-Based Invoice. 138
Sample cXML for a Non-PO Invoice. 141
Sample cXML for a Contract-Based Invoice. 146
Creating an Approval Rule for External cXML Invoices. 150

About Exporting Invoices. 152
Invoice Data Export Task. 153
Exporting Invoices. 153
Invoice Data in Generic Variants. 154
About Invoice Header Data in Generic Variants. 154
About Invoice Detail Data in Generic Variants. 155
About Invoice Split Accounting Data in Generic Variants. 156
Invoice Data in SAP Variants. 157
About Payment Header Data in SAP Variants. 157
About Invoice Detail Data in SAP Variants. 158
About Invoice Split Accounting Data in SAP Variants. 160
Invoice Data in PeopleSoft Variants. 161
About Payment Header Data in PeopleSoft Variants. 161
About Invoice Detail Data in PeopleSoft Variants. 162
About Invoice Split Accounting Data in PeopleSoft Variants. 163

Exporting Payment Requests in SAP Ariba Buying and Invoicing 165
About Exporting Payment Requests in SAP Ariba Buying and Invoicing 165
About Payment Request Data Export Tasks. 166
Exporting Payment Requests Manually. 168
PeopleSoft Payment Request Data. 168
Payment Header Data in the PeopleSoft Variant. 169

Payment Detail Data in the PeopleSoft Variant. 170
Payment Bank Data in the PeopleSoft Variant. 171
Payment Split Accounting Data in the PeopleSoft Variant. 172
Advance Payment Detail Data in the PeopleSoft Variant. 173
SAP Payment Request Data. 174
Payment Header Data in the SAP Variant. 174
Payment Line Item Data in the SAP Variant. 175
Payment Account Data in the SAP Variant. 177
Payment Detail Data in the SAP Variant. 178
Payment Tax Export Data in the SAP Variant. 180
Advance Payment Detail Data in the SAP Variant. 181
Generic Payment Request Data. 181
Payment Header Data in the Generic Variant. 182
Payment Bank Export Data in the Generic Variant. 183
Payment Detail Data in the Generic Variant. 184
Payment Split Accounting Data in the Generic Variant. 186
Advance Payment Detail Data in the Generic Variant. 187
Payment Detail with Split Accounting Data in the Generic Variant. 188
Amount Precision in Payment Request Export Files. 189
Columns Affected by Amount Precision. 190

About Importing Remittances. 191
Introduction to Remittance Information. 191
Import Remittances Data Import Task. 193
About Importing Remittances Asynchronously. 195

About Importing and Exporting Advance Payment Data. 196
About Importing Document Types for Advance Payments. 199
Import Document Types for Advance Payment: Sample File. 199
About Importing Translations for Advance Payment Document Types. 199
Import Translations for Advance Payment Document Types: Sample File. 200
About Importing GL Indicators for Advance Payment. 200
Import GL Indicators for Advance Payment: Sample File. 200
About Importing Translations for Advance Payment GL Indicators. 201
Import Translations for Advance Payment GL Indicators: Sample File. 201
About Exporting Advance Payments. 201
Exporting Advance Payments Using the File Channel. 202
Exporting Advance Payments Using the Web Services Channel. 205
About Exporting Payment Requests with Advance Payments. 209
Export Payment Requests with Advance Payments: Sample Files - SAP ERP. 209
Export Payment Requests with Advance Payments: Sample Files - PeopleSoft ERP. 219
Export Payment Requests with Advance Payments: Sample Files - Generic ERP. 224

About Importing Advance Payment ID. 230
Import Advance Payment ID: Sample File. 231
About Importing Advance Payment Errors. 231
Import Advance Payment Errors: Sample File - Generic ERP Systems. 232
Import Advance Payment Errors: Sample File - SAP ERP Systems. 232
Import Advance Payment Errors: Sample File - PeopleSoft ERP Systems. 233
About importing advance payment remittance. 234
Import Advance Payment Remittance: sample file. 235
About Exporting Cancel Advance Payments. 237
Exporting Cancel Advance Payments Using the File Channel. 237
Exporting Cancel Advance Payments Using the Web Services Channel. 238
About Importing Cancel Advance Payment ID. 239
Import Cancel Advance Payment ID: Sample File. 240
About Importing Cancel Advance Payment Errors. 240
Import Cancel Advance Payment Errors: Sample File. 240
About Importing Advance Payment Cancellation Reasons. 242
Import Advance Payment Cancellation Reasons: Sample File. 243
About Exporting Advance Payment Cancellation Reasons. 243
Export Advance Payment Cancellation Reasons: Sample File. 243

Configuring Support for One-Time Vendor Invoicing. 245
Non-PO Invoices for One-Time Vendors. 245
One-Time Vendors in SAP Ariba Solutions. 246
Changes to Invoicing Functionality When One-Time Vendor Invoicing Is Enabled. 249
Configuration Overview for One-Time Vendor Invoicing. 251
Configuring a One-Time Vendor in SAP Ariba Buying and Invoicing. 256
Configuring SAP Business Network to Support One-Time Vendor ICS Invoices. 258
Supplier and Payment Information on One-Time Vendor Invoices. 258
Multiple One-Time Vendor Invoices for the Same Supplier. 260
Preparing ICS Providers for One-Time Vendor Invoices. 262
ICS Exceptions for One-Time Vendor Invoices. 268
Configuring Approval Rules for One-Time Vendor Invoices. 268
Integration with External Systems. 270
Creating a Non-PO Invoice for a One-Time Vendor. 273

Invoicing and Payment Data CSV Files Reference. 275

Cancellation of Approved Invoices in the ERP System. 278
Configuring a Web Service for Auto-Rejecting Approved Invoices Canceled in the ERP System. 282
Configuring an ERP SOAP Request to Send Invoice Status Updates to SAP Ariba Buying and Invoicing 283

Resending Invoice Documents to SAP Business Network. 285

Invoicing Data Import and Administration Guide

This guide is for SAP Ariba administrators responsible for configuring invoice exception types, managing payment and funds management information, and importing and exporting invoicing data.

Buyers process invoices from suppliers in order to pay for goods and services.

This guide applies to:

- SAP Ariba Buying and Invoicing
- SAP Ariba Invoice Management
- SAP Ariba Contract Invoicing

Related guides

Invoicing and Payment Process Guide

Common Data Import and Administration for SAP Ariba Procurement Solutions

Administration and Data Maintenance Fundamentals

Approval Process Management Guide

Creating and Managing Invoices

Reconciling Invoices

Managing Contract Compliance

SAP Business Network Guide to Invoicing

Tax and Charge Configuration

Getting Started with Invoicing Data Import and Administration

Invoicing data includes master data such as supplier payment information and invoice exception configurations as well as transactional data such as exporting approved invoices for payment.

Before you begin, make sure you read the Administration and Data Maintenance Fundamentals to learn about the data import process in general, entering data in comma-separated-value (CSV) files, downloading and using the Data Dictionary, and using Ariba Administrator.

Most of the master data setup is common in all SAP Ariba Procurement solutions. In general, you define and import common data before defining and importing invoicing and payment data. See the Common Data Import and Administration for SAP Ariba Procurement Solutions for master data import and administration that apply to all SAP Ariba Procurement solutions.

Some data objects discussed here can be imported as part of the **Import Batch Data** data import task, but if you are planning to manually import single files, make sure you import them in the expected order. Many files require that other information has already been loaded to 'link' information together using unique names. See the `ImportBatchData` tab in the data dictionary for the order of import events.

About Invoice Exceptions and Reconciliation

The invoicing process can result in discrepancies between the data on the supplier invoice and the order, contract, or receipt data in SAP Ariba Procurement solutions. These invoice exceptions are handled through a reconciliation process.

An invoice is a request for payment, typically sent by a supplier. Invoices are usually created by the supplier, to reflect an amount due for a purchase order or a contract. In the most typical case, invoices are sent electronically to your company as cXML documents, over SAP Business Network.

For each invoice, the reconciliation process creates an additional approvable document, called an invoice reconciliation document, that lists any invoice exceptions on that document. An invoice exception represents a single discrepancy between the data on the supplier invoice and the order, contract or receipt data in SAP Ariba Procurement solutions. Exceptions can represent a variety of issues, such as missing receipts, mismatched quantities or prices, duplicate invoices, or tax variances.

About Approvers for Invoice Exceptions [page 9]

Rejection Reason Codes for Invoices [page 10]

About invoice exception types and conditions [page 13]

Maintaining invoice exception types in Ariba Administrator [page 36]

Data import tasks for invoice exception types [page 45]

Forcing reconciliation to change the status of an invoice [page 50]

How to re-reconcile an invoice reconciliation against updated invoice exception types [page 51]

Holding line-item credit and debit memos until the ERP system receives the invoice [page 51]

How to configure exception types for price-adjustment debit and credit memos [page 53]

Configuring exceptions that check for pending credit memos [page 54]

Configuring goods receipt-based invoice verification [page 55]

About Approvers for Invoice Exceptions

In the default configuration, each invoice exception type is associated with a group, which determine the list of users on the approval flow for each invoice reconciliation document.

For example, in the default configuration, users who are members of the:

- **Tax Manager** group can reconcile tax exceptions
- **Invoice Manager** group can reconcile unmatched invoice exceptions
- **Receiving Manager** group can reconcile receiving invoice exceptions
- **Invoice Agent** group can reconcile contract invoice exceptions

The default approval rules use the groups associated with exception types to determine the set of approvers. The approval rules look over the invoice reconciliation document, find all exceptions on that document, and define as

approvers those users who have permission to reconcile the specified exceptions. These approvers are typically the first approvers on the flow, and are the users who are expected to reconcile any listed exceptions.

SAP Ariba Invoice Management and SAP Ariba Buying and Invoicing include a standard set of invoice exception types.

Related Information

Maintaining Invoice Exception Types in Ariba Administrator [page 36]
Approval Process Management Guide

Rejection Reason Codes for Invoices

Reason rejection codes provide a framework for buyers to develop a structured approach to capturing and communicating why an invoice was rejected. If you enable and configure reason rejection codes, users who reject an invoice must select one or more rejection reasons from a list of rejection reasons you define.

Without reason rejection codes, when an invoice agent or an invoice approver rejects an invoice, they can enter a free-form comment visible to the supplier explaining the rejection. While comments to suppliers for rejected invoices can be made mandatory, different users might use different explanations to explain the invoice rejection, which can be confusing to suppliers. Furthermore, such a free-form format does not allow for development of reports and metrics to understand and improve invoice quality issues.

Enabling and Configuring Rejection Reason Codes for Invoices

Before enabling the feature for your users by setting the parameter **Require reason codes for invoice rejection** (`Application.Invoicing.UseInvoiceRejectionReasonCodes`) to **Yes**, you use data import tasks to load your rejection reason codes. Once you have enabled this feature, users must select a rejection reason before they can submit their invoice rejection.

> **Note**
>
> If you are currently requiring supplier comments for invoice rejections, and you implement the rejection reason codes for invoices, you might want to make comments to suppliers optional.

You use the **Import Invoice Rejection Reasons** data import task to define your rejection reasons. This task reads data from the CSV file named `InvoiceRejectionReason.csv`. **ReasonCode** must have unique values.

Following is an example of `InvoiceRejectionReason.csv`:

```
UTF8
ReasonCode,Description
A100,Price is different from negotiated price for one or more line items.
```

```
B200,Tax amount on invoice is different from expected tax.
C300,Invoiced and received quantity do not match for one or more line items.
D400,We found miscellaneous issues with this invoice.
```

→ Tip

- There is no limit to the number of rejection reasons you can load. However, we recommend to keep the list manageable to drive user compliance. In particular, if your site is configured for email approval/rejection, the rejection email includes all available rejection reasons for users to choose.
- Rejection reasons are ordered in the UI and in emails by the reason code you provide using text-based ordering. For example, RR_1, RR_2, and RR_12 are ordered RR_1, RR_12, RR_2. If the order in which the rejection reasons are presented to the user is important, make sure you use values for the reason codes that provides the expected order.
- Do not use colons (:) or commas (,) in the **ReasonCode** field. Colons and commas are used by SAP Ariba solutions as text separators in the UI. Using these characters can lead to confusing text display for users.

Translations for invoice rejection reasons

You can use the **Import Invoice Rejection Reasons Translations** data import task to import translations for invoice rejection reason. The language for your users is determined by the locale settings for their account. The language for your suppliers is determined based on the country/region in the supplier location's address. If no translations are available for that geographical location, the default language is used (the language you used to define the rejection reasons using the **Import Invoice Rejection Reasons** data import task).

Following is an example of a CSV file of invoice rejection reason translations:

```
UTF8
ReasonCode,Description,Language
A100,Der Preis fuer eins oder mehrere Teile ist unterschiedlich vom verhandelten
Preis.,German
A100,Le prix est différent du prix négocié pour un ou plusieurs éléments.,French
A100,El precio es diferente del precio negociado para una o más líneas de
pedido.,Spanish
```

Integration with external systems

ⓘ Note

This section applies only to SAP Ariba Invoice Management customers.

SAP Ariba Invoice Management provides an option to send reconciled IRs in **Approving** status to an external system for final approval. After the invoice is approved for payment in the ERP, the ERP sends a status update request document to SAP Business Network, which is used to update the invoice status in SAP Business Network for suppliers to track the invoice progress and also update the invoice status in SAP Ariba Invoice Management.

If your external system supports rejected reason codes, you can send the rejection reason codes as an Extrinsic in the invoice status update request document. Suppliers will see the rejection reason in the **Rejected Invoice** text box when reviewing the invoice and the History tab of the invoice:

```
<Request>
  <StatusUpdateRequest>
     <DocumentReference payloadID="xW/9u5XlhUklaIw5F49fpR9k="/>
        <Status text="" code="200"></Status>
        <InvoiceStatus type="rejected">
           <InvoiceIDInfo invoiceDate="2017-07-27T18:35:03-07:00" invoiceID="N-
NonPO-0727-6"/>
           <Comments xml:lang="en">
            Please correct the invoice and resubmit it again.</Comments>
        </InvoiceStatus>
        <Extrinsic name="RejectionReasons">
            <Extrinsic name="RejectionReason">
               <Extrinsic name="ReasonCode">A100</Extrinsic>
               <Extrinsic name="Description">Price is different from negotiated
price for one or more line items.</Extrinsic>
            </Extrinsic>
            <Extrinsic name="RejectionReason">
               <Extrinsic name="ReasonCode">C300</Extrinsic>
               <Extrinsic name="Description">Invoiced and received quantity do not
match for one or more line items.</Extrinsic>
            </Extrinsic>
         </Extrinsic>
  </StatusUpdateRequest>
</Request>
```

This functionality requires custom integration and mapping. Make sure that the reason codes and descriptions are synchronized between your external ERP system and SAP Ariba Invoice Management. If the status update request document includes a reason code that is not configured in SAP Ariba Invoice Management, the status update request process fails in SAP Ariba Invoice Management and the invoice status is not updated.

> **Note**
>
> You can find out if a status update request could not be processed in SAP Ariba Invoice Management by checking the document status in your SAP Business Network account.

Rejection Reason Codes in the User Interface and Email Notifications

When rejection reason codes for invoices is enabled, users must select one or more rejection reason codes when rejecting an invoice. The selection of rejection reasons is supported in the user interface and when rejecting from emails.

Rejection reason codes in the UI

When users reject an IR, they must select one or more applicable reason codes from the **Rejection Reasons** chooser. Subsequent users on the approval flow of an invoice rejection can override or change the rejection reasons selected by the previous approver.

In the default configuration, IRs in **Reconciling** or **Approving** status can be rejected, but your site can be configured to not allow rejecting IRs in **Approving** status.

Rejection reasons are included in the IR document, on the History tab of the IR, and are sent to the supplier on SAP Business Network, where suppliers can see them on the invoice review page and the History tab of the invoice.

> **Note**
>
> If an invoicing agent first resolves an invoice exception by adjusting the invoiced amount with the expected amount from the associated reference document such as the purchase order, contract, or receipt, that information is sent to the supplier when the invoice is approved for payment. (In contrast, information that an exception was resolved by accepting the values in the invoice is not included in the invoice update to the supplier.) This means that if your invoice validation processes allow approvers to resolve exceptions by accepting the amount specified on the associated reference document before rejecting an invoice, suppliers will see the invoice exception description of the disputed invoice exception, the rejection reason codes, and any additional comments to them.

Rejection reason codes in email rejections

When users reject an IR through email, the way to select the rejection reason codes differs depending on the email format:

- In plain text format, which includes embedded links and allows users to approve, deny, or reject requests using checkboxes, the list of rejection reason codes and descriptions is displayed in the reply message when users click the **Reject** button in the email. Users place an 'X' in the bracket ([ ]) in front of the rejection reason code to select applicable reasons for rejection.
- In compact text format, which does not include embedded links and requires a user to type customizable keywords to indicate their action, users manually enter the action ('reject') followed by a comma separated list of rejection reason codes in the next line. The list of rejection reason codes and their descriptions are included in the email message for reference.
- In HTML format, users cannot reject an IR via email. Instead, they will be redirected to the SAP Ariba application when they click the **Reject** button.

About invoice exception types and conditions

Invoice exception types describe the situations that result in exceptions. Conditions, such as tolerance calculations and invoice exception ranking, describe factors that affect when an exception is triggered and how exceptions are ranked in importance when there are multiple exceptions.

Default invoice exception types [page 14]

Applying invoice exception types to specific types of invoices [page 20]

Restricting dispute on invoice exceptions [page 21]

How to restrict users from disputing an invoice exception [page 22]

Defining invoice exception types for specific invoice source documents [page 23]

Invoice exception type overrides [page 28]

Tolerance calculations [page 29]

About validation behavior [page 32]

Skipping validation for an invoice exception type or override [page 32]

About invoice exception ranking [page 34]

About standard line categories [page 35]

Procurement line types [page 35]

Default invoice exception types

The default configuration defines typical validation rules for an extensive set of possible invoice exception types.

These exception types handle situations, such as:

- Matching exceptions, such as those for unmatched invoice or line item
- General exceptions, such as those for shipping charges or tax amounts
- Purchase order exceptions, reflecting differences between an order and an invoice
- Contracts exceptions, reflecting differences between a contract and an invoice
- Payment exceptions, such as differences in payment terms

Some exception types interact with other functionalities available in your SAP Ariba solution. These include:

- Tax Calculation Exceptions [page 14]
- Receiving Exceptions [page 15]
- Exception Types for Partial Invoicing [page 15]
- Unmatched Exceptions [page 16]
- PO Variance Exceptions [page 16]
- Exception types for goods receipt-based invoice verification [page 17]
- Invalid Requester Exception [page 19]

Tax calculation exceptions

Tax tables are used to verify the tax amount on invoice line items. When the tax amount on the invoice does not match the calculated tax amount, a tax exception occurs.

The `TaxCalculationFailed` and `WHTTaxCalculationFailed` exception types use a tolerance operation of `any`. These exceptions occur if an invoice has an indirect or withholding tax but SAP Ariba Procurement solutions are unable to calculate an applicable tax on the invoice. These exceptions indicate a possible gap in your tax tables. If your tax tables are correct, these exceptions indicate an invalid tax applied on the invoice. These exception types use only a percentage tolerance. Absolute tolerances are not appropriate for these exceptions.

If both the invoice and calculated taxes exist, the `UnderTaxVariance`, `OverTaxVariance`, `WHTUnderTaxVariance`, and `WHTOverTaxVariance` exceptions are used to compare the two tax amounts and validate the difference against the tolerances defined in your configuration. For withholding taxes, the absolute

values of the taxes are considered. For example, a withholding tax of USD -20 if considered greater than a withholding tax of USD -10.

> **Note**
>
> For `UnderTaxVariance` and `WHTUnderTaxVariance`, if you set an absolute tolerance, specify a negative number. It's incorrect to specify a positive number.

If your company uses accrual accounting, the differences in tax amounts might reflect accounting differences rather than actual exceptions. For example, if the vendor does not record tax at the point of sale, the tax amount on the invoice might differ from the calculated tax amount. In this situation, your SAP Ariba application includes the ability to accumulate accrued taxes as an expected payment, and to recognize the difference in tax amounts as an accrued tax amount.

Also, if sales tax was included for a line item on the invoice that was meant for accrual, an `AccrualTaxVariance` exception is returned.

The date based on which taxes are calculated in invoice reconciliation documents is customizable. By default, SAP Ariba Procurement solutions use the creation date of the invoice reconciliation document; but for more acurate tax calculations, SAP Ariba recommends you to use the invoice date. Contact SAP Ariba Support to change the tax calculation date.

Receiving exceptions

The default configuration defines exception types based on received quantities. The behavior of these exception types depends on how receiving is set up for your site.

The received quantity variance exception checks the `OrderLineItem.NumberAcceptedInvoiceable` field. This field is set during receiving, and updated when receipts are generated.

Related Information

Exception types for goods receipt-based invoice verification [page 17]

Exception types for partial invoicing

When comparing invoice amounts for partial invoicing, a cumulative amount invoiced accumulator is used to calculate invoice exceptions for partial invoicing.

In exceptions such as the reject amount variance, this accumulator is used to make sure that the percentage variance does not become artificially high as the amount left to be invoiced decreases. For calculating such exceptions, the cumulative amount invoiced is compared to the amount ordered. This comparison ensures that the percentage tolerance remains constant, even as the amount left to be invoiced decreases.

Unmatched invoice exceptions

Exceptions of type `UnmatchedInvoice` occur when the invoice reconciliation document cannot be matched to the order or contract.

The `FieldPathToValidate` value is ignored during the validation, but is used for formatting the exception description, for example:

```
Ariba Invoice is unable to match a purchase order to the invoice of amount {1}
```

Unmatched exceptions have a higher priority than other exception types. If an invoice reconciliation document has an unmatched exception, no other exceptions are initially generated. Header exceptions and line item exceptions are ranked separately—if an unmatched line item exception occurs, it does not prevent other header-level exceptions from being generated.

PO variance exceptions

`POAmountVariance` Exception Type

Exceptions of type `POAmountVariance` occur when the `POAmountInvoiced` field on the invoice does not match the `Order.AmountOrderedInvoiceable` field on the order. The `POAmountInvoiced` field includes regular line invoiced amounts as well as charge amounts, if any, such as taxes, shipping, and handling.

If your site supports taxes, charges, and discounts on purchase orders, the `Order.AmountOrderedInvoiceable` field includes any taxes, charges, and discounts applied on the purchase order. In this case, the default configuration for the `POAmountVariance` exception is correct.

If your site does not support taxes, charges, and discounts on purchase orders, you can configure the `POAmountVariance` exception type to compare the `Order.AmountOrderedInvoiceable` field on the purchase order with the derived field `POAmountInvoicedLessCharges` on the invoice. The `POAmountInvoicedLessCharges` field is calculated by excluding the charge amounts from the `POAmountInvoiced` field.

For example:

```
Cp1252
"UniqueName","HeaderOnly","InvoiceCategory","LineCategory","LineType","FieldPathToVa
lidate",
"FieldPathTarget","FieldPathToValidateAgainst","FieldPathSourceOnDispute","NullAllow
ed",
"ToleranceOperation","AbsoluteTolerance","PercentageTolerance","AbsoluteRank","Relat
iveRank",
"AutoReject","QuantifiableOnly","SpendCategoryDriver","SpendCategory","IgnoreAutoAcc
ept"
"POAmountVariance",true,3,0,"","POAmountInvoicedLessCharges","TotalCost",
"Order.AmountOrderedInvoiceable","POAmountOrderedLessPreviouslyInvoiced",false,"or",
10,1.20,10,2,false,false,"","",false
```

`POAmountInvoicedLessCharges` is the amount that is already invoiced on the matched purchase order, including the current invoice amount, but excluding any tax and shipping amounts.

Exceptions of type `POPriceVariance` are filtered by line category, and apply to standard line items only, not to items assigned to line-type categories. When your SAP Ariba application extracts line items to validate, it filters the

set of line items and checks only those that are of line category 1 (exception line items). This exception specifies that a line item has an exception when the `Description.Price` field does not match the `ExpectedPrice` field within the defined tolerances.

`POCatalogPriceVariance` Exception Type

Exceptions of type `POCatalogPriceVariance` are filtered both by line category and by line type. Your SAP Ariba application validates only line items that are procure line items and then filters again to select only line items that are of line type `_CatalogItem`.

Purchase Order Tax and Charge Amount Variance Exception Types

If your site supports taxes, charges, and discounts on purchase orders, you can activate the `POTaxAmountVariance`, `POTaxLineAmountVariance`, `POChargeAmountVariance`, and `POChargeLineAmountVariance` invoice exception types to generate invoice exceptions when the tax or charge amounts on the purchase order do not match the tax and charge amounts on the invoice. To resolve these exceptions, you can only accept the invoice tax or charge amounts.

- `POTaxAmountVariance`: This exception is displayed if the total tax amount on the invoice and the total tax amount on the purchase order vary by a value greater than the tolerance set.
- `POTaxLineAmountVariance`: This exception is displayed if the line-level tax amount on the invoice and the line-level tax amount on the purchase order vary by a value greater than the tolerance set.
- `POChargeAmountVariance`: This exception is displayed if the total charge amount on the invoice and the total charge amount on the purchase order vary by a value greater than the tolerance set.
- `POChargeLineAmountVariance`: This exception is displayed if the line-level charge amount on the invoice and the line-level charge amount on the purchase order vary by a value greater than the tolerance set.

Exception types for goods receipt-based invoice verification

For SAP Ariba Invoice Management sites that support goods receipt-based invoicing, there are invoice exceptions for discrepancies between the amount, quantity, and unit price between the invoice and receipt. There's also an exception if no matching receipt is found.

If your site supports goods receipt-based invoice verification, we recommend activating these exception types.

When invoices are validated, if these exceptions arise, SAP Ariba Invoice Management adds the **Receiving Manager** group to the approval flow of the invoice reconciliation to resolve them.

Receipt Amount Variance

The amount for a line item on the invoice doesn't match the amount for the item on the associated receipt, and the difference is greater than the tolerance defined for the exception. For example:

- **Receipt Amount Variance** tolerance set to 4%.
- 10 keyboards were ordered, invoiced, and received.
- Amount on invoice: 1,050 USD.
- Amount on receipt: 1,000 USD.

Receipt Amount Variance exception is triggered because the amount difference of 50 is greater than 4%.

Receipt Quantity Variance

The quantity for a line item on the invoice doesn't match the quantity for the item on receipts that reference the same ship notice, and the difference is greater than the tolerance defined for the exception.

This exception is also triggered if the quantity for the invoice item's matching receipt item has already been fully allocated. For example:

- **Receipt Quantity Variance** tolerance set to 10%.
- 10 keyboards were ordered and invoiced.
- 8 keyboards were received.

Receipt Quantity Variance exception is triggered because quantity difference of 2 is greater than 10%.

Receipt Unit Price Variance

The unit price of a line item on an invoice doesn't match the unit price of the item on the receipt, and the difference is greater than the tolerance defined for the exception. For example:

- **Receipt Unit Price Variance** tolerance set to 10%.
- 10 keyboards were ordered, invoiced, and received.
- Price on invoice: 120 USD.
- Price on receipt: 100 USD.

Receipt Unit Price Variance exception is triggered because price difference of 20 is greater than 10%.

Matching Receipt Not Found

The invoicing solution could not find a receipt referencing the ship notice associated with a line item on the invoice. For example:

- 10 keyboards were ordered and invoiced.
  The item was marked as **Ship Notice Required** on the order, but there's no receipt for that ship notice.

Matching Receipt Not Found exception is triggered.

> **Note**
>
> An electronic invoice typically arrives in SAP Ariba Invoice Management before goods are received. You can set the parameter **Days to delay creating an IR document** (`Application.Invoicing.DaysToDelayIRCreation`) to delay the start of reconciliation for a specific amount of time to avoid triggering the **Matching Receipt Not Found** exception all the time.

Invalid requester exceptions

Use the invalid requester invoice exception to autoreject non-PO invoices submitted from SAP Business Network.

> **Note**
>
> Autorejection does not apply to non-PO invoices submitted through an invoice conversion service. For information about working with an invoice conversion service, see the SAP Business Network Guide to Invoice Conversion.

In addition to allowing to autoreject non-PO invoices with invalid email addresses, the Requester Defaulting feature defaults the requester (on behalf of) on an invoice to the requester of the order for manually entered PO-based invoices or PO-based invoices submitted from SAP Business Network and validates the requester on all invoices.

Use the Invalid Requester invoice exception to configure autorejection for non-PO supplier entered invoices that have invalid Sold To addresses. If you want to disable autorejection, you must deactivate the invoice exception, which allows members of the **Invoice Agent** group to correct any invalid email addresses.

About Invalid Requester invoice exception is activated

Non-PO invoices that have been entered manually by the supplier or submitted through EDI or cXML (`SubmissionMethod = Online or EDI or cXML` and `InvoiceOrigin = Supplier`) are auto-rejected.

This also applies to non-PO invoices that reference invalid purchase order IDs or invalid contract IDs (Customer Order # or Contract # field in the Create Invoice supplier user interface in SAP Business Network).

Invoices are rejected with the following messages:

`The email address (Sold To) <email address> on the invoice is invalid.`

All other types of non-PO invoices with invalid or missing requester (Sold To) addresses including non-PO invoices that have been submitted through an invoice conversion service (ICS invoices identified by `SubmissionMethod = PaperViaICS`) generate a validation error and the **Invoice Agent** group is added to the approval flow to choose a valid requester and process the invoice reconciliation. The requester is set to 'aribasystem'.

About Invalid Requester invoice exception is deactivated

Regardless of the submission method or origin, the invoice is forwarded to an exception handler to resolve the invalid requester validation error. The requester is set to 'aribasystem'.

> **Note**
>
> The Invalid Requester invoice exception is only checked in sites where the requester defaulting feature is enabled.

Applying invoice exception types to specific types of invoices

You can choose which types of invoices an invoice exception type applies to.

Each invoice exception type includes a **Apply to Specific Invoice Types** field that lets you limit the types of invoices the exception type applies to. If you limit an exception type to specific types of invoices, an exception is raised only for the invoice types you specify, increasing flexibility for the invoice reconciliation processes.

Setting **Apply to Specific Invoice Types** to **Yes** displays a list of choices. You can choose one or more of the following:

- **Standard** (for standard invoices)
- **Credit Memo**
- **Line Item Credit Memo**
- **Line Item Debit Memo**

The **Import Invoice Exception Type Validation Data** and **Export Invoice Exception Type Validation Data** tasks include an `InvoicePurposes` column, which holds a comma-separated list of the types of invoice documents are applicable for the invoice exception type. This column corresponds to the **Apply to Specific Invoice Types** field in the user interface. An empty value in the `InvoicePurposes` column means the invoice exception type is not restricted; it's applied to all invoice types.

> **Note**
>
> Choosing invoice types for an invoice exception type doesn't override site parameter settings. For example, if the parameter `Application.Invoicing.SkipLineCreditMemoExceptions`, which is set by SAP Ariba Support, is set to `Yes`, exceptions aren't raised against line-item credit memos, regardless of the **Apply to Specific Invoice Types** field.

Related Information

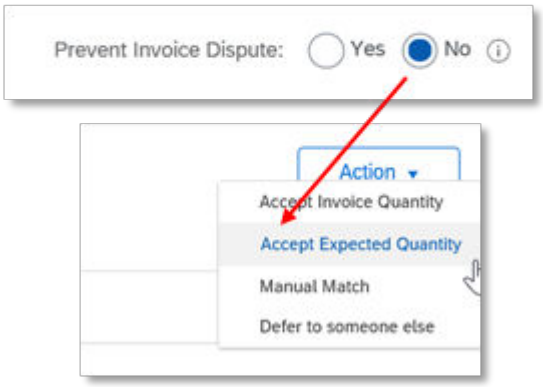
Defining invoice exception types for specific invoice source documents [page 23]
Invoice exception type overrides [page 28]
Skipping validation for an invoice exception type or override [page 32]
Tolerance calculations [page 29]

Restricting dispute on invoice exceptions

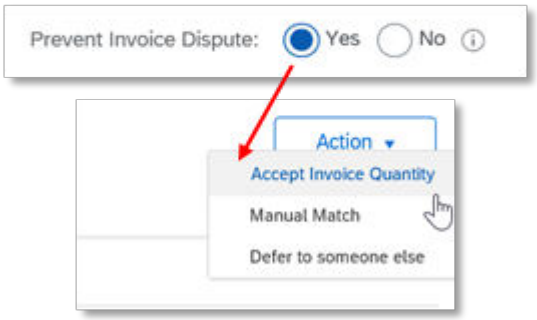
You can restrict users from disputing certain invoice exceptions, preventing them from disputing some exceptions while allowing them to dispute other exceptions.

You control which exceptions users can and cannot dispute on the edit page of the exception type. To prevent users from disputing an exception, set the **Prevent Invoice Dispute** field to **Yes** for the invoice exception type. To allow users to dispute an exception, set the **Prevent Invoice Dispute** field to **No**.

When the **Prevent Invoice Dispute** field is set to **No**, the Accept Expected option is available from the Action dropdown.



When the **Prevent Invoice Dispute** field is set to **Yes**, the Accept Expected option is not available from the Action dropdown.



> **Note**
>
> Your site might be using a custom label other than **Accept Expected** for the dispute action.
>
> When the parameter **Prevent dispute of invoices**(`Application.Invoicing.DisallowDispute`) is set to **No**, you can disable the **Accept Expected** option using the **Prevent Invoice Dispute** field on the invoice exception type configuration page.

Related Information

How to restrict users from disputing an invoice exception [page 22]

How to restrict users from disputing an invoice exception

You can prevent users from disputing individual exceptions on an invoice. For example, when legally mandated taxes are applied to an invoice, you can restrict users from disputing exceptions related to those taxes.

Prerequisites

Members of the **Customer Administrator** or **Invoice Administrator** group can perform this task.

The parameter **Prevent dispute of invoices** (`Invoicing.Application.DisallowDispute`) must be set to **No** in order for the **Prevent Invoice Dispute** option to take effect.

The `SINV-6861: Restrict dispute per invoice exception type` feature must be enabled.

Context

You can choose which invoice exceptions they want to restrict users from disputing. By restricting which exceptions users can dispute, users must accept what the supplier is offering. The **Prevent Invoice Dispute** field on the **Invoice Exception Types** page controls whether or not the user can dispute the exception. When the **Prevent Invoice Dispute** field is set to **Yes**, users cannot dispute the exception. When the **Prevent Invoice Dispute** field is set to **No**, users can dispute the exception.

If your site has the parameter **Prevent dispute of invoices** set to **Yes**, but you want to allow users to dispute certain exceptions and prevent them from disputing all exceptions, you can configure the exception types and then turn the parameter off. Set the **Prevent Invoice Dispute** field to **Yes** for the exception type you want to allow users to dispute, and then set the parameter to **No**. This allows users to dispute certain exceptions and prevents undesired disputes from occurring.

When the parameter **Prevent dispute of invoices** is set to **Yes**, the **Prevent Invoice Dispute** field can be changed to **Yes** or **No**, but the **Accept Expected** option is never available. When the parameter **Prevent dispute of invoices** is set to **No**, the **Prevent Invoice Dispute** field can be changed and the **Accept Expected** option is available.

> **Note**
>
> Your site might be using a custom label other than **Accept Expected** for the dispute action.

With the following exception types, the **Prevent Invoice Dispute** field is set to **Yes** and cannot be edited:

Canceled Order	**Contract Amount Variance**	**Contract Hard Maximum Limit Exceeded**
Contract Line Amount Variance	**Contract Not Invoiceable**	**Contract Not Invoicing**
Invalid Invoice Date	**Invalid Requester**	**Invoice currency mismatch**
Invoice has mathematical errors	**Invoice has mixed amounts**	**Invoice has mixed currency**
Invoice Line matched to a Pcard Order	**Matching Receipt Not Found**	**PO Amount Variance**
PO Charge Amount Variance	**PO Closed for Invoicing**	**PO Line Charge Amount Variance**

PO Line Closed For Invoicing	**PO Line Rejected for Invoicing**	**PO Line Tax Amount Variance**
PO Max Amount Variance	**PO Rejected for Invoicing**	**PO Service Date Variance**
PO Tax Amount Variance	**Tax Calculation Failed**	**Withholding Tax Calculation Failed**

You can also prevent users from disputing exceptions based on the invoice. The **Prevent Invoice Dispute** field can be overridden for any one of the following invoices:

- Invoices that use a specific supplier
- Invoices with a specific commodity code for line items
- Invoices that belong to a specific procurement unit
- Invoices that are based on a specific currency

Procedure

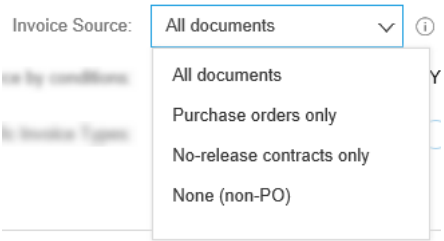
1. From the dashboard, choose **Manage** > **Core Administration**.
2. Navigate to the invoice exception type by performing one of the following actions:
   - In SAP Ariba Buying and Invoicing, choose **Procure-to-Pay Manager** > **Invoice Exception Types**.
   - In SAP Ariba Invoice Management, choose **Invoicing Manager** > **Invoice Exception Types**.
3. Search for the invoice exception type you want to apply the restriction to.
4. From the **Actions** dropdown, select **Edit**.
5. In the **Prevent Invoice Dispute** field, choose one of the following options:
   - Select **Yes** to hide the **Accept Expected** option on the **Invoice Exception Types** page.
   - Select **No** to allow users to select the **Accept Expected** option on the **Invoice Exception Types** page.
6. Click **Save**.

Defining invoice exception types for specific invoice source documents

You can define a custom invoice exception type so it's applied only to invoices that have a particular type of order or other source document.

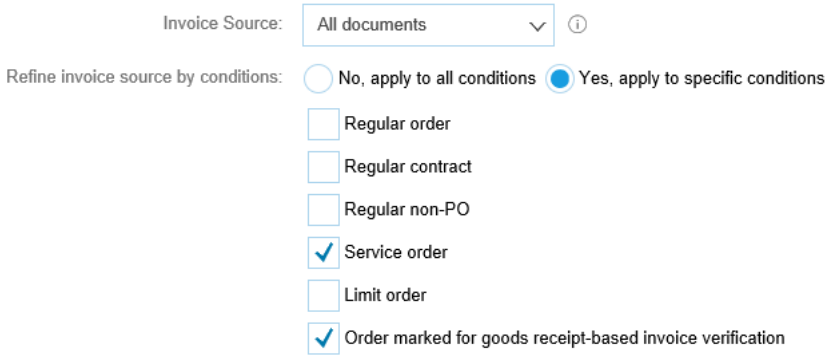
For example, some customers want to raise certain invoice exceptions for service order invoices and other invoice exceptions for invoices for material goods orders. Limiting the situations in which exceptions are raised can ease the invoice reconciliation process.

In Ariba Administrator, each invoice exception type has an **Invoice Source** field (formerly called **Invoice Category**) for specifying which invoices the exception type applies to based on the source document for the invoice. Exceptions are raised only for invoices that are based on the specified source.

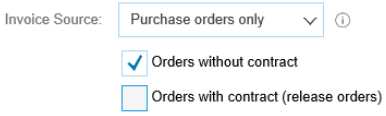


If you choose **All documents** or **Purchase orders only** as the invoice source, the field **Refine invoice source by conditions** is available, with the following options:

- **No, apply to all conditions**: there is no additional refinement of the selected invoice source. This is the default setting.
- **Yes, apply to specific conditions**: use the checkboxes to specify conditions that refine the invoice source. For example, you can refine the source to service orders only, or to both service orders and orders marked for goods receipt-based invoice verification.
  The following figure shows the checkboxes available when the invoice source is **All documents**. In this example, the invoice exception type is applicable to invoices based on either service orders or orders marked for goods receipt-based invoice verification.



The following table describes invoice source documents and the conditions for further refining them.

Invoice source	Description	Predefined conditions for refining the source
All documents	The exception type applies to invoices regardless of the type of source document.	• **Regular order** (not a limit order, an order marked for goods receipt-based invoice verification, or a service order with a service sheet) • **Regular contract** • **Regular non-PO** (for example, orders created in an external ERP system) • **Service order** (with a service sheet) > **Note** > Service orders without service sheets are treated as regular orders. • **Limit order** • **Order marked for goods receipt-based invoice verification**
Purchase orders only	The exception type applies to invoices based on purchase orders created in the SAP Ariba solution. Checkboxes let you narrow the orders down to orders with or without a contract: 	• **Regular order** (not a service order, limit order, or order marked for goods receipt-based invoice verification) • **Service order** (with a service sheet) • **Limit order** • **Order marked for goods receipt-based invoice verification**
No-release contracts only	The exception type applies to invoices based on no-release contracts.	No conditions exist for contract source documents. This option is the same as choosing **All documents** and refining by the **Regular contract** condition.
None (non-PO)	The exception type applies to invoices based on purchase orders or other documents created outside the SAP Ariba solution (non-PO invoices).	No conditions exist for non-SAP Ariba source documents. This option is the same as choosing **All documents** and refining by the **Regular non-PO** condition.

Invoice source fields in data import and export tasks

The **Import Invoice Exception Type Validation Data** and **Export Invoice Exception Type Validation Data** tasks include columns for indicating the applicable invoice source documents.

Table 1: Invoice source document fields in **Import Invoice Exception Type Validation Data** and **Export Invoice Exception Type Validation Data**

Import/export field	Value	Field name in the user interface
`InvoiceCategory`	Integers represent invoice source document types. Here are the possible integer values with their corresponding settings in the user interface. • 0: **All documents** • 1: **Purchase orders only**, with the **Orders without contract** checkbox selected • 2: **Purchase orders only**, with the **Orders with contract (release orders)** checkbox selected • 3: **Purchase orders only**, with the **Orders without contract** and **Orders with contract (release orders)** checkboxes selected • 4: **No-release contracts only** • 8: **None (non-PO)**	**Invoice Source**

Import/export field	Value	Field name in the user interface
`SpecialConditions`	An expression that defines the conditions for refining the specified source document. • `RegularOrder` • `RegularContract` • `RegularNonAriba` (corresponds to **None (non-PO)** in the user interface) • `ServiceOrder` • `LimitOrder` • `GRBased` For example: `RegularNonAriba` or `GRBased` means the invoice exception type applies to invoices that have no source document in SAP Ariba **or** are for orders marked for goods receipt-based invoice verification. To remove all conditions, enter `null`. This is the same as choosing **No, apply to all conditions** for the exception type in Ariba Administrator. An empty value leaves the current refinement conditions in place. In other words, it makes no changes to the `SpecialConditions` field.	The refinement options under **Invoice Source**.

Example

The following 3-column excerpt of the `InvoiceExceptionTypeValidationExport.csv` file shows an example of `InvoiceCategory` and `SpecialConditions` values.

Sample Code

```
UniqueName,InvoiceCategory,SpecialConditions
MyCustomExceptionType,3,RegularOrder or ServiceOrder or LimitOrder
```

In the example, the invoice exception type `MyCustomExceptionType`:

- Applies to purchase orders with or without a contract: `InvoiceCategory` = 3
- Applies to invoices based on regular orders, service orders, or limit orders
- Doesn't apply to invoices based on goods receipts

Related Information

Applying invoice exception types to specific types of invoices [page 20]
Invoice exception type overrides [page 28]
Skipping validation for an invoice exception type or override [page 32]
Tolerance calculations [page 29]

Invoice exception type overrides

The Invoice Exception Type Override feature allows you to define tolerances at the supplier level and the commodity code level for each invoice exception type.

This allows for more efficient invoice reconciliation and exception handling, where, for example, a greater tolerance is more suitable for some suppliers, while for others, a more narrow tolerance is desirable.

Configuring invoice exception type overrides is a two step process: creating the override, and applying it.

1. Create available exception type overrides by suppliers, commodity code, purchasing unit, or currency.
   You can create these overrides on the **Invoice Exception Types** page for any invoice exception type.
   Each exception type has an **Overrides** tab. Regardless of which invoice exception type you're editing when you create an override, the override will be available for use with any invoice exception type.
2. Apply the override to the invoice exception type, and specify the absolute and percentage tolerances, a rank for the applied override, and whether to automatically reject invoices that trigger exceptions.

> **Note**
>
> You cannot create invoice exception type overrides for boolean (yes/no) or field type (Supplier, Commodity) checks on **PO Payment Terms mismatched** exceptions.

Invoice exception overrides by currency are applied in one of two ways, depending on site configuration:

- In sites that apply exception type absolute tolerance values in the site's default currency (the default behavior), if the site's default currency is USD, an absolute tolerance value of $10 is applied to all invoices, and is converted into the invoice currency for invoices that use non-USD currencies. An invoice manager can create an exception override for JPY of 10; that amount is converted from dollars to yen and then applied to invoices in JPY. In this case, the absolute tolerance amount for the currency override fluctuates with the exchange rate, and if there is a significant change to the exchange rate between when the purchase order is created and invoiced, the invoice reconciliation might generate unintended variance exceptions that must be manually reconciled. It also might not reflect purchasing power or other local conditions for the specific currency.
- In sites that are configured to apply absolute tolerance amounts in the invoice currency rather than the site default currency (the recommended configuration), invoice managers can set an exception type absolute tolerance of 10, which is applied as $10 on a USD invoice and ¥10 on a JPY invoice. The invoice manager can then set an exception type override with an absolute tolerance of 110 for JPY. In this case, the absolute tolerance amount for the currency override is fixed and is applied consistently as ¥110. It can be adjusted up or down to reflect purchasing power or other local conditions for the specific currency, rather than being tied to the site's default currency and the current exchange rate.

Related Information

About exception type override ranking [page 35]
Applying an override to an invoice exception type [page 42]
Skipping validation for an invoice exception type or override [page 32]
How to configure an invoice exception type to be skipped [page 43]

Tolerance calculations

Tolerance calculations are based on operations that define how the absolute and percentage tolerances are evaluated to trigger exceptions.

The absolute tolerance setting is evaluated first. The absolute tolerance values can be either based on the invoice amount or quantity only (invoice-based absolute tolerance) or based on the difference between the invoice and the PO or contract amount or quantity (difference-based absolute tolerance).

> **Note**
>
> For the `UnderTaxVariance` and `WHTUnderTaxVariance` exception types, if you set an absolute tolerance, specify a negative number. It's incorrect to specify a positive number.

Absolute Tolerance Threshold	Tolerance Operator.	Result
Exceeded	And	Exception is triggered
Exceeded	Or	Evaluate percentage tolerance: • Percentage tolerance met --> No exception • Percentage tolerance exceeded --> Exception is triggered
Met (below threshold)	And	Evaluate percentage tolerance: • Percentage tolerance met --> No exception • Percentage tolerance exceeded --> Exception is triggered
Met (below threshold)	Or	No exception

Absolute Tolerance Threshold is Exceeded

Absolute tolerance threshold exceeded means the absolute tolerance threshold is not met.

If the tolerance operation is **And**, the percentage tolerance does not have to be evaluated, and an exception is triggered by having exceeded the absolute tolerance value alone. If the tolerance operation is **Or**, the percentage tolerance is evaluated to determine if the values on the invoice are within the percentage tolerance calculation (no exception) or if the values on the invoice exceed the percentage tolerances (exception is triggered).

Absolute Tolerance Threshold is Met

Absolute tolerance threshold met means that the values on the invoice (for invoice-based absolute tolerance configuration) or the difference between the values on the invoice and PO or contract (for difference-based absolute tolerance configuration) are below the threshold settings.

If the tolerance operation is **Or**, no further calculations are necessary and the invoice line or invoice is accepted. If the tolerance operation is **And**, the percentage tolerance is evaluated. If the percentage tolerance is met, no invoice exception is triggered. If the percentage tolerance is not met, an invoice exception is triggered.

If the absolute tolerance value is 0 (zero), the tolerance operation is always evaluated, and the percentage tolerance setting determines exception handling.

If the percentage tolerance is 0, invoice amount or quantities are evaluated using the absolute tolerance values alone.

Tolerance threshold examples

Tolerance threshold examples are based on this scenario: an absolute tolerance value of 50 is set on the PO Line Amount Variance exception, and the tolerance is applied to the difference between the invoice and the PO. The percentage tolerance value is 1.03 (3%).

Tolerance threshold example 1:

The PO line item amount is 1,000.00 and the invoice line item amount is 1,045.00. The tolerance operation is **Or**.

Result: The absolute tolerance (the difference between the PO line amount and invoice amount, 45, is less than the tolerance threshold of 50) is met, and the tolerance operation is evaluated. The invoice line is accepted because the tolerance operation is **Or**.

Tolerance threshold example 2:

The PO line item amount is 1,000.00 and the invoice line item amount is 1,045.00. The tolerance operation is **And**.

Result: The absolute tolerance (the difference between the PO line amount and invoice amount, 45, is less than the tolerance threshold of 50) is met, and the tolerance operation is evaluated. The percentage tolerance is evaluated because the tolerance operation is **And**. The percentage tolerance threshold (3% of PO line amount of 1,000.00 = 30) is not met. An invoice exception is triggered.

Tolerance threshold example 3:

The PO line item amount is 1,000.00 and the invoice line item amount is 1,055.00. The tolerance operation is **Or**.

Result: In this case, regardless of the tolerance operation, an invoice exception is triggered because both, the absolute (50) and percentage (3% of PO line amount of 1,000.00 = 30) tolerance thresholds are exceeded.

Tolerance threshold example 4:

The PO line amount is 5,000.00, invoice line amount is 5,065.00. Tolerance operation is **Or**.

Result: The absolute tolerance difference threshold (50) is exceeded, and the tolerance operation is evaluated. The percentage tolerance (3% of 5,000.00 = 150.00) is met (i.e. the difference between the line item invoice amount and PO line item amount is within the threshold set by the percentage tolerance setting). The invoice line is accepted.

Tolerance threshold example 5:

The PO line amount is 5,000.00, invoice line amount is 5,065.00. Tolerance operation is **And**.

Result: The absolute tolerance difference threshold (50) is exceeded, and the tolerance operation is evaluated. Since the tolerance operation is **And**, and the absolute tolerance is not met, an invoice exception is triggered.

About absolute variance tolerances with contract limits

When defining a contract request, contract managers can set a percent tolerance value on the contract's maximum limit. That tolerance is checked during exception calculation to decide if an invoice triggers an exception or not.

Suppose a contract of USD 10,000.00 has a percentage tolerance of 2%. If the invoice line amount is for USD 10,150.00, the invoice is accepted because the amount is within the allowable tolerance limit.

If the contract line item variance exception is configured for an absolute variance tolerance of 100, and the contract tolerance limit is not a hard limit, then only invoice line item amount values greater than 10,300 will trigger an exception. In effect, the allowable tolerance values of the contract and the invoice exception are combined to determine the maximum allowable amount that does not trigger an exception.

If the contract limit is defined as a hard limit, the variance tolerance defined for the exception is disregarded. In this example, any invoice amount over USD 10,200.00 will lead to an automatic rejection of the invoice.

About validation behavior

The behavior of validation depends on the values in **Field Path to Validate**, **Field Path to Validate Against**, and **Tolerance Operation** (columns `FieldPathToValidate`, `FieldPathToValidateAgainst`, and `ToleranceOperation` in `InvoiceExceptionTypeValidation.csv`).

The process is as follows:

- If you specify Field Path To Validate, but not Field PathTo Validate Against or Tolerance Operation, the validity check tests only the specified Field Path to Validate to see if it is null.
- If you specify Field Path To Validate and Field Path To Validate Against, but not ToleranceOperation, the validity check tests to see if the two fields are equal.
- If you specify Tolerance Operation of **or** or **and**, the validity check tests the absolute tolerance first and then the percentage tolerance (if necessary).
- If you specify Tolerance Operation of **any**, the validity check tests for the existence of a field.
- If you specify Tolerance Operation of **boolean**, the validity check tests for the Boolean value of the field Field Path To Validate Against, which can be set to any Boolean field. If the boolean is set to True, no exception will be generated.
- If you specify Tolerance Operation of **boolean**, the validity check skips NullAllowed.

Skipping validation for an invoice exception type or override

During invoice reconciliation, it's possible to skip validation of an exception type in some cases while applying it in other cases.

For example, you might want to skip currency mismatch validation for one or two purchasing units but apply it to all other purchasing units. Or you might want to skip an exception type for all suppliers except one or two.

Skipping exception types in specific circumstances can reduce the number of unnecessary exceptions, speeding up the reconciliation process.

Workflow

The following high-level steps describe the workflow when you configure the invoice reconciliation process to skip validation for an invoice exception type.

1. A user with administrative permissions configures an invoice exception type to be skipped either by default or with an override.
2. The invoice reconciliation process checks the **Skip Exception** settings for the active invoice exception types, checks the information in the invoice (if there are overrides for skipping exception types), and determines whether to validate for an exception type or skip it.

> **Note**
>
> If more than one exception type override is applicable to the invoice, only the highest-ranking applied override is used. Keep this in mind if the override for skipping (or applying) the exception type has a lower rank.

3. In cases where the invoice reconciliation process skips validation for an exception type, no exceptions are raised. In cases where validation is **not** skipped, exceptions are raised if applicable.

Example

Suppose you have purchasing units set up based on countries/regions, and you want to skip currency mismatch validation for Russia but apply it to all other countries/regions.

1. In Ariba Administrator, open the **Invoice currency mismatch** exception type.
2. Because you want the reconciliation process to validate for this exception type for almost all purchasing units, make sure **Skip Exception** is set to **No** on the **Default** tab of the exception type.
3. On the **Overrides** tab for the exception type, create an override based on purchasing unit, and specify the Russia purchasing unit.
4. Add the override to **Applied Overrides**, and check **Skip Exception**. Checking **Skip Exception** tells the reconciliation process not to raise currency mismatch exceptions for invoices associated with the Russia purchasing unit.

> **Note**
>
> You might have to use the **Table Options** icon to display the **Skip Exception** column in the **Applied Overrides** table.

Behavior of Skip Exception combinations for exception types and overrides

The following table describes the effects of the possible combinations of **Skip Exception** settings for a particular invoice exception type.

- A setting of **No** means the reconciliation process can raise exceptions based on the exception type. Validation is not skipped.
- A setting of **Yes** means the reconciliation process skips validation for the exception type and doesn't raise exceptions.

Skip Exception setting on the Default tab	Skip Exception setting in the applied override	Result
No	**No**	The reconciliation process always validates for the exception type.

Skip Exception setting on the Default tab	Skip Exception setting in the applied override	Result
No	Yes	The reconciliation process validates for the exception type by default and skips validation for invoices that satisfy the conditions set in the override.
Yes	No	The reconciliation process skips validation for the exception type by default but validates for invoices that satisfy the conditions set in the override.
Yes	Yes	The reconciliation process always skips validation for the exception type. This is similar to what happens when you deactivate an exception type. The difference is that, as long as an exception type is active, a higher-ranked applied override with a different **Skip Exception** setting can affect the reconciliation process.

Related Information

How to configure an invoice exception type to be skipped [page 43]
Applying invoice exception types to specific types of invoices [page 20]

About invoice exception ranking

The reconciliation engine uses the **Absolute Rank** value to determine the order for checking exceptions.

The process is as follows:

1. The reconciliation engine generates exceptions for the header.
   A higher-ranked header exception prevents any lower-ranked header exceptions from being generated.
2. The reconciliation engine executes the same logic for lines. It uses any previously generated header exceptions in the comparison. A higher-ranked header exception can supersede all line-level exceptions.
3. If an invoice has two or more exceptions with the same absolute rank, the results depend on whether the invoice exception types have the same target field (**Field Path Target**):
   - If the target field is the same, the **Relative Rank** field determines the relative precedence of the exceptions. The exception with the lowest precedence is ignored.
   - If the target field is different, no exceptions are ignored.

The reconciliation engine does not re-check lower-ranked exceptions when higher-ranked exceptions are reconciled. The ranking is used only as a filtering mechanism during the exception generation phase.

About exception type override ranking

If an invoice exception has multiple overrides defined for them, and both overrides apply to the invoice line, it uses the override rank to determine which override takes precedence.

For example, suppose the invoice exception PO Quantity Variance has both a commodity code override and a supplier override applied to it:

Exception Type	Override Type	Field Value	% Tolerance	Override Rank
PO Qty Variance	Supplier	JCN	10%	10
PO Qty Variance	Commodity Code	Electronic	8%	20

In this example, because the rank of the Supplier override is higher than the commodity code override, the PO Quantity Variance for the Supplier takes precedence.

About standard line categories

The **Line Category** field (`LineCategory` column in `InvoiceExceptionTypeValidation.csv`) specifies a line category of the line item. Line categories separate procurement line items from the various additional categories, such as tax, shipping, and discount. This field is used to filter line items to validate.

The following table describes the default line categories.

Category Number	Category Name	Description
1	Line Item	For all line items, including non-catalog, customer catalog, and PunchOut
2	Tax	For all tax lines
4	Shipping Charge	For all charge lines with the out-of-the-box shipping charge type
8	Special Handling Charge	For all charge lines with the out-of-the-box special handling charge type
16	Charge	For all charge lines with customer-defined charge types
32	Discount	For all discount lines. Discount lines have a negative amount.
64	Withholding Tax	For withholding tax lines. These lines have a negative amount.

Procurement line types

The **Line Type** value (`LineType` column in `InvoiceExceptionTypeValidation.csv`) specifies the unique internal identifier of a procurement line type. This field is used to restrict validation to line items of a certain type.

The following table describes the internal identifiers and user-visible descriptions of the default procurement line types.

Internal ID	User-Visible Description
`_CatalogItem`	Customer catalog Item
`_NonCatalogItem`	Non-Catalog Item
`_PunchOutItem`	PunchOut Item
`_ContractExpenseItem`	Contract Expenses Item
`Discount`	Discount
`_FeeItem`	Fee (Fixed/Recurring) Item
`GSTTax`	Goods and Services Tax
`HSTTax`	Harmonized Sales Tax
`_MiscExpenseItem`	Miscellaneous Expenses Item
`PSTTax`	Provincial Sales Tax
`QSTTax`	Quebec Sales Tax
`SUTTax`	(PeopleSoft only) Sales/Use Tax
`SalesTax`	(SAP and Generic only) Sales Tax
`UseTax`	(SAP and Generic only) Use Tax
`_SLAItem`	Service Level Agreement Item
`FreightCharge`	Shipping
`HandlingCharge`	Special Handling
`VATTax`	Value Added Tax
`WHTTax`	Withholding Tax

> **Note**
>
> The leading underscore character (_) in an internal identifier for a procurement line type indicates that the procurement line type is required.

Maintaining invoice exception types in Ariba Administrator

Members of the **Invoice Administrator**, **Payment Administrator**, **Invoice Manager**, **Payment Manager**, **Procurement Manager**, and **Customer Administrator** groups can manage invoice exception types using the **Invoice Exception Types** task in the **Invoicing Manager** workspace (SAP Ariba Invoice Management) or the **Procure-to-Pay Manager** workspace (SAP Ariba Buying and Invoicing) in Ariba Administrator.

If you need to add or modify multiple invoice exception types, SAP Ariba recommends you define your data in CSV files and run the **Import Invoice Exception Types** and **Import Invoice Exception Type Validation Data** data import tasks.

Adding or modifying invoice exception types [page 37]
About configuring invoice exception type overrides [page 41]
Applying an override to an invoice exception type [page 42]
How to configure an invoice exception type to be skipped [page 43]
How to view or delete invoice exception types [page 44]

Related Information

About Data Import Tasks for Invoice Exception Types [page 45]

Adding or modifying invoice exception types

You can create a new invoice exception type or modify an existing invoice exception type.

Prerequisites

You must be a member of the **Invoice Administrator**, **Payment Administrator**, **Invoice Manager**, **Payment Manager**, **Procurement Manager**, or **Customer Administrator** group to add or modify exception types.

Procedure

1. Choose **Ariba Procure-to-Pay Manager** > **Invoice Exception Types** (SAP Ariba Buying and Invoicing) or **Payment Manager** > **Invoice Exception Types** (SAP Ariba Invoice Management) and click **Create New** to create a new invoice exception type, or select the invoice exception type you want to modify and click **Edit**.
2. Enter or modify information on the **Create Invoice Exception Types** or **Edit Invoice Exception Type** page. The following table describes the fields on these pages.

> **Note**
>
> Certain fields, such as **Field Path To Validate**, specify field names. You will need to work with SAP Ariba Support when specifying field names.

Field	Description
Partition	This information is for SAP Ariba internal use.

Field	Description
Adapter Source	The adapter source, which can be one of the following values: • **AribaManaged**—The object is not maintained in an external system. Objects with this data source are maintained in Ariba Administrator. • **External**—The object and all its data fields are maintained in an external system, such as an ERP system or a set of CSV files. Objects with this adapter source can be overwritten or deleted when data is imported from the external system. • **ManuallyMaintained**—The source of the object is an external system, but the object cannot be overwritten or deleted when data is imported from the external system. You cannot modify this field in edit mode.
Unique Name	The unique internal identifier for the invoice exception type. You cannot modify this field in edit mode.
Name	The display name for the invoice exception type.
Description	The user-visible description for the invoice exception type. The string can take parameters, indicated as `{0}` and `{1}`, where: • `{0}` is the field designated in the **Field Path To Validate Against** field. • `{1}` is the field designated in the **Field Path To Validate** field.
Group	The **Group** field specifies the unique identifier of a group. Members of this group are authorized to reconcile exceptions of the invoice exception type. Depending on the group assigned to reconcile an invoice exception and the users' group memberships, users may be required to correct validation errors before they can submit the IR. For example, users of the Invoice Agent group have full editing access and are therefore required to correct all validation errors. Users of the Invoice Exception Processor group have no editing access to the IR and can only reconcile exceptions (and validation errors will need to be corrected by subsequent approvers or other exception handlers with editing capabilities). Create custom invoice exception groups, for example combining the Invoice Exception Processor, Invoice Header Editor, and Invoice Rejection Specialist group, if you want to give exception handlers limited editing capabilities while ensuring an efficient invoice exception handling workflow. You can also associate an exception type with an approval queue by creating an approval queue for the group specified in the Group field. When an approval queue exists for a group, the invoice exception appears as an item in the approval queue instead of on the To Do page of the users who belong to the group. For information on configuring approval queues, see Approval Queues, Approval Escalations, and Withdrawal Periods.

Field	Description
Requester Allowed	A Boolean value that determines if the requester is granted temporary permission to resolve this exception type. In the default implementation, the following invoice exceptions have the **Requester Allowed** flag set to **Yes**: • **Contract Received Line Amount Variance** • **Contract Milestone Amount Variance** • **Contract Received Quantity Variance** • **PO Received Line Amount Variance** • **PO Received Quantity Variance** • **Item Unmatched** If during reconciliation any of these invoice exceptions are found, the requester is added as a watcher to the approval flow. They can then reconcile the exception that they have been granted permission to resolve. How the requester on the order is determined depends on your site's configuration and integration with your ERP: • On orders created in the SAP Ariba procurement solution, the requester is the requester on the original requisition. • On copy orders imported from an ERP system, the requester on the last order line is used to determine the requester on the order.
Header Only	A Boolean value that specifies whether the validation considers only header-level fields. If set to **No**, the validation iterates through the line items on the invoice. Header-only exceptions are not generated for summary (or multi-PO) invoices.
Invoice Source	Specifies the invoices to apply this exception type to, based on the source document for the invoice, such as purchase order or no-release contracts. A value of **All documents** means the exception type applies to invoices from all invoice sources. Any other value means the exception type applies only to invoices with the specified invoice source document. See also **Refine invoice source by conditions**.
Refine invoice source by conditions	Lets you further refine the type of source document to which you want to apply the exception type. For example, buyers might want to raise certain invoice exceptions for service order invoices and other invoice exceptions for invoices for material goods.
Apply to Specific Invoice Types	Lets you limit the types of invoices the exception type applies to: standard invoices, credit memos, line-item credit memos, line-item debit memos.
Line Category	The line category used to filter line items to validate. A value of **Apply to all line categories and line types** means no filtering is performed. Any other value means that the validation applies only to lines of the specified line category. This value is not relevant for header-level exceptions.
Line Type	The procurement line type used to restrict validation to line items of a certain type. This field is hidden if **Line Category** is set to **Apply to all line categories and line types**.
Field Path To Validate	The invoice field being validated. If the **Header Only** field is set to **Yes**, the field path starts from the invoice reconciliation object. Otherwise, the field path starts from the invoice reconciliation line item. If you are defining a custom exception type, the value of the **Field Path To Validate** field must name an extrinsic, which you compute and maintain. Do not specify an internal field. Using internal fields can change the value of those fields, disrupting the logic of the application and causing errors.

Field	Description
Field Path Target	The invoice field to be set when the invoice reconciliation is accepted or disputed. This field is particularly useful when **Field Path To Validate** is a derived field and cannot be used as the target field. If **Field Path Target** is null, **Field Path To Validate** is used as the default target.
Field Path To Validate Against	The invoice field to be used for comparison. This path often names a field on the purchase order, accessible from the invoice reconciliation document.
Field Path Source On Dispute	The path of the field on the invoice reconciliation document or invoice reconciliation document line item that is used to select the value to test on the target field when the invoice exception is disputed.
Null Allowed	A Boolean value that indicates whether a null is allowed in the **Field To Validate Against** field. This field is useful for comparing limits associated with contracts.
Tolerance Operation	A value that specifies how the tolerance is chosen. If the tolerance operation is not specified, the tolerance is not checked and only an exact match is valid.
Absolute Tolerance	A floating-point number that defines an absolute acceptable amount. **Absolute Tolerance** is typically combined with a **Percentage Tolerance**, using the OR operator. For example, if you specify $5 OR 1.05%, items with tax under $5 are ignored, and items with tax over $5 are checked to make sure the amount is within the designated percentage. Absolute tolerance can be a negative number. For example, if you set an absolute tolerance for the **Under Tax Variance** exception type, specify a negative number.
Apply Absolute Tolerance to Difference	If set to **Yes**, applies the value in Absolute Tolerance to the difference between the invoice (**Field Path to Validate**) and the purchase order or contract amount or quantify (**Field Path to Validate Against**) (difference-based absolute tolerance). If set to **No**, the value in **Absolute Tolerance** is applied to the invoice amount or quantity (**Field Path to Validate**) only (invoice-based absolute tolerance).
Percentage Tolerance	A floating-point number that defines the percentage of the field validated against. If the value in **Percentage Tolerance** is 0, only the value in **Absolute Tolerance** is applied.
Absolute Rank	An integer that determines the order for checking exceptions when multiple exceptions exist on the invoice. Exceptions with lesser values have higher rank and are given precedence. For example, the reconciliation engine checks exceptions of rank 1 first. If the reconciliation engine generates a higher-rank exception due to a discrepancy, it does not check the lower-rank exceptions. Only the higher-rank exception is presented to the user. The header exceptions and line exceptions are ranked separately.
Relative Rank	An integer that determines how to rank two exceptions with the same absolute rank and target field (**Field Path Target**). If an invoice has multiple exceptions with the same absolute rank and target path, the relative rank determines the relative precedence of those exceptions. The exception with the lowest precedence is ignored.
Auto Reject	A Boolean value that indicates whether failure of the validation causes the automatic reconciliation engine to reject the invoice reconciliation document. A value of **Yes** means an exception of this type causes the invoice to be auto-rejected.
Ignore Auto Accept	A Boolean value that indicates whether the exception type disables auto-accept. If this field is set to **Yes**, auto-accept is disabled when exceptions of this type are present. If this field is **No**, exceptions of this type have no effect on auto-accept.

Field	Description
Skip Exception	A Boolean value that indicates whether to skip validation for the exception type. Use this option along with the corresponding option on the **Overrides** tab to have the invoice reconciliation process skip validation of an exception type in some cases while applying it in other cases. If this field is set to **Yes**, the reconciliation process skips validation for (ignores) the exception type.
Prevent Invoice Dispute	A Boolean value that indicates whether to disable the **Accept Expected** option for invoice exception types. If this field is set to **Yes**, the **Accept Expected** option is not available for invoice exceptions, preventing users from disputing an exception. If this field is set to **No**, the **Accept Expected** option is available for invoice exceptions, allowing users to dispute an exception. **Note** This field is only applicable if the parameter **Prevent dispute of invoices** (`Application.Invoicing.DisallowDispute`) is set to **No**.

3. Click **Save** to save your changes, or **Cancel** to return to the previous page without saving your changes.

Related Information

About Standard Line Categories [page 35]
Procurement Line Types [page 35]
About Validation Behavior [page 32]
About Tolerance Calculations [page 29]
About Invoice Exception Ranking [page 34]
Skipping validation for an invoice exception type or override [page 32]
Applying invoice exception types to specific types of invoices [page 20]
Defining invoice exception types for specific invoice source documents [page 23]

About configuring invoice exception type overrides

Configuring invoice exception type overrides is a two step process: creating the override, and applying it.

1. Create available exception type overrides by suppliers, commodity code, purchasing unit, or currency. You can create these overrides on the **Invoice Exception Types** page for any invoice exception type.
2. Apply the override to the invoice exception type and specify the absolute and percentage tolerances, and whether to automatically reject invoices that trigger exceptions.

> **Note**
> You cannot create invoice exception type overrides for boolean (yes/no) or field type (Supplier, Commodity) checks on PO Payment Terms mismatched exceptions.

Invoice exception overrides by currency are applied in one of two ways, depending on site configuration:

- In sites that apply exception type absolute tolerance values in the site's default currency (the default behavior), if the site's default currency is USD, an absolute tolerance value of $10 is applied to all invoices, and is converted into the invoice currency for invoices that use non-USD currencies. An invoice manager can create an exception override for JPY of 10; that amount is converted from dollars to yen and then applied to invoices in JPY. In this case, the absolute tolerance amount for the currency override fluctuates with the exchange rate, and if there is a significant change to the exchange rate between when the purchase order is created and invoiced, the invoice reconciliation might generate unintended variance exceptions that must be manually reconciled. It also might not reflect purchasing power or other local conditions for the specific currency.
- In sites that are configured to apply absolute tolerance amounts in the invoice currency rather than the site default currency (the recommended configuration), invoice managers can set an exception type absolute tolerance of 10, which is applied as $10 on a USD invoice and ¥10 on a JPY invoice. The invoice manager can then set an exception type override with an absolute tolerance of 110 for JPY. In this case, the absolute tolerance amount for the currency override is fixed and is applied consistently as ¥110. It can be adjusted up or down to reflect purchasing power or other local conditions for the specific currency, rather than being tied to the site's default currency and the current exchange rate.

Applying an override to an invoice exception type

You can apply an existing override to an invoice exception type.

Prerequisites

An exception override must exist in the **Available Overrides** table of the **Overrides** tab before you can apply it.

Procedure

1. On the **Invoice Exceptions Type** page, use **List All** to show all invoice exceptions types, or use the search filter to find the invoice exception type you want to apply an override for. Select **Edit** from the **Actions** button.
2. Click the **Overrides** tab.
3. Click **Create New** in the **Applied Overrides** section.
4. Select one of the available overrides from the pull-down menu.
5. Assign the user group responsible to resolve any invoice exceptions for this override (for example, **Invoice Agent**).
6. Specify the percentage tolerance and absolute tolerance values.
7. Specify whether or not to auto-reject the invoice when this invoice exception occurs for the supplier or commodity code on the invoice.
8. Use the **Applied Driver Rank** column to make sure each applied override has a different rank in the **Applied Overrides** table.

> **Note**
>
> Changing the values in the **Applied Driver Rank** column doesn't affect the rank assigned to the override in the **Available Overrides** table.

9. Click **Save**.

Related Information

How to configure an invoice exception type to be skipped [page 43]

How to configure an invoice exception type to be skipped

You can have the invoice reconciliation process skip validation of an invoice exception type in some cases while applying it in other cases.

Prerequisites

You must be a member of the **Invoice Administrator**, **Payment Administrator**, **Invoice Manager**, **Payment Manager**, **Procurement Manager**, or **Customer Administrator** user group in order to manage invoice exception types in Ariba Administrator.

Context

Configuring invoice reconciliation to skip an exception type in some cases but not others typically involves setting the **Skip Exception** option one way by default and setting it the opposite way in an override. In other words:

- Setting the **Skip Exception** field to **Yes** by default and applying an override where **Skip Exception** is set to **No**.
- Setting the **Skip Exception** field to **No** by default and applying an override where **Skip Exception** is set to **Yes**.

This procedure shows how to configure an exception type to be skipped using the **Invoice Exception Types** task in Ariba Administrator. You can do the same thing using the **Import Invoice Exception Type Validation Data** and **Import Invoice Exception Type Override Validation Data** data import tasks.

Procedure

1. On the dashboard, choose **Manage** > **Core Administration**.
2. Perform one of the following actions:

- In SAP Ariba Buying and Invoicing, choose **Procure-to-Pay Manager** > **Invoice Exception Types**.
- In SAP Ariba Invoice Management, choose **Invoicing Manager** > **Invoice Exception Types**.

3. Find the invoice exception type you want to modify, and choose **Actions** > **Edit**.
4. On the **Default** tab, scroll down to the **Skip Exception** field, and choose **Yes** or **No**:
   - To have the reconciliation process skip validation for this exception type by default, choose **Yes**. You can create an override for the cases where you want to validate for the exception type.
   - To have the reconciliation process validate for this exception type by default, choose **No**. You can create an override for the cases where you want to skip validation.
5. On the **Overrides** tab, if you don't see the **Skip Exception** column in the **Invoice Exception Type Default Values** table and the **Applied Overrides** table, use the **Table Options** icon to display it.
6. In the **Available Overrides** section, click **Create New** to create an override for skipping (or not skipping) validation for the exception type.
7. In the **Applied Overrides** section, click **Create New** to apply the override you just created.
8. Set the **Skip Exception** checkbox as follows:
   - If you set **Skip Exception** to **Yes** on the **Default** tab, and you want the override to **apply** validation, leave the **Skip Exception** checkbox unchecked.
   - If you set **Skip Exception** to **No** on the **Default** tab, and you want the override to **skip** validation, check the **Skip Exception** checkbox.
9. Adjust the ranks in **Applied Driver Rank** column of the **Applied Overrides** table as needed to ensure that each applied override has a different rank.

Related Information

Skipping validation for an invoice exception type or override [page 32]

How to view or delete invoice exception types

You can see a list of your existing invoice exception types and view their details. if you don't need an invoice exception type, you can delete it.

Prerequisites

Before deleting any invoice exception types, export your invoice exception type data to CSV files in case you need to add them back later by running the **Export Invoice Exception Types** and **Export Invoice Exception Type Validation Data** data export tasks.

Procedure

1. Choose **Ariba Procure-to-Pay Manager** > **Invoice Exception Types** (SAP Ariba Buying and Invoicing) or **Payment Manager** > **Invoice Exception Types** (SAP Ariba Invoice Management) and click **List All** to display all invoice exception types, or enter search criteria and then click **Search** or press the Enter key.

   The following table describes the columns available on the **Invoice Exception Types** page.

Column	Description
Name	The display name for the invoice exception type.
Description	The user-visible description for the invoice exception type.
Header Only	When checked, this column indicates that the validation considers only header-level fields. When unchecked, this column indicates that the validation iterates through the line items on the invoice.
Invoice Source	The invoices to apply this exception type to, based on the source document for the invoice, such as purchase order or no-release contracts.
Invoice Type Restriction	The types of invoices to which the exception type applies (standard, credit memo, line-item credit memo, line-item debit memo). A value of **All** means the exception type applies to all types. This column corresponds to the field **Apply to Specific Invoice Types** on the details page of invoice exception types.

2. To view the details for an exception type, click its name in the **Name** column. To delete it, select the invoice exception type and click **Delete**.
3. Click **Done** to return to the previous page.

Data import tasks for invoice exception types

Data import tasks define invoice exception types and specify whether the data is required. Data import tasks are recommended if you are creating multiple exception types at the same time.

The following table lists data import task in the recommended order of use.

Data Import Task	CSV Files Imported	Required
About Import Invoice Exception Types [page 46]	`InvoiceExceptionType.csv`	No
About Import Invoice Exception Type Validation Data [page 47]	`InvoiceExceptionTypeValidation.csv`	No
About Import Invoice Exception Overrides by Common Commodity Code [page 48]	`InvoiceExceptionTypeCCCOverride.csv`	No
Import Invoice Exception Overrides by Supplier [page 49]	`InvoiceExceptionTypeSupplierOverride.csv`	No
Import Invoice Exception Overrides [page 49]	`InvoiceExceptionTypeOverrideDriver.csv`	No

Data Import Task	CSV Files Imported	Required
Import Invoice Exception Type Override Validation Data task [page 50]	`InvoiceExceptionTypeOverrideValidation.csv`	No

You manually enter data in these CSV files.

About import invoice exception types

You can use the `Import Invoice Exception Types` data import task to define your own set of invoice exception types. This task reads data from the CSV file named `InvoiceExceptionType.csv`.

Following is an example of `InvoiceExceptionType.csv`:

```
UTF8
UniqueName,Name,Description,Group,RequesterAllowed
UnmatchedInvoice,Invoice Unmatched,Ariba Invoice is unable to match a purchase
order to the invoice of amount {1}.,Invoice Manager
```

About Translations for Invoice Exception Types

You can use the `Import Invoice Exception Type Translations` data import task to load translations for invoice exception type names and descriptions.

Additionally, for each exception type, you can use this task to customize the preprogrammed names of the accept and the dispute exception-handling actions to meet the requirements of your organization. For example, in the default configuration, the accept and dispute actions to resolve a PO Received Quantity Variance exception appear in the **Action** dropdown menu on an invoice reconciliation document as follows:

- Accept Invoice Quantity
- Accept Expected Quantity

In this scenario, you can choose to configure a different name for either or both of these action labels. For example, you can choose to configure a different name, Accept Received Quantity, instead of Accept Expected Quantity.

The following is a sample of the `InvoiceExceptionType.csv` file used by the `Import Invoice Exception Type Translations` data import task:

```
UTF8
Name,UniqueName,Description,AcceptString,DisputeString,Language
Over Tax Variance,OverTaxVariance,The Amount field on the invoice
tax ,AcceptingOverTax_Custom,Not Accepting OverTax_Custom,English
Under Tax Variance,UnderTaxVariance,The Amount field on the invoice tax ,Accepting
UnderTax_Custom,Not Accepting UnderTax_Custom,English
```

About the Import Invoice Exception Type Validation Data task

Use the **Import Invoice Exception Type Validation Data** data import task to define your own invoice exception type validation data. This data defines exactly which fields to compare and what tolerances to allow for each type of invoice exception.

You can configure behavior such as:

- Which fields to compare when determining if an exception exists. For example, you can specify that an amount variance exception occurs when the field `TotalInvoiced` on the invoice does not match the field `Order.AmountOrderedLessInvoiced` on a purchase order.
- Whether the exception applies to the header or to line items, and whether it applies to all line items or only line items of a certain line item type or category.
- Whether the exception applies to all types of invoices or specific types of invoices only (standard, credit memo, line-item credit memo, line-item debit memo).
- Whether the exception is filtered by the type of source document that the invoice is based on. For example, an exception might apply only to invoices based on purchase orders, or to contracts without release orders.
- Whether the exception includes a tolerance buffer. You can define tolerances either as absolute amounts, or percentage amounts, or both. You can combine tolerance values with the operators AND, OR, and ANY.
- Whether an exception type is related to other exception types, and which takes precedence when the same exception can potentially be described in several ways. For example, if an invoice has unmatched item exceptions, it probably also has quantity exceptions, amount exceptions, and price exceptions. Displaying the unmatched item exception first makes sense, because resolving the first exception might resolve the other exceptions as well.
- Whether an exception triggers an automatic rejection during the automatic reconciliation phase.
- Whether to skip validation for the exception type (if you want to apply it only through an override rather than by default).
- Whether an exception type can be disputed or if the user must accept the exception type.

The `Import Invoice Exception Type Validation Data` data import task reads data from the CSV file named `InvoiceExceptionTypeValidation.csv`.

Following is an example of `InvoiceExceptionTypeValidation.csv`:

```
UTF8
UniqueName,HeaderOnly,InvoiceCategory,SpecialConditions,LineCategory,LineType,FieldP
athToValidate,FieldPathTarget,FieldPathToValidateAgainst,FieldPathSourceOnDispute,Nu
llAllowed,ToleranceOperation,AbsoluteTolerance,PercentageTolerance,AbsoluteRank,Rela
tiveRank,AutoReject,QuantifiableOnly,SpendCategoryDriver,SpendCategory,IgnoreAutoAcc
ept,ApplyAbsoluteToleranceToDifference,InvoicePurposes,SkipException,AppliedDriverRa
nk,PreventInvoiceDispute
UnmatchedInvoice,TRUE,3,ServiceOrder,0,TotalCost,IsMatched,IsMatched,FALSE,boolean,0
,0,1,1,FALSE,FALSE,,,FALSE,Standard,FALSE,-1,FALSE
```

> **Note**
>
> The following fields in the pre-defined default exception types are not editable: `FieldPathToValidate`, `FieldPathToValidateAgainst`, `FieldPathSourceOnDispute`, and `FieldPathTarget`. The fields in custom invoice exception types remain editable.

About Percentage Tolerance behavior

The `PercentageTolerance` column specifies a percentage of the field path to validate against (`FieldPathToValidateAgainst` column).

Following is an example of `InvoiceExceptionTypeValidation.csv` for the POLineReceivedAmountVariance invoice exception with a 10% percentage tolerance setting.

```
"UniqueName","HeaderOnly","InvoiceCategory","LineCategory","LineType","FieldPathToVa
lidate",
"FieldPathTarget","FieldPathToValidateAgainst","FieldPathSourceOnDispute","NullAllow
ed",
"ToleranceOperation","AbsoluteTolerance","PercentageTolerance","AbsoluteRank","Relat
iveRank",
"AutoReject","QuantifiableOnly","SpendCategoryDriver","SpendCategory","IgnoreAutoAcc
ept",
”ApplyAbsoluteToleranceToDifference”
"POLineReceivedAmountVariance",false,3,1,"","OrderLineItem.AmountInvoiced","Amount",
"OrderLineItem.AmountAcceptedInvoiceable","RLIAmountAcceptedLessPreviouslyInvoiced",
true,
"or",1,1.1,4,3,false,false,"","",false
```

In this exception:

- `FieldPathToValidate` is set to `OrderLineItem.AmountInvoiced` (the invoiced amount)
- `FieldPathToValidateAgainst` is set to `OrderLineItem.AmountAcceptedInvoicable` (the accepted amount)
- `PercentageTolerance` is set to `1.1` (110 percent)

Suppose the accepted amount is 116,169.90 and the invoiced amount is 124,690.20. In this case, multiplying the accepted amount by 110 percent results in 127,786.89:

```
116,169.90 * 1.1 = 127,786.89
```

Because 124,690.20 (the invoiced amount) is less than 127,786.89 (the accepted amount multiplied by 110 percent), the exception is not generated. In other words, the `POLineReceiveAmountVariance` exception is generated only if the invoiced amount is more than 10 percent larger than the accepted amount.

About import invoice exception overrides

You can define your own set of invoice exception overrrides using data import tasks.

Import Invoice Exception Overrides by Common Commodity Code

Use the **Import Invoice Exception Overrides by Common Commodity Code** data import task to define your own set of invoice exception overrides by commodity codes. This task reads data from the CSV file named `InvoiceExceptionTypeCCCOverride.csv`.

Following is an example of `InvoiceExceptionTypeCCCOverride.csv`:

```
UTF8
```

```
Rank,CommodityCodeDomain,DriverName,CommodityCodeId
3,unspsc,OverrideforComputers,4321
```

Import Invoice Exception Overrides by Supplier

Use the **Import Invoice Exception Overrides by Suppliers** data import task to define your own set of invoice exception overrides by suppliers. This task reads data from the CSV file named `InvoiceExceptionTypeSupplierOverride.csv`.

Following is an example of `InvoiceExceptionTypeSupplierOverride.csv`:

```
UTF8
Rank,SupplierUniqueName,DriverName
3,11,OverrideforJCN
```

Import Invoice Exception Overrides

Use the **Import Invoice Exception Overrides** data import task to define your own set of invoice exception overrides by purchasing unit and currency. This task reads data from the CSV file named `InvoiceExceptionTypeOverrideDriver.csv`.

This file contains the following fields:

`DriverName`	A name for either the purchasing unit or currency override; for example, `PUOverride`.
`FieldPath`	`PurchasingUnit` for a purchasing unit-level override or `Currency` for a currency-level override.
`Rank`	The numerical rank for the override, which determines which override takes precedence when an exception has multiple overrides defined for it. Lower numbers denote a higher priority; higher numbers denote a lower priority.
`Key`	The unique name for the procurement unit or currency as defined in your site.

For purchasing units, as for commodities, if you define an invoice exception override for a purchasing unit in one part of a hierarchy, the override applies for all purchasing units underneath it in the hierarchy. For example, an override for a purchasing unit for the entire United States applies to any state-specific purchasing units underneath it in the hierarchy.

Following is an example of `InvoiceExceptionTypeOverrideDriver.csv`:

```
UTF-8
DriverName,FieldPath,Rank,Key
CurrencyOverride,Currency,10,BHD
PUOverride,PurchasingUnit,20,US100
```

Import Invoice Exception Type Override Validation Data task

Use the **Import Invoice Exception Type Override Validation Data** task to define the specific tolerance override values to apply to an invoice exception for specific suppliers or specific commodity codes. For each invoice exception, you can override the following values:

For each invoice exception, you can override the following values:

- Auto-acceptance and auto-rejection of invoices if an exception is found based on this override
- Percentage and absolute tolerance value overrides
- The relative rank of the override, which is used to determine the override to apply if multiple overrides are defined for an invoice exception
- The group that is added to resolve invoice exceptions that result from applying the override. For example, this allows the invoice administrators to add specific groups to the approval flow that are responsible to resolve invoice exceptions with specific suppliers or for certain commodities.
- Whether to skip validation for the exception type.
- Whether an exception type can be disputeed or if the user must accept the exception type.

Use the `AppliedDriverRank` column to assign a rank to each applied override, making sure that each one has a different rank.

This task reads data from the CSV file named `InvoiceExceptionTypeOverrideValidation.csv`. The following shows an example of this file:

```
UTF-8
UniqueName,IgnoreAutoAccept,PercentageTolerance,DriverName,RelativeRank,AutoReject,G
roup,AbsoluteRank,AbsoluteTolerance,SkipException,AppliedDriverRank,PreventInvoiceDi
spute
POPriceVariance,No,3,OverrideforJCN,1,No,Invoice Agent,3,1,FALSE,0,TRUE
POPriceVariance,No,2,OverrideforComputer,1,No,Invoice Agent,3,1,TRUE,1,FALSE
```

Related Information

Adding or Modifying Invoice Exception Types [page 37]

Forcing reconciliation to change the status of an invoice

In some cases, you might need to force an invoice to be reconciled.

Invoices originating from SAP Ariba Procurement solutions are routed to your supplier, but by default, they're not tracked on SAP Business Network. Your SAP Business Network account might be configured to store a copy of the invoice originating from SAP Ariba Procurement solutions and route that copy to your supplier. The supplier can then view the status of their invoices and monitor their progress through your invoice reconciliation process.

If the system fails to route the invoice copy, the invoice can stall and remain in the **CCInvoicetoAN** status. To resolve such cases, a **Force Reconcile** button is available when you view the details of an invoice. Click **Force**

Reconcile to manually change the status from **CCInvoicetoAN** to **Reconciling**. The status of all related invoices also changes to **Reconciling**.

> ⓘ Note
>
> When you click **Force Reconcile**, a warning that explains what the force reconcile action does and doesn't do is displayed. For more information, see Enhanced warnings for force actions on invoices and payments.

Only members of the **Invoice Administrator** or **Customer Administrator** group can force reconcile an invoice.

How to re-reconcile an invoice reconciliation against updated invoice exception types

Misconfigured invoice exception types can cause unnecessary exceptions to occur. After the exception types are corrected, invoice administrators can use the **Update Exceptions** button to re-reconcile the IR against the exception types.

Prerequisites

Members of the **Invoice Administrator** or **Customer Administrator** group can perform this task.

Procedure

1. From the dashboard, find and open the IR document, and then click **Edit**.
2. Click **Update Exceptions**.
3. Click **Save** or **Submit** to save the changes.

Holding line-item credit and debit memos until the ERP system receives the invoice

You can configure invoice exception types to prevent the SAP Ariba solution from sending line-item credit and debit memos to the ERP system before the ERP system has received the associated invoice.

- Make sure your site allows line-item credit memos to raise exceptions during invoice reconciliation. The site configuration parameter that controls this is `Application.Invoicing.SkipLineCreditMemoExceptions`, which SAP Ariba sets for you.
- In Ariba Administrator, find and activate the exception type **Credit Memo References Unapproved Invoice Reconciliation**.

Make sure to leave the **Header Only** field set to **Yes** for this exception type so that individual lines are accumulated even if this exception is raised. (This makes it possible for lines on the credit memo to clear exceptions on the original IR.) Leave the **Restrict to Specific Invoice Types** field set to **Yes**, with **Line Item Credit Memo** selected.

- If your site accepts line-item debit memos (used for price increases from suppliers), find and activate the exception type **Debit Memo References Unapproved Invoice Reconciliation**.
  Leave **Restrict to Specific Invoice Types** set to **Yes**, with **Line Item Debit Memo** selected.
- To prevent users from accepting line-item credit and debit memos with these exceptions, have a member of the **Customer Administrator** group add the following values to the self-service parameter **Prevent acceptance of specific exceptions** (`Application.Invoicing.DisallowAcceptOnExceptionTypes`):
  - `CreditMemoReferencesUnapprovedInvoice`
  - `DebitMemoReferencesUnapprovedInvoice`

  Members of the **Customer Administrator** group can access self-service parameters in Ariba Administrator under **Intelligent Configuration Manager** > **Manage Configurations**.

> **Note**
>
> For sites that were enabled after this feature was released, the parameter **Prevent acceptance of specific invoice exceptions** already includes `CreditMemoReferencesUnapprovedInvoice` and `DebitMemoReferencesUnapprovedInvoice`.

Workflow

The following high-level steps describe how invoice exception types prevent line-item credit and debit memos from reaching the ERP before the original invoice.

1. The supplier sends an invoice.
2. SAP Business Network sends the invoice to the SAP Ariba solution.
3. The SAP Ariba solution creates an invoice reconciliation (IR) document.
4. The supplier sends a line-item credit memo or line-item debit memo for the original invoice.
5. SAP Business Network sends the line-item credit or debit memo to the SAP Ariba solution.
6. Upon receiving the line-item credit or debit memo, the SAP Ariba solution checks the status of the associated invoice.
   - If the invoice has a status of **Paying**, **Paid**, or **Paying Failed**, the line-item credit or debit memo proceeds with the usual reconciliation and approval process. (These three statuses indicate that the ERP system has received the invoice.)
   - If the invoice has any other status, one of the following exceptions is raised: **Credit Memo References Unapproved Invoice Reconciliation** or **Debit Memo References Unapproved Invoice Reconciliation**.

     > **Note**
     >
     > For line-item credit memos, individual lines are accumulated even if the **Credit Memo References Unapproved Invoice Reconciliation** exception is raised, because the exception occurs at the header level, not the line level. This makes it possible for lines on the credit memo to clear exceptions on the original IR. For example, suppose an IR has a price variance exception, and a line-item credit memo brings the price down within the tolerance. In this case, the exception on the original invoice is cleared even though the credit memo won't be fully processed until the ERP system receives the invoice.

As soon as the invoice moves to a status of **Paying**, **Paid**, or **Paying Failed**, the line-item credit or debit memo is rereconciled, which resolves the initial exception and regenerates the approval flow.

7. Once any exceptions for the line-item credit or debit memo is resolved, and the documents are approved, the SAP Ariba sends the documents to the ERP system.

How to configure exception types for price-adjustment debit and credit memos

If your site supports price-adjustment debits and credits, consider making the following modifications to invoice exception types.

Prerequisites

- Members of the **Customer Administrator** and **Invoice Administrator** groups can perform this task.
- Make sure your site allows line-item credit memos to raise exceptions during invoice reconciliation. The site configuration parameter that controls this is `Application.Invoicing.SkipLineCreditMemoExceptions`, which SAP Ariba sets for you.

Context

It's important that line-level credit memos (for price or quantity adjustments) and line-level debit memos are **not** processed until the original invoice is reconciled, approved, and fully posted in the ERP system. To ensure this sequence, you activate exception types that raise an exception if a line-level debit or credit memo arrives before the IR for the original invoice is in **Paid**, **Paying**, or **Paying Failed** status.

We recommend additional changes to ensure that price variance exceptions take price adjustments into account.

Procedure

1. From the dashboard, choose **Manage** > **Core Administration**.
2. Perform one of the following actions:
   - In SAP Ariba Buying and Invoicing, choose **Procure-to-Pay Manager** > **Invoice Exception Types**.
   - In SAP Ariba Invoice Management, choose **Invoicing Manager** > **Invoice Exception Types**.
3. Find and activate the following invoice exception types:
   - **Credit Memo References Unapproved Invoice Reconciliation**
   - **Debit Memo References Unapproved Invoice Reconciliation**

4. If you're using a **Receipt Unit Price Variance** exception type (for goods receipt-based invoice verification), a **PO Price Variance** exception type, or a **Contract Price Variance** exception, change the **Field Path To Validate** setting from `Description.Price` to `PriceIncludingReferencedInvoiceAdjustments`.

   This change ensures that the adjusted price is taken into account during invoice re-reconciliation. That way, a price-adjustment credit can clear this price variance exception on an invoice, depending on the size of the price adjustment. Price-adjustment debits, on the other hand, can trigger a price variance exception.

   > ⓘ Note
   >
   > If you're using the default price variance exception types rather than custom exception types, you can't change the **Field Path To Validate** setting. In order to do this step, you create a new invoice exception type that duplicates the default exception type, except that the **Field Path To Validate** setting is set to `PriceIncludingReferencedInvoiceAdjustments`.

5. If you're using price variance exception types, edit the **Apply to Specific Invoice Types** field of those exception types to exclude line-item credit memos. Select only the following invoice types: **Standard**, **Credit Memo**, **Line Item Debit Memo**.

   Otherwise, price variance exceptions can be raised for price-adjustment line-item credit memos, which is unnecessary because the price variance exception on the original invoice is sufficient.

6. (Optional) If you're using the **Receipt Quantity Variance** exception type (for goods receipt-based invoice verification), apply it to standard invoices only using the **Apply to Specific Invoice Types** field.

   This prevents the exception from being raised on a price-adjustment debit or credit, which don't change the quantity. It's sufficient to raise this exception on the original invoice only.

Next Steps

Have a member of the **Customer Administrator** group make sure that the exception types **Credit Memo References Unapproved Invoice Reconciliation** and **Debit Memo References Unapproved Invoice Reconciliation** have been added to the parameter **Prevent acceptance of specific invoice exceptions**(`Application.Invoicing.DisallowAcceptOnExceptionTypes`) to prevent users from accepting price-adjustment debits and credits that raise the exceptions.

Configuring exceptions that check for pending credit memos

If you've implemented approval processes for credit memos, you might not be notified of pending credit memos for invoices with quantity- or price-related discrepancies. For example, if the line item quantity is greater than allowed by the contract, you might have expected to see the **Contract Received Quantity Variance** or the **PO Received Quantity Variance** exception raised, but the exception wasn't raised because of a pending credit memo against the invoice. You can configure an exception type to alert invoice agents when there's a pending credit memo against the contract or purchase order.

Following are two models for configuring invoice exception types based on the Boolean field, `HasNoCreditMemo`, which determines whether pending credit memos exist for the contract or order.

Example settings for an invoice exception type for credit memos against a contract:

Name	Contract has open credit memo
Description	An open credit memo exists for the same contract
Field Path To Validate Against	`HasNoOpenCreditMemo`
Field Path Source On Dispute	`HasNoOpenCreditMemo`
Header Only	Yes
Invoice Category	Applies to contracts without release orders only
Line Category	Applies to regular line types

Example settings for an invoice exception type for credit memos against a purchase order:

Name	Purchase order has open credit memo
Description	An open credit memo exists for the same purchase order
Field Path To Validate Against	`HasNoOpenCreditMemo`
Field Path Source On Dispute	`HasNoOpenCreditMemo`
Header Only	Yes
Invoice Category	Applies to contracts without release orders only
Line Category	Applies to regular line types

> **⚠ Restriction**
>
> The invoice exception types you create with `HasNoOpenCreditMemo` don't apply to consolidated invoices

Configuring goods receipt-based invoice verification

A member of the **Customer Administrator** group configures goods receipt-based invoice verification by setting parameters in Intelligent Configuration Manager workspace.

To enable goods receipt-based invoicing:

- Set the parameter Enable goods receipt-based invoice verification (`Application.Procure.EnableGRBasedInvoice`) to **Yes** to turn on goods receipt-based invoice verification.
- Set the parameter Use receipt unit price to validate invoice unit price (`Application.Procure.EnablePriceDateControl`) to **Yes** so that the SAP Ariba solution uses the unit price on the receipt for price verification on invoices.

Also configure the following parameters as needed for your business requirements:

- Set the parameter Match receipts to invoices without ship notices (`Application.Invoicing.MatchReceiptToInvoiceWithoutShipNotice`)
- Include matched receipts in payment request export (`Application.Invoicing.IncludeMatchedReceiptsInPaymentExport`)

- Allow edits to matched quantity in receipt matching (`Application.Invoicing.AllowMatchedReceiptQuantityEdit`)
- Days to delay creating an IR document (`Application.Invoicing.DaysToDelayIRCreation`)
- Application.Invoicing.RespectMatchedReceiptsFromAribaNetwork (set by SAP Ariba Support)

  In sites enabled for goods receipt-based invoice verification, determines if the invoicing solution uses the receipt information in SAP Business Network invoices when matching line items to receipts. If set to Yes, the invoicing solution matches invoice line items to receipts based on receipt information in the invoice. If set to No, matching is based on receipt information on the purchase order (PO). For invoices without receipt information, matching is always based on the PO. A setting of No can lead to unexpected results for PO line items with multiple receipts.

On SAP Business Network, enable the following invoice transaction rules:

- **Require suppliers to include only shipped quantities on invoices** ensures that suppliers can't create invoices for more than they've shipped. This rule is visible only if the rule **Require suppliers to include only received quantities on invoices** is disabled.
- **Require references to ship notices on invoices** ensures that the invoice includes references to ship notices for each line item. If there are multiple ship notices for an order, suppliers can select one or more ship notices to create the invoice for. This rule is visible if **Require suppliers to include only shipped quantities on invoices** is selected.

If your SAP Ariba Buying and Invoicing site is integrated with an SAP connected system through SAP Integration Suite, managed gateway for spend management and SAP Business Network, set the parameter Web service for exporting payment requests asynchronously (`Application.Invoicing.PaymentAsyncEvent`) to `PaymentAsyncExport_v2` and configure the integration events **Export Payment Requests with Receipts Asynchronously** and **Import Status of Payment Requests with Receipts Asynchronously**.

Where goods receipt-based invoice verification shows up in the user interface

Changes to requisitions and orders

Each line item on a submitted requisition, and on the resulting order, includes the read-only field **Goods Receipt-based Invoice Verification**. This field indicates whether the invoice line for the item will be reconciled using goods receipt-based invoice verification. The field is set to **Yes** or **No** based on supplier location. Users can see this field when they view line-item details.

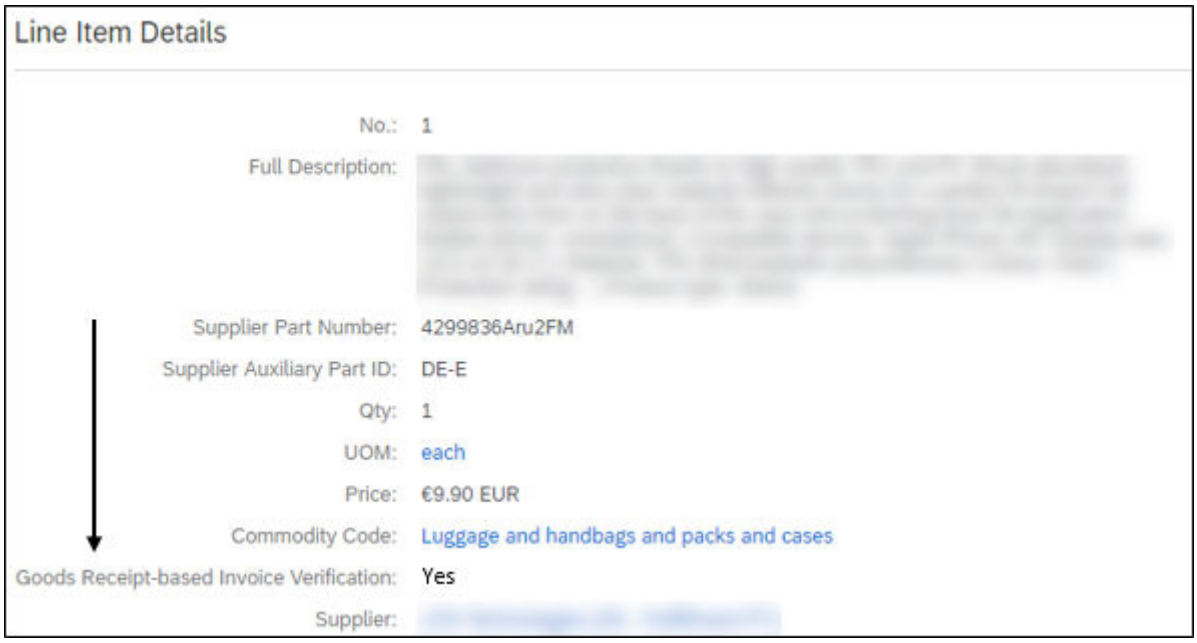


Figure 1: Indicator of goods receipt-based invoice verification in requisition and order line items

When the **Goods Receipt-based Invoice Verification** field is set to **Yes** for an item, receipts for the item include a field for specifying a ship notice or other shipment ID, and invoices are matched to receipts.

For supplier locations that aren't configured for goods receipt-based invoice verification, the value of this field is **No**.

Changes to receipts

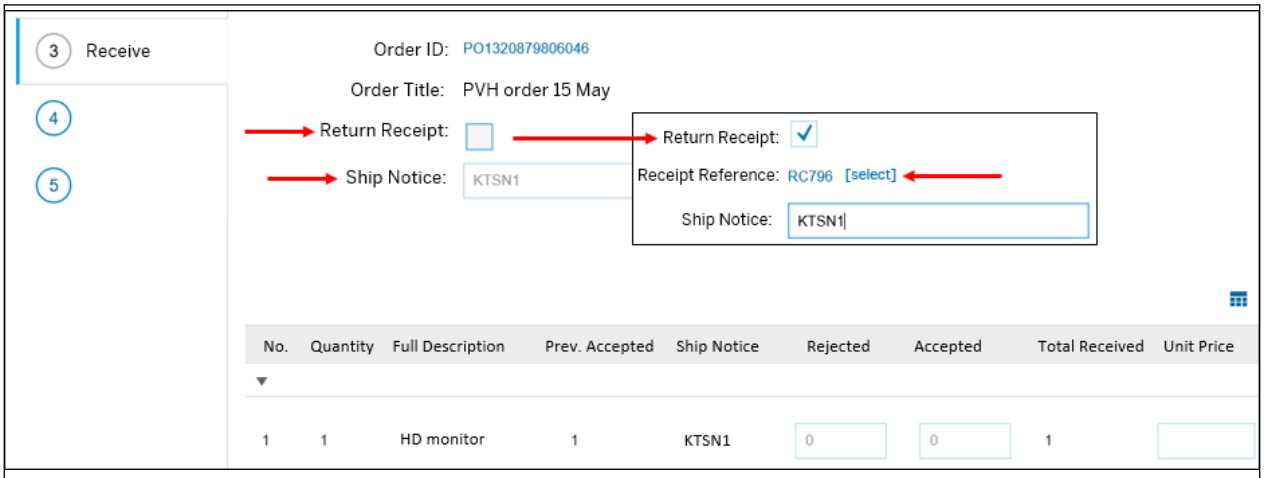


Figure 2: Receiving items marked for goods receipt-based invoice verification

- Receipts created for items marked for goods receipt-based invoice verification include the following fields:
  - A text field called **Ship Notice** in the header, for specifying a ship notice, delivery note, or other shipment identifier. The value the user enters is copied to all receipt lines.
  - A **Return Receipt** checkbox to indicate if the receipt is a return receipt. When **Return Receipt** is checked, an additional field, **Receipt Reference**, identifies the receipt from which the items are being returned so the system can then adjust the actual accepted quantity on the original receipt. On return receipts:
    - The accepted quantity is a negative value.

    - You can't return more than the actual accepted quantity of the original receipt.
    - You can't change the rejected quantity and amount.
- The details of line items on the **Summary** tab on receipts can include the following columns:
    - **Ship Notice**: The ship notice, delivery note, or other shipment identifier associated with a line item.
    - **Unit Price**: The unit price of a line item on the receipt. This price might be different from the unit price of the item on the associated order.
      With goods receipt-based invoice verification, the SAP Ariba solution uses the unit price on receipts to validate the unit price on invoices, as long as the parameter `Application.Procure.EnablePriceDateControl` is set to **Yes** (recommended). The **Unit Price** field for receipt line items is editable unless the receipt is a return receipt. If you leave it blank, the receipt gets the unit price from the receipt (or from the order, if the parameter is set to **No**).

  By default, the **Ship Notice** and **Unit Price** columns aren't displayed on the receipt page. To include these columns, select **Ship Notice** and **Unit Price** from the **Table Options** menu .
- **Available Qty** on the receipt is recalculated for returns associated with it (return quantity is deducted from previous available quantity when the return is processed).
- Receipts include an **Invoices** tab to display the details of the associated invoices.

Changes to invoices

The **Invoice Entry** page for PO-based invoices includes a **Receipts** chooser if the selected purchase order contains at least one item marked for goods receipt-based invoice verification. The **Receipts** chooser includes associated processed receipts that have an uninvoiced quantity remaining. Users can search for the receipt they want based on receipt ID, the value in the receipt's **Ship Notice** field, or an external receipt ID.

For each receipt selected, a line is added to the invoice. This receipt information is carried to the invoice reconciliation (IR) document.

The **Line Items** table on the invoice **Summary** tab includes the following columns:

- **Receipts**: If the invoice line is matched to a single receipt, this column shows the receipt ID. If the invoice line is matched to multiple receipts (including return receipts), this column indicates **Multiple** and shows the number of associated receipts, for example, **Multiple (3)**. This value is a link that opens the details of the associated receipts in a read-only view.
- **Ship Notice**: The ID of the ship notice, delivery note, or other shipment document associated with a line item on the invoice.

If a reconciliation delay period is configured (through the parameter `Application.Invoicing.DaysToDelayIRCreation`), goods receipt-based invoices without at least one associated receipt initially have a status of **Awaiting Receipts**. Invoices in this status include a **Start Reconciliation** button that members of the **Invoice Administrator** group can click to force the reconciliation process to start.

> **Note**
>
> When an invoice administrator clicks the **Start Reconciliation** button, the reconciliation process starts. There's no onscreen confirmation.

Changes to IR documents

- There's a **Ship Notice Title** on the **Exceptions** and **Line View** tabs
- You can display a **Receipts** column on the **Line View** tab to show the receipts associated with invoice lines.
- The **Line-Item Details** page has a **Ship Notice** field.

- Invoice lines with an exception related to goods receipt-based invoicing include the action **Manual Match to Receipts**. Choosing **Manual Match to Receipts** to reconcile an invoice line opens a page that lists the receipts that have been matched to the line and additional processed receipts that are available for matching. For details, see Information on the Manual Match to Receipts Page.

Changes to the Manual Match of Line Items page

Similar to the invoice page, the **Manual Match of Line Items** page for matching invoice lines with order lines can include **Receipt ID** and **Ship Notice** columns. If you don't see the new columns, click the **Table Options** icon to display them.

> **Note**
>
> The **Manual Match of Line Items** page appears when you select **Action** > **Manual Match to Order** on the **Line View** tab on the invoice reconciliation page.

Restrictions

- This feature works with manual, quantity-based receiving. It doesn't support amount-based receiving or automatic receiving.
- This feature supports PO-based invoices. It doesn't support invoices for release contracts, invoices for service orders, or invoices created through evaluated receipt settlement (ERS). The feature hasn't been tested with summary (multi-PO) invoices.
- To export payment requests with information about matched receipts, use Web services. This feature doesn't support exporting payment requests in CSV files.

Related Information

Goods Receipt-Based Invoice Verification

How to delay invoice reconciliation for goods receipt-based invoices

Buyers who create goods receipts **after** receiving invoices from suppliers can delay the automatic start of reconciliation for a specified number of days, preventing invoice lines from triggering the **Matching Receipt Not Found** exception too soon in the context of your business processes.

Prerequisites

Members of the **Customer Administrator** group can perform this task.

Context

Electronic invoices generally arrive in the SAP Ariba solution before the goods are received, triggering the exception **Matching Receipt Not Found**. This exception would occur for most invoice lines against order items that have **Receipt Required** turned on. If you don't want this exception to be raised so often, you can delay the start of invoice reconciliation using the parameter **Days to delay creating an IR document** (`Application.Invoicing.DaysToDelayIRCreation`).

Procedure

1. On the dashboard, choose **Manage** > **Core Administration**.
2. Choose **Intelligent Configuration Manager** > **Manage Configurations**.
3. Choose **Buying and Invoicing** > > **Parameters**.
4. Find the parameter **Days to delay creating an IR document** (`Application.Invoicing.DaysToDelayIRCreation`).
5. In the **New Value** field, enter an integer that represents the number of days you want to delay the start of reconciliation when supporting documents required for automatic invoice reconciliation are still outstanding.
6. Click **Save**.

Results

When a goods receipt-based invoice is received and approved in the SAP Ariba solution, and receipts are still outstanding, the invoice remains in **Awaiting Receipts** status during the delay period. The invoice reconciliation document is created within 6 hours after the delay period ends or when an invoice administrator manually starts the reconciliation process (by clicking the **Start Reconciliation** button).

Invoice exception types for goods receipt-based invoices

There are several default line-level invoice exception types specifically for use with goods-receipt based invoice verification.

Exception type	Situation that triggers the exception	Example
Receipt Unit Price Variance	The unit price of a line item on an invoice doesn't match the unit price of the item on the receipt, and the difference is greater than the tolerance defined for the exception.	• **Receipt Unit Price Variance** tolerance set to 10%. • 10 keyboards were ordered, invoiced, and received. • Price on invoice: 120 USD. • Price on receipt: 100 USD. **Receipt Unit Price Variance** exception is triggered because price difference of 20 is greater than 10%.
Receipt Quantity Variance	The quantity for a line item on the invoice doesn't match the quantity for the item on receipts that reference the same ship notice, and the difference is greater than the tolerance defined for the exception. This exception is also triggered if the quantity for the invoice item's matching receipt item has already been fully allocated. ⓘ Note If a quantity variance exception is raised on an invoice **without** a ship notice, and the invoice was automatically matched to receipts on the PO, the only way to resolve the exception is to **manually** match any additional receipts to the invoice.	• **Receipt Quantity Variance** tolerance set to 10%. • 10 keyboards were ordered and invoiced. • 8 keyboards were received. **Receipt Quantity Variance** exception is triggered because quantity difference of 2 is greater than 10%.
Receipt Amount Variance	The amount for a line item on the invoice doesn't match the amount for the item on the associated receipt, and the difference is greater than the tolerance defined for the exception.	• **Receipt Amount Variance** tolerance set to 4%. • 10 keyboards were ordered, invoiced, and received. • Amount on invoice: 1,050 USD. • Amount on receipt: 1,000 USD. **Receipt Amount Variance** exception is triggered because the amount difference of 50 is greater than 4%.

Exception type	Situation that triggers the exception	Example
Matching Receipt Not Found	The invoicing solution could not find a receipt referencing the ship notice associated with a line item on the invoice. An electronic invoice typically arrives in the SAP Ariba solution before goods are received. You can configure your site to delay the start of reconciliation for a specific amount of time to avoid triggering the **Matching Receipt Not Found** exception all the time.	• 10 keyboards were ordered and invoiced. The item was marked as **Ship Notice Required** on the order, but there's no receipt for that ship notice. **Matching Receipt Not Found** exception is triggered.

These exceptions are available in the list of exceptions in the **Procure-to-Pay Manager** or **Invoice Manager** workspace in Ariba Administrator. When invoices are validated, if these exceptions arise, the SAP Ariba solution adds the **Receiving Manager** group to the approval flow of the invoice reconciliation to resolve them.

You can also define custom invoice exception types to apply only to invoices based on orders marked for goods receipt-based invoice verification. For information, see Defining invoice exception types for specific invoice source documents [page 23].

Setting goods receipt-based invoice verification for items based on supplier location

When the goods receipt-based invoice verification feature is enabled for SAP Ariba Buying and Invoicing, you can configure specific supplier locations for goods receipt-based invoice verification. When users add items to a requisition, the supplier location for the item determines whether the item goes through goods receipt-based invoice verification.

To configure supplier locations for goods receipt-based invoice verification, use one of the following supplier location import tasks:

- **Import Supplier Location Contacts**
- **Import Supplier Locations Data (Consolidated File)**

Include the column `GRBasedInvoicingPreferred` in the CSV import file. This column determines whether the items associated with the supplier location are set up for goods receipt-based invoice verification by default.

- If you use **Import Supplier Location Contacts**, include the `GRBasedInvoicingPreferred` column in the file `SupplierLocationSupplement.csv`.
- If you use **Import Supplier Locations Data (Consolidated File)**, include the `GRBasedInvoicingPreferred` column in the file `SupplierLocationConsolidated.csv`.

For the supplier locations with which you plan to use goods receipt-based invoice verification, enter `true` in the `GRBasedInvoicingPreferred` column. When users order an item from the supplier, the **Goods Receipt-based Invoice Verification** field for the item will be set to **Yes**.

When `GRBasedInvoicingPreferred` is set to `false` or is empty, the **Goods Receipt-based Invoice Verification** field on requisitions is set to **No**.

> **Note**
>
> When the **Set Goods Receipt-based invoice verification on requisition line items based on supplier location** (`Application.Procure.SetGRBasedInvoiceOnRequisitionLineItemBasedOnSupplierLocation`) parameter is set to **1**, SAP Ariba sets the **Goods Receipt-based Invoice Verification** field on requisition line items based on the value defined at the supplier location. For more information, see Set Goods Receipt-based Invoice Verification on copied requisitions based on supplier location.

Integration support for goods receipt-based invoice verification

You can integrate payment requests that reference one or more good receipts with your ERP system.

- SAP Integration Suite, managed gateway for spend management and SAP Business Network supports goods receipt-based invoice verification for sites integrated with SAP ERP or SAP S/4HANA.
- You can export payment requests using Web services, not CSV files.

Supplier location master data from SAP connected systems

The vendor master data export program exports the `GR-Based Invoice Verif` flag from the buyer's SAP ERP or SAP S/4HANA to SAP Ariba Buying and Invoicing.

For more information, see Import Supplier Data.

Purchase orders and receipts

Exported purchase orders include the field that indicates whether a line item is enabled for goods receipts-based invoicing, `GRBasedInvoice`.

To support return receipts:

- Imported and exported receipts include the ship notice ID. The field is `ASNReference` for Web services events. File events aren't supported.
- Exported receipts include the ERP receipt number of the reference receipt. The field is `ERPReceiptNumber` for Web services events, `ERPReferenceReceiptNumber` for file events.

For more information, see the following topics:

- Integration of Orders and Return Receipts for Goods Receipt-Based Invoice Verification in Procurement Data Import and Administration Guide.
- Import Supplier Data in SAP Integration Suite, Managed Gateway for Spend Management and SAP Business Network Overview Guide.

> **Note**
>
> In SAP ERP or SAP S/4HANA, the delivery note in the goods receipt contains the ship notice ID. After you export this to SAP Ariba Buying and Invoicing, the delivery note is displayed in ship notice ID. See Maintaining Parameters to Export Goods Receipts.

Including receipts in exported payment requests

You can configure your site to include matched receipts in payment requests exported through Web services by setting the parameter Include matched receipts in payment request export.

- When an invoice has been approved, invoice lines with references to order lines marked as **Goods Receipt-based Invoice Verification** are split according to the goods receipt lines that were matched to the invoice item.
- For each goods receipt item, a `ReceiptItem` element is generated.
- For SAP variants, exported payment request line items have a `GrBasedInvoice` element to show whether the supplier location is configured for goods receipt-based invoice verification.

Asynchronous payment request export through SAP Integration Suite, managed gateway for spend management and SAP Business Network using Web services

To integrate payments with SAP connected systems using goods receipts, each invoice reconciliation (IR) line in the exported payment request can be matched to only one receipt. To accommodate this requirement, configure your site to use the export event **Export Payment Requests with Receipts Asynchronously**. This export event splits IR lines to match the number of receipts. For example, if there are two receipts matched to an invoice line, the line is split to occur twice in the export, once per receipt. Each split reflects the appropriate receipt quantity. The original IR remains unchanged.

Export Payment Requests with Receipts Asynchronously works regardless of whether an invoice has goods receipt-based lines and regardless of whether goods receipt-based invoice verification is enabled.

To enable asynchronous export of payment requests with receipts:

- A member of the **Customer Administrator** group sets the parameter Web service for exporting payment requests asynchronously.
  Also remember to set Include matched receipts in payment request export.
- A member of the **Integration Admin** group configures and enables the following events in the **Integration Manager** workspace of Ariba Administrator.
  - **Export Payment Requests with Receipts Asynchronously**
  - **Import Status of Payment Requests with Receipts Asynchronously**

In the WSDL for the event **Export Payment Requests with Receipts Asynchronously**:

- At the payment level, the element `Payment_PaymentMatchedReceiptsDetForExport_Item` contains item-level details about the payment.
- Within the element `Payment_PaymentMatchedReceiptsDetForExport_Item`, the element `InvoiceReconciliationForExport` contains receipt information from the associated IR.

- Within the element `InvoiceReconciliationForExport`, the element `MatchedReceipts` contains all the receipts matched to one line.
- The `MatchedReceipts` element contains one or more `MatchedReceiptItems` elements. Each `MatchedReceiptItems` element contains only one receipt line, to accommodate SAP connected systems.

The following WSDL excerpt illustrates how lines are split when one IR line with a quantity of 90 is matched to two receipts with quantities of 20 and 70.

Sample Code

WSDL excerpt showing the split IR lines with receipt information.

```
[...]
<Payment_PaymentMatchedReceiptsDetForExport_Item>
            <item>
               <InvoiceReconciliationForExport>
                  <MatchedReceipts>
                     <item>
                        <LineNumber>1</LineNumber>
                        <MatchedReceiptItems>
                           <item>
                              <Quantity>20.00000 element contains the split IR
lines, one</Quantity>
                              <ReceiptItem>
                                 <ERPReceiptDate>2021-11-03T20:18:06Z</
ERPReceiptDate>
                                 <NumberInCollection>2</NumberInCollection>
                                 <Receipt>
                                    <ERPReceiptNumber/>
                                    <UniqueName>RC12</UniqueName>
                                 </Receipt>
                              </ReceiptItem>
                           </item>
                        </MatchedReceiptItems>
                     </item>
                     <item>
                        <LineNumber>3</LineNumber>
                        <MatchedReceiptItems>
                           <item>
                              <Quantity>70.00000</Quantity>
                              <ReceiptItem>
                                 <ERPReceiptDate>2021-11-03T20:18:23Z</
ERPReceiptDate>
                                 <NumberInCollection>2</NumberInCollection>
                                 <Receipt>
                                    <ERPReceiptNumber/>
                                    <UniqueName>RC13</UniqueName>
                                 </Receipt>
                              </ReceiptItem>
                           </item>
                        </MatchedReceiptItems>
                     </item>
                  </MatchedReceipts>
               </InvoiceReconciliationForExport>
            </item>
         </Payment_PaymentMatchedReceiptsDetForExport_Item>
[...]
```

For information about SAP Integration Suite, managed gateway for spend management and SAP Business Network support for asynchronous export of payment requests with receipts, see the documentation for feature IG-32627.

Synchronous payment request export through Web services

For synchronous export of payment requests, use the **Export Payment Requests** Web service. To include receipt information in the export, remember to set the parameter Include matched receipts in payment request export.

If an invoice line is matched to multiple receipts, the receipts are grouped together under the `MatchedReceiptItems` element.

Sample Code

WSDL excerpt showing receipt information in the `<Payment_PaymentLineItemDet_Item>` and `Payment_PaymentMatchedReceiptsDet_Item` groups of the `PaymentExportRequest` element

```
[...]
    <Payment_PaymentLineItemDet_Item>
        <item>
            <InvoiceReconciliation>
                <LineItems>
                    <item>
                        <AdjustedCostInERPPrecision>
                            <Amount>300.0000</Amount>
                        </AdjustedCostInERPPrecision>
                        <Amount>
                            <Amount>300.0000</Amount>
                        </Amount>
                        <Description>
                            <Description>Sample item</Description>
                            <UnitOfMeasure>
                                <UniqueName>EA</UniqueName>
                            </UnitOfMeasure>
                        </Description>
                        <ERPPONumber>4500061373</ERPPONumber>
                        <IsComposite>false</IsComposite>
                        <LineType>
                            <Category>1</Category>
                            <UniqueName>_NonCatalogItem</UniqueName>
                        </LineType>
                        <NumberInCollection>1</NumberInCollection>
                        <Order>
                            <UniqueName>VPO310</UniqueName>
                        </Order>
                        <OrderLineItem>
                            <ERSAllowed>false</ERSAllowed>
                            <SAPPOLineNumber>00010</SAPPOLineNumber>
                        </OrderLineItem>
                        <POLineNumber>1</POLineNumber>
                        <POLineReceivingType>2</POLineReceivingType>
                        <Quantity>220</Quantity>
                        <ReferencedInvoiceLineNumber>0</
ReferencedInvoiceLineNumber>
                        <GRBasedInvoice>true</GRBasedInvoice>
                    </item>
                </LineItems>
            </InvoiceReconciliation>
            <UniqueName>PAYT4500061373_3-169</UniqueName>
        </item>
    <Payment_PaymentAccountDet_Item>
    <Payment_PaymentGLAccDet_Item>
    <Payment_PaymentTaxExport_Item>
    <Payment_PaymentMatchedReceiptsDet_Item>
        <item>
            <InvoiceReconciliation>
                <MatchedReceipts>
                    <item>
                        <MatchedReceiptItems>
```

```
                                <item>
                                    <Quantity>16.0000</Quantity>
                                    <ReceiptItem>
                                        <ERPReceiptDate>20210901</ERPReceiptDate>
                                        <NumberInCollection>1</NumberInCollection>
                                        <Receipt>
                                            <ERPReceiptNumber>_RCT_123</
ERPReceiptNumber>
                                            <UniqueName>CRC222</UniqueName>
                                        </Receipt>
                                        <NumberInCollection>1</NumberInCollection>
                                    </ReceiptItem>
                                </item>
                                <item>
                                    <Quantity>9.0000</Quantity>
                                    <ReceiptItem>
                                        <ERPReceiptDate>20210901</ERPReceiptDate>
                                        <NumberInCollection>1</NumberInCollection>
                                        <Receipt>
                                            <ERPReceiptNumber>_RCT_456</
ERPReceiptNumber>
                                            <UniqueName>CRC226</UniqueName>
                                        </Receipt>
                                        <NumberInCollection>1</NumberInCollection>
                                    </ReceiptItem>
                                </item>
                            </MatchedReceiptItems>
                            <LineNumber>1</LineNumber>
                        </item>
                    </MatchedReceipts>
                </InvoiceReconciliation>
            </item>
        </Payment_PaymentMatchedReceiptsDet_Item>
[...]
```

Managing Payment Methods

A payment method defines how a buying organization makes a payment to a supplier. Typical payment methods include check, Automatic Clearing House (ACH) electronic transfer, wire transfer, and credit card (purchasing card).

You define payment methods for SAP Ariba Buying and Invoicing and SAP Ariba Invoice Management.

When a payment request is created, your SAP Ariba solution sets a default payment method. Users can see payment methods when viewing a payment request on the **Payment** page. Information displayed includes the payment method, terms, approval amount, due date, pay date, adjustment, amount to pay, originating bank, and receiving bank, among other fields.

Payment methods are associated with remittance locations. Each supplier location must have its own remittance location.

To see which payment methods are loaded in your site, run the **Export Payment Method Types** data export task.

About Maintaining Payment Methods in Ariba Administrator [page 68]

About Importing Payment Methods [page 70]

About Maintaining Payment Methods in Ariba Administrator

Members of the Customer Administrator and Payment Administrator groups can manage payment methods by using the Procure-to-Pay Manager workspace (SAP Ariba Buying and Invoicing) or the Payment Manager workspace (SAP Ariba Invoice Management) in Ariba Administrator.

You can view, add, modify, delete, and import payment methods.

Viewing Payment Methods

You can review existing payment methods.

Procedure

Choose **Ariba Procure-to-Pay Manager** > **Payment Methods** (SAP Ariba Buying and Invoicing) or **Payment Manager** > **Payment Methods** (SAP Ariba Invoice Management) and click **List All** to display all payment methods, or enter search criteria and then click **Search** or press the Enter key.

The following table describes the columns on the Payment Methods page:

Column	Description
ID	The unique internal identifier of the payment method.
Description	The user-visible description of the payment method.
Electronic Payment	Indicates whether this is an electronic payment method.
Preferred	Indicates whether this is the preferred payment method.
Clearance	The number of days after which a payment made with this payment method is considered cleared.
Payment Method Type	The SAP Business Network canonical payment method that corresponds to the payment method. You choose a value from the list, which includes `ach`, `creditCard`, `wire`, `check`, and `other`. These method types are mandated by the cXML protocol.

Adding or Modifying Payment Methods

You can create new payment methods and modify existing payment methods.

Procedure

1. Choose **Ariba Procure-to-Pay Manager** > **Payment Methods** (SAP Ariba Buying and Invoicing) or **Payment Manager** > **Payment Methods** (SAP Ariba Invoice Management) and click **Create New** to create a new payment method, or select the payment method you want to modify and click **Edit**.
2. Enter or modify information about the payment method on the **Add Payment Method** or **Edit Payment Method** page. The following table describes the fields on those pages:

Field	Description
Payment Method Name	The unique internal identifier of the payment method. You cannot modify this field in edit mode.
Name	The user-visible name of the payment method.
Description	The user-visible description of the payment method.
Preference Rank	The numeric rank of the payment method, which is used to select a preferred payment method when several are available. Lower numbers have higher precedence.
Clearance	The number of days after which a payment made with this payment method is considered cleared.
Electronic Payment	Whether this is an electronic payment method.
Preferred	Indicates whether this is the preferred payment method to be used if a payment method cannot be determined from the remittance location. You can only set a preferred payment method on the Payment Methods administration interface.

Field	Description
Payment Method Type	The SAP Business Network canonical payment method that corresponds to the payment method. Choose **other** for third-party financing payment methods.

3. Click **Save** to save your changes, or **Cancel** to return to the previous page without saving your changes.

Deleting Payment Methods

When a payment method is no longer valid, for example, because a credit card has expired, you can delete it.

Prerequisites

Before deleting any payment methods, Ariba recommends you export your payment method data to a CSV file by running the **Export Payment Method Types** data export task.

Procedure

1. Choose **Ariba Procure-to-Pay Manager** > **Payment Methods** (SAP Ariba Buying and Invoicing) or **Payment Manager** > **Payment Methods** (SAP Ariba Invoice Management), select the payment method, and then click **Delete**.
2. Click **OK** to delete the payment method, or **Cancel** to return to the previous page without deleting the payment method.

About Importing Payment Methods

The **Import Payment Method Types** data import task defines payment methods. This data import task is required for SAP and Generic variants. It is optional for PeopleSoft variants.

The **Import Payment Method Types** data import task reads data from the CSV file named `PaymentMethodType.csv`. You manually enter data in this CSV file for PeopleSoft and Generic variants. For SAP variants, this file can be manually maintained, or you can retrieve the data from your SAP ERP system.

> **Note**
>
> If you want to configure a default (preferred) payment method, edit the payment method you want to configure as the default payment method using the **Payment Methods** task in Ariba Administrator.

PaymentMethodType.csv: PeopleSoft Example

Following is an example of `PaymentMethodType.csv` for PeopleSoft variants:

```
UTF8
UniqueName,Name,Description,Rank,Electronic,ClearancePeriod,CanonicalName
ach,ACH,Payment via Automated Clearing House,1,true,3,ach
CC,Credit Card,Payment via Credit Card,4,true,5,creditCard
CHK,Check,Payment via check,2,false,15,check
DD,Direct debit,Payment via direct debit,4,true,5,debitCard
DR,Draft,Payment via draft,2,false,15,draft
EFT,EFT,Payment via Electronic Funds Transfer,3,true,3,wire
orbian,Orbian Financing,Payment via Orbian 3rd Party Financing,7,true,3,other
zorbian,3rd Party Financing via Zorbian,Payment via Zorbian 3rd Party
Financing,8,true,3,other
```

PaymentMethodType.csv: SAP Example

Following is an example of `PaymentMethodType.csv` for SAP variants:

```
UTF8
UniqueName,Name,Description,Rank,Electronic,ClearancePeriod,CanonicalName
A,ACH,Payment via Automated Clearing House,2,TRUE,3,ach
C,Check,Payment through check,1,FALSE,15,check
S,Single Check,Payment through check,3,FALSE,15,check
T,Bank Transfer,Payment via Bank Transfer,4,TRUE,3,wire
U,International Bank Transfer,Payment via International Bank Transfer,5,TRUE,3,wire
wire,Fed WIRE,Payment via WIRE,6,TRUE,3,wire
Orbian,Orbian Financing,Payment via Orbian 3rd Party Financing,7,TRUE,3,other
Zorbian,3rd Party Financing via Zorbian,Payment via Zorbian 3rd Party
Financing,8,TRUE,3,other
```

PaymentMethodType.csv: Generic Example

Following is an example of `PaymentMethodType.csv` for Generic variants:

```
UTF8
UniqueName,Name,Description,Rank,Electronic,ClearancePeriod,CanonicalName
ach,ACH,Payment via Automated Clearing House,1,true,3,ach
check,Check,ERP payment through check,2,false,15,check
wire,WIRE,Payment via WIRE,3,true,3,wire
creditCard,Credit Card,Payment via Credit Card,4,true,5,creditCard
3rd Party Financing via Zorbian,zorbian,Payment via Zorbian 3rd Party
Financing,8,Yes,3,other
Orbian Financing,orbian,Payment via Orbian 3rd Party Financing,7,Yes,3,other
```

About Translations for Payment Methods

You can use the `Import Payment Method Type Translations` data import task to import translations for payment method names and descriptions.

Payment Method Tanslations: PeopleSoft Example

The CSV file contains the following columns: `UniqueName`, `Name`, `Description`, `Rank`, `Electronic`, `ClearancePeriod`, `CanonicalName`, `Language`.

For example:

```
UTF8
UniqueName,Name,Description,Rank,Electronic,ClearancePeriod,CanonicalName,Language
ach,ACH,Payment via Automated Clearing House,1,true,3,ach,English
ach,ACH,Bezahlung durch ACH,1,true,3,ach,German
```

Payment Method Tanslations: SAP Example

The CSV contains the following columns: `UniqueName`, `Name`, `Description`, `Rank`, `Electronic`, `ClearancePeriod`, `CanonicalName`, `Language`.

For example:

```
UTF8
UniqueName,Name,Description,Rank,Electronic,ClearancePeriod,CanonicalName,Language
A,ACH,Payment via Automated Clearing House,2,TRUE,3,ach,English
ach,ACH,Bezahlung durch ACH,1,true,3,ach,German
```

Payment Method Tanslations: Generic Example

The CSV contains the following columns: `UniqueName`, `Name`, `Description`, `Rank`, `Electronic`, `ClearancePeriod`, `CanonicalName`, `Language`.

For example:

```
UTF8
UniqueName,Name,Description,Rank,Electronic,ClearancePeriod,CanonicalName,Language
ach,ACH,Payment via Automated Clearing House,1,true,3,ach,English
ach,ACH,Bezahlung durch ACH,1,true,3,ach,German
```

Managing Supplier Payment Information

Supplier payment information consists of the supplier's bank information, payment model and payment method used to pay the supplier, and the remittance location where payment is sent.

Remittance Locations and Payment Models [page 73]

External System Payment Model [page 75]

Data Relationships of Supplier Payment Information [page 76]

Supplier Location Matching on Invoices [page 80]

Examples of Remittance Information [page 80]

Maintaining Supplier Payment Information in Ariba Administrator [page 82]

About Importing Remit To Information [page 86]

Related Information

Managing Payment Methods [page 68]
Import Supplier Bank Payment Locations

Remittance Locations and Payment Models

SAP Business Network-enabled suppliers define remittance locations on SAP Business Network (in the **Settlement** configuration section), to indicate the address to which payment is sent.

To obtain information about suppliers (including profile settings about the payment methods that each supplier remittance location supports, bank names, and bank account IDs), the SAP Ariba solution downloads supplier profile information from SAP Business Network.

You can associate a payment model with a specific supplier or supplier location.

Validation of Remit To addresses

The remit to address on an invoice is validated against the remittance address location configured in your site. Validation behavior depends on your site configuration.

- The following parameters affect validation of invoice remit to addresses:
  - If you pay your suppliers through an external ERP system (**External System** payment model), and the source of truth for remittance information is the ERP system, SAP Ariba can disable remittance location

validation; and defining a master list of valid remittance locations is optional. This behavior is controlled by the parameter `Application.Invoicing.ValidateRemittanceLocation`, set by SAP Ariba Support.

- Members of the **Customer Administrator** group can set the parameter **Remit To on invoice reconciliations is from invoice** (`Application.Invoicing.RemittanceLocationUsingRemitToId`) in the **Intelligent Configuration Manager** workspace in Ariba Administrator.
- SAP Ariba Support can set the parameter `Application.Invoicing.SupportSupplierBankInfoOnInvoices`, which determines whether the SAP Ariba solution processes the supplier bank details in cXML invoices (in the payment method extrinsic) and whether the supplier bank details in cXML invoices from SAP Business Network are used on invoice reconciliation (IR) documents.

- SAP Ariba sites enabled in the year 2010 (Release 10s2) or later: By default, if the remit to address on an invoice received from SAP Business Network does not match any configured address, an ad hoc remit to address is created. An ad hoc address is an address that's from the supplier and is not part of your master data. Users must resolve an ad hoc remit to address before the IR can be fully reconciled and approved. Resolving an ad hoc remittance location might require adding the missing remittance location to the master data loaded in your site.
- SAP Ariba sites enabled before Release 10s2: By default, if the remit to address on the invoice is different from any of the remittance locations for the supplier location, the remit to address is defaulted to the first remittance location found for the supplier location. If no remittance locations are configured, the remit to address on the invoice is set to blank.

The Ariba Pay payment model

The Ariba Pay payment model is not available for new implementations of SAP Ariba Procurement solutions.

To prevent suppliers from adding additional remittance addresses and receiving payments to those addresses fraudulently, you must define a master list of valid remittance locations. The SAP Ariba solution uses that list to verify the remittance locations defined by the suppliers on SAP Business Network.

The remittance location for the supplier must match the remittance location configured in SAP Business Network for that supplier. If the values do not match, validation errors occur in the user interface.

If your site is configured to send payments to your suppliers through SAP Business Network, the remittance address chooser in the user interface shows only the valid addresses pulled from your master list. The SAP Ariba solution sends payments to SAP Business Network only if the remittance address selected from the chooser matches a remittance address read from SAP Business Network.

Related guides

Common Data Import and Administration for SAP Ariba Procurement Solutions

Related Information

Managing Supplier Payment Information [page 73]

Remit To on invoice reconciliations is from invoice
External System Payment Model [page 75]

External System Payment Model

Payment models determine where payments are scheduled. SAP Ariba supports the External System payment model, which means payments are scheduled and processed through an external ERP system.

The previously available Ariba Pay payment model option is no longer available to customers who are not already using it.

The payment process consists of the following phases:

- The payment scheduling phase, in which you review and finalize a payment request
- The payment settlement phase, in which the actual payment is executed

In the external system payment model, initial scheduling of payments occurs in the SAP Ariba invoicing solution, but the final payment scheduling and payment processing occurs in the external system. The following steps describe the payment process for external system payments:

1. The SAP Ariba invoicing solution initially schedules the payment based on the payment terms defined in the invoice.
2. SAP Ariba Invoice Management sends invoices approved for payment through your SAP Business Network account in cXML format to your external ERP system. SAP Ariba Buying and Invoicing send approved invoices in OK-to-Pay files to the external system.
3. The external system creates a matching invoice record and schedules the payment.
4. Remittance information is imported from the external system into your SAP Ariba solution , and the invoice reconciliation is marked **Paid** after the remittance data is received and the clearance period defined for the payment method has passed for all payment (remittance) transactions associated with the invoice.

When the external system creates a matching invoice record, it sends an update back to SAP Ariba, where the invoice reconciliation is updated with the external system invoice ID. Your external system can submit payment request updates and remittance information through the payment integration toolkit to SAP Business Network to ensure that payment-related information is synchronized between SAP Business Network, your SAP Ariba invoicing solution, and the external system.

When an IR is approved, the associated payment request status changes to **Scheduled**. If you get back payment remittance data from your external system, the system then generates a payment transaction with a status of **Paid**. If the data indicates that the invoice was not paid in full (underpayment), the system creates a new payment request document for you to track and process payment of the outstanding balance. The ID of the new payment request is the same as the original payment request, plus a suffix, such as -1.

Underpayments and overpayments are noted on the **History** tab of the IR document.

If the payment is canceled by the external system, when that cancellation data is received, a payment transaction with the status **Cancelled** is created in the SAP Ariba invoicing solution.

If your external system fails to create an associated invoice for the payment, the payment request remains in **Scheduled** status and the IR status changes to **Push Failed**. The **History** tab of the payment also notes that the push has failed.

Payments for advance payment documents in SAP Ariba Buying and Invoicing

The payment process for advance payment documents is similar to that of invoices. Approved advance payments are exported to the external system for payment processing, using the web services or file channel. The status of the advance payment document changes to **Paying** when the external system sends an acknowledgment for the advance payment document. The external system then processes the payment and sends the remittance information. The status of the advance payment document changes to **Paid** when remittance information is received for its entire amount.

Related Information

Payment Document Status Flow
Advance Payment Workflow
Advance Payment Status Flow

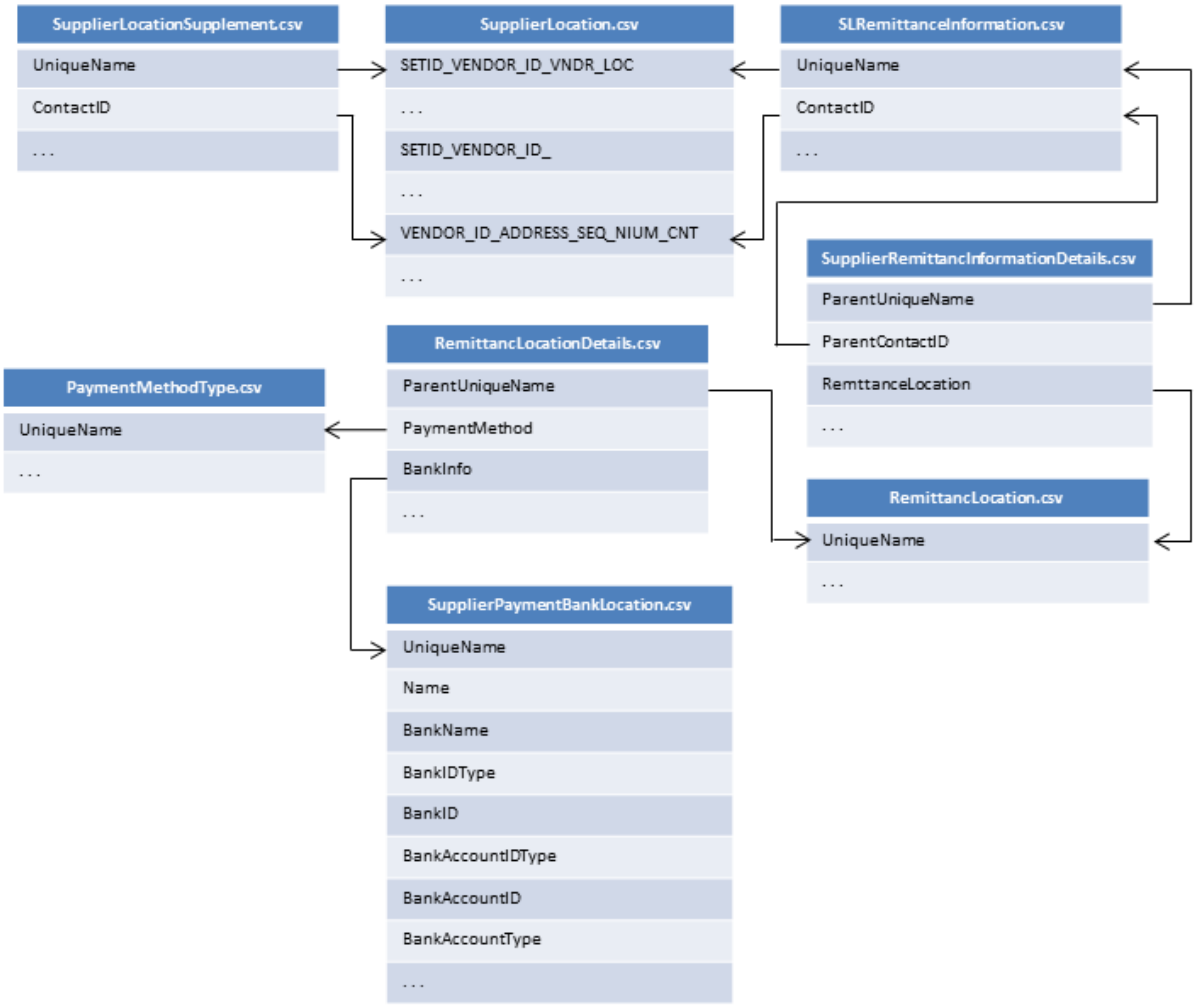
Data Relationships of Supplier Payment Information

In SAP Ariba Buying and Invoicing and SAP Ariba Invoice Management, all supplier payment information, from supplier and remittance locations to supplier bank information, is interconnected.

PeopleSoft Supplier Payment CSV File Relationships

PeopleSoft supplier payment CSV file relationships exist between `SupplierLocation.csv`, `SupplierLocationSupplement.csv`, `SLRemittanceInformation.csv`, `SupplierRemittancenformationDetails.csv`, `RemittanceLocation.csv`, `RemittanceLocationDetails.csv`, `PaymentMethodType.csv`, and `SupplierPaymentBankLocation.csv`.

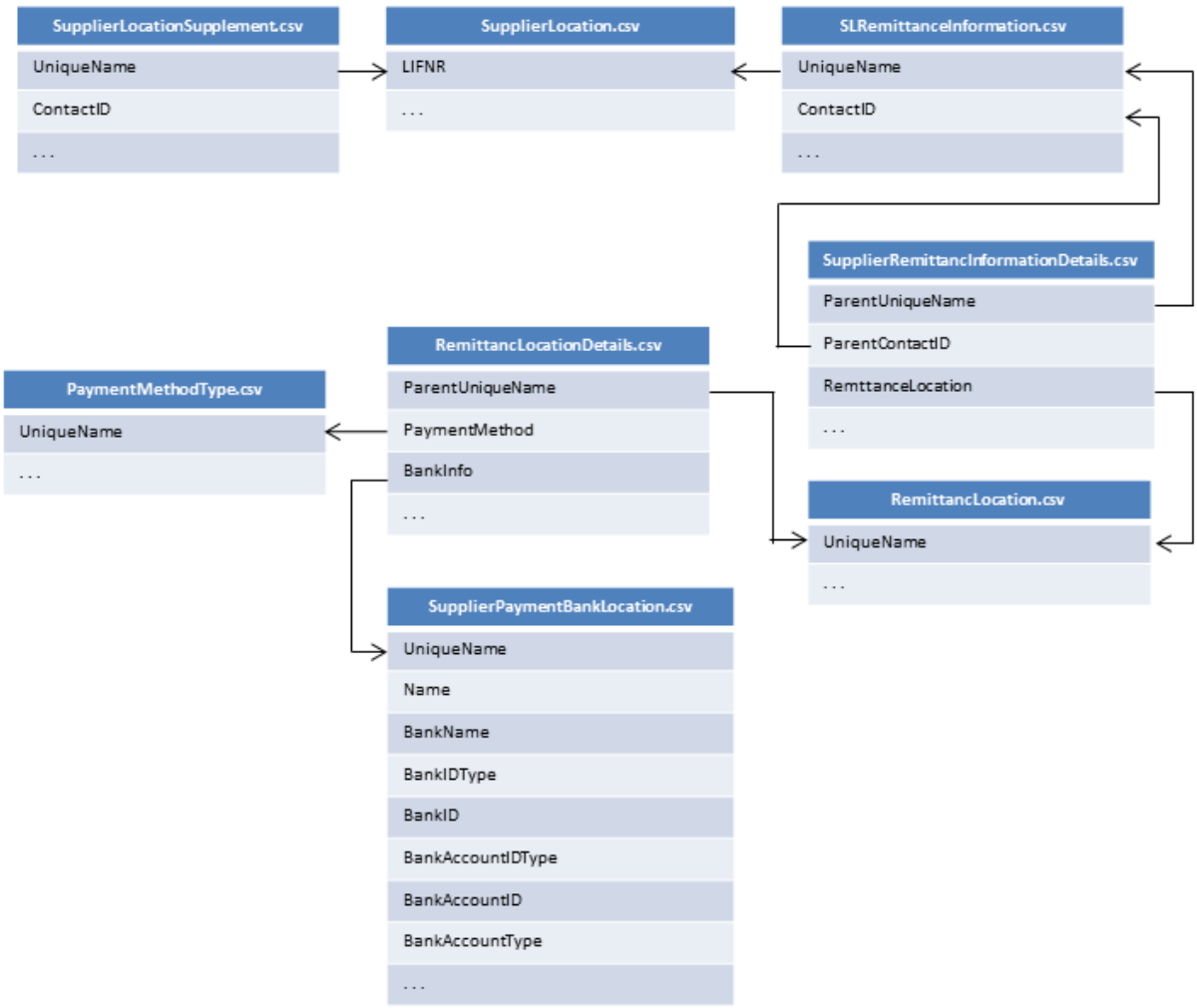
The following diagram illustrates how you associate payment information with supplier locations for PeopleSoft.



SAP Supplier Payment CSV File Relationships

SAP supplier payment CSV file relationships exist between SupplierLocation.csv, SupplierLocationSupplement.csv, SLRemittanceInformation.csv, SupplierRemittancenformationDetails.csv, RemittanceLocation.csv, RemittanceLocationDetails.csv, PaymentMethodType.csv, and SupplierPaymentBankLocation.csv.

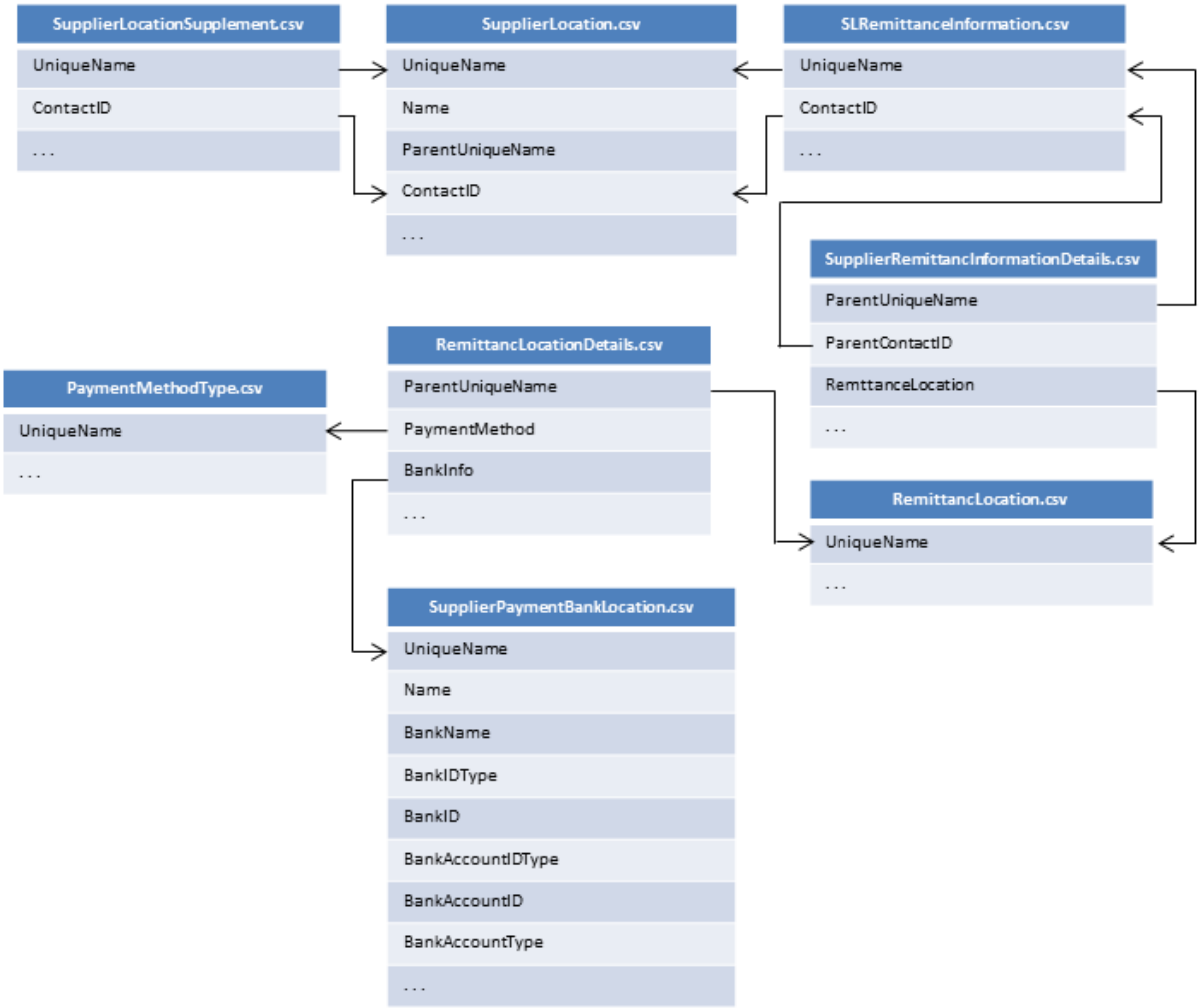
The following diagram illustrates how you associate payment information with supplier locations for SAP.



Generic Supplier Payment CSV File Relationships

Generic supplier payment CSV file relationships exist between SupplierLocation.csv, SupplierLocationSupplement.csv, SLRemittanceInformation.csv, SupplierRemittancenformationDetails.csv, RemittanceLocation.csv, RemittanceLocationDetails.csv, PaymentMethodType.csv, and SupplierPaymentBankLocation.csv.

The following diagram illustrates how you associate payment information with supplier locations for Generic.



Related Information

Import Supplier Bank Payment Locations

Supplier Location Matching on Invoices

When you receive a supplier invoice from SAP Business Network, your SAP Ariba solution checks the `From` credentials in the invoice cXML to determine which supplier sent the invoice. The `From` credential might also contain a supplier location.

For suppliers with multiple supplier locations, if the cXML invoice doesn't include a valid supplier location, the SAP Ariba solution adds the information as follows:

1. If the invoice has a remittance address, the SAP Ariba solution uses the remittance address information to determine the supplier location and the supplier location's Remit To ID. That Remit To ID becomes the Remit To on the invoice header (effectively overwriting any incorrect `addressID` on the cXML invoice).
2. If no matching remittance address is found for any supplier location associated with the supplier, the SAP Ariba solution uses a default supplier location based on zip code distances. An ad-hoc remittance location is created for the remittance address on the cXML invoice.

By default, when creating a payment request, SAP Ariba invoicing solutions use the remittance location associated with the supplier location. Your site might instead use the remittance information on the cXML invoice. This behavior is determined by the following parameters:

- `Application.Invoicing.SupportSupplierBankInfoOnInvoices` (set by SAP Ariba Support)
- `Application.Invoicing.RemittanceLocationUsingRemitToId`, **Remit To on invoice reconciliations is from invoice**

Related Information

Remit To on invoice reconciliations is from invoice

Examples of Remittance Information

These examples of the remittance information CSV files explains how `Remittance.csv`, `RemittanceLocationDetails.csv`, `SLRemittanceInformation.csv`, and `SLRemittanceInformationDetails.csv` files link together through common values.

RemittanceLocation.csv

`RemittanceLocation.csv` defines the address to which a remittance is made

The following example defines a remittance location for the Schafer Office in Hamburg called RL1.

```
UTF8
UniqueName,AddressUniqueName,Name,Lines,City,State,PostalCode,CountryUniqueName
RL1,RA1,Schafer Office,Heinerstraße 98-100,Hamburg,,27563,DE
```

RemittanceLocationDetails.csv

RemittanceLocationDetails.csv is linked by the Parent column to the remittance location defined in RemittanceLocation.csv. RemittanceLocationDetails.csv defines information about the payment method, and the supplier's payment bank location.

The value in the `Parent` column is a remittance location defined in `RemittanceLocation.csv`.

```
UTF8
Parent.UniqueName,PaymentMethod,BankInfo,RemittancePCardNumber
RL1,C,SupplierBankLocation-1,
```

The value in the `PaymentMethod` column is a payment method defined in `PaymentMethod.csv`. The value in the `BankInfo` column links a supplier payment bank location to the remittance location.

SLRemittanceInforamtion.csv

`SLRemittanceInformation.csv` associates a supplier locations with SAP Business Network payment information. The value in the `UniqueName` column is a supplier location defined in `SupplierLocation.csv`.

For example:

```
UTF8
UniqueName,ContactID,ANPayEnabled
0000000001,0000000001,NO
```

When you have different contacts within the same supplier location, you use the contact ID to differentiate between the locations.

SLRemittanceInformationDetails.csv

`SLRemittanceInformationDetails.csv` is linked by the value in the `Parent.UniqueName` column to a supplier location specified in `SLRemittanceInformation.csv`. The value in the `RemittanceLocation` column links the supplier location to a remittance location in `RemittanceLocation.csv`.

For example:

```
UTF8
Parent.UniqueName,Parent.ContactID,RemittanceLocation
0000000001,0000000001,RL1
```

When you have different contacts within the same supplier location, you use the contact ID to differentiate between the locations.

Related Information

Managing Payment Methods [page 68]

Import Supplier Bank Payment Locations

Maintaining Supplier Payment Information in Ariba Administrator

In Ariba Administrator you can view supplier payment information, add or modify a supplier remittance location, add or modify supplier payment methods, and add or modify supplier models.

Viewing Supplier Payment Information

You can examine payment information for all suppliers or you can use search filters to view payments for one or more specific suppliers.

Procedure

1. Choose **Ariba Procure-to-Pay Manager** > **Supplier Payment Information** (SAP Ariba Buying and Invoicing) or **Payment Manager** > **Supplier Payment Information** (SAP Ariba Invoice Management) and click **List All** to display all suppliers, or enter search criteria and then click **Search** or press the Enter key.
2. Click the triangle icon next to a supplier name to view city and contact information for each supplier location.
3. To view the payment model and the remittance locations for a supplier location, click **Payment Info**.
4. To view the payment methods, supplier payment bank locations, and ghosted purchasing cards (SAP Ariba Buying and Invoicing only) for a remittance location, click **Edit** to view the **Remittance Locations - Add Remittance Locations** page.
5. To view details for a particular payment method, click the links in the **Payment Method** and **Bank Information** columns.
6. Click **Done** to return to the previous page.

Adding or Modifying Supplier Remittance Locations

You can create new supplier remittance locations or modify the information about existing locations.

Procedure

1. Choose **Ariba Procure-to-Pay Manager** > **Supplier Payment Information** (SAP Ariba Buying and Invoicing) or **Payment Manager** > **Supplier Payment Information** (SAP Ariba Invoice Management), click

the triangle icon next to the supplier name, and then click **Payment Info** for the supplier location to view the **Remittance Locations** page.

2. Click **Create New** to create a remittance location, or click **Edit** to modify a remittance location.

   You can add or modify the name and address information for the remittance location. You can also add or edit the payment method associated with a remittance location. .

   > **Note**
   >
   > When a supplier uses the SAP Business Network payment model, the remittance location for the supplier must match the remittance location configured in SAP Business Network for that supplier. If the values do not match, validation errors occur in the user interface.

3. Click **OK** to save your changes, or **Cancel** to return to the previous page without saving your changes.

Related Information

Adding or Modifying Payment Methods [page 69]

Adding or Modifying Supplier Payment Methods

You can create new supplier payment methods or modify the information about existing methods.

Procedure

1. Choose **Ariba Procure-to-Pay Manager** > **Supplier Payment Information** (SAP Ariba Buying and Invoicing) or **Payment Manager** > **Supplier Payment Information** (SAP Ariba Invoice Management), click the triangle icon next to a supplier name, and then click **Payment Info** for the supplier location. The **Remittance Locations** page appears.
2. Click **Edit** next to the remittance location. The **Add Remittance Locations** page appears.
3. Click **Add** to add a payment method, or click **Edit** to modify a payment method.

   If no payment methods exist in the chooser, you have not created any payment methods.
4. To choose or modify the bank information, click **Select**.

   If no bank information exists in the chooser, you have not created any supplier bank locations.

   In SAP Ariba Buying and Invoicing, if you choose a credit card as the payment method, the **Bank Information** menu changes to **Buyer PCard Number** and you can select a ghosted purchasing card number.
5. Click **OK** to save your changes, or **Cancel** to return to the previous page without saving your changes.

Related Information

Managing Payment Methods [page 68]

Adding or Modify Supplier Payment Models for Suppliers

You can add or modify supplier payment models for suppliers. Payment models determine where payments are scheduled.

Prerequisites

If you need to modify the payment model for a supplier, ensure that all of your payments are processed completely before you modify the payment model.

Context

SAP Ariba supports the External System payment model, which means payments are scheduled and processed through an external ERP system. The previously available Ariba Pay payment model option is only available in older SAP Ariba sites.

This procedure pertains to suppliers. There's a separate task for setting payment models for supplier locations.

Procedure

1. Choose **Ariba Procure-to-Pay Manager** > **Supplier Payment Information** (SAP Ariba Buying and Invoicing) or **Payment Manager** > **Supplier Payment Information** (SAP Ariba Invoice Management) and click **Edit** next to the supplier name.

   The **Supplier Payment Information** page appears.
2. Choose the payment model in the **Payment Model** menu.
   - **Default**: The payment model defaults to the payment model configured for your site. SAP Ariba Support configures the payment model for your site during the site enablement process using the parameter `Application.Settlement.PaymentModel`.
   - **External System**: Payment requests are handled in an external system, such as an ERP system. This is the default setting of the parameter `Application.Settlement.PaymentModel`.
   - **Ariba Network**: Ariba Pay payment model. Payment requests are sent to SAP Business Network, which makes the payment. This model is only available in older SAP Ariba sites.
3. Click **Save** to save your changes.

Related Information

External System Payment Model [page 75]

Adding or Modify a Payment Model for a Supplier Location

You can add or modify supplier payment models for individual supplier locations. Payment models determine where payments are scheduled.

Prerequisites

If you need to modify the payment model for a supplier location, ensure that all of your payments are processed completely before you modify the payment model.

Context

SAP Ariba supports the External System payment model, which means payments are scheduled and processed through an external ERP system. The previously available Ariba Pay payment model option is only available in older SAP Ariba sites.

This procedure pertains to supplier locations. There's a separate task for setting payment models for suppliers.

Procedure

1. Choose **Ariba Procure-to-Pay Manager** > **Supplier Payment Information** (SAP Ariba Buying and Invoicing) or **Payment Manager** > **Supplier Payment Information** (SAP Ariba Invoice Management), click the triangle icon next to the supplier name, and then click **Payment Info** for the supplier location.

   The **Remittance Locations** page opens.
2. In the **Payment Info** section, choose or modify the payment model in the **Payment Model** menu.
   - **Default**: The payment model defaults to the payment model for the supplier. If a payment model is not associated with the supplier, the payment model configured for your site is used. SAP Ariba Support configures the payment model for your site during the site enablement process.
   - **External System**: Payment requests are handled in an external system, such as an ERP system.
   - **Ariba Network**: Ariba Pay payment model. Payment requests are sent to SAP Business Network, which makes the payment. This model is only available in older SAP Ariba sites.
3. Click **Save** to save your changes.

Related Information

External System Payment Model [page 75]

About Importing Remit To Information

Two data import tasks enable you to import remittance location information: `Import Remittance Location Information`, and `Import Supplier Location Remittance Information`.

The following table lists the data import tasks and CSV files that define remittance location information. Tasks appear in the recommended data import order.

Data Import Task	CSV Files Imported	Required
About Import Remittance Location Information [page 86]	`RemittanceLocations.csv` `RemittanceLocationDetails.csv`	Yes
About Import Supplier Location Remittance Information [page 87]	`SLRemittanceInformation.csv` `SLRemittanceInformationDetails.csv`	Yes

Import Remittance Location Information

The `Import Remittance Locations` data import task defines remittance locations.

`Import Remittance Locations` reads data from the following CSV files:

CSV File	Description
`RemittanceLocation.csv`	Defines the physical address for each remittance location.
`RemittanceLocationDetails.csv`	Defines the details of each remittance location specified in `RemittanceLocation.csv`.

Following is an example of `RemittanceLocation.csv`:

```
UTF8
UniqueName,AddressUniqueName,Name,Lines,City,State,PostalCode,CountryUniqueName
sid480-be,sid480-be,1-800-flowers.com: Belgique,Boulevard du 800 Fleur
480,Brussels,,B-1000,BE
sid480-hq,sid480-hq,1-800-flowers.com: HQ,480-800 Flowers Alley,Algonac,MI,48001,US
sid481-si,sid481-si,1-800-flowers.com: SI,481-800 Flowers Alley,Algonac,MI,48001,US
sid482-hq,sid482-hq,Barnes and Noble: HQ,496 BN Circus,Cadillac,MI,49601,US
```

Following is an example of `RemittanceLocationDetails.csv`:

```
UTF8
Parent.UniqueName,PaymentMethod,BankInfo,RemittancePCardNumber
sid480-be,ach,sid480-be-ach,
```

```
sid480-be,CHK,sid480-be-check,
sid480-be,EFT,sid480-be-wire,
sid480-hq,ach,sid480-hq-ach,
sid480-hq,CHK,sid480-hq-check,
```

Import Supplier Location Remittance Information

The `Import Supplier Location Remittance Information` data import task defines SAP Business Network payment information for each supplier location, and associates each supplier location with a remittance location.

`Import Supplier Location Remittance Information` reads data from the following CSV files:

CSV File	Description
`SLRemittanceInformation.csv`	Defines SAP Business Network payment information for supplier locations.
`SLRemittanceInformationDetails.csv`	Associates supplier locations with remittance locations. Each supplier location must have its own remittance location. You cannot associate multiple supplier locations with the same remittance location. For information on defining and importing supplier locations, see the Common Data Import and Administration for SAP Ariba Procurement Solutions.

Following is an example of `SLRemittanceInformation.csv`:

```
UTF8
UniqueName,ContactID,ANPayEnabled
9,9,NO
9,64,YES
13,13,NO
14,63,YES
15,14,NO
```

Following is an example of `SLRemittanceInformationDetails.csv`:

```
UTF8
Parent.UniqueName,Parent.ContactID,RemittanceLocation
9,9,RL6-1
9,64,RL6-2
13,13,RL10-1
14,63,RL11-1
15,14,RL10-2
```

Managing Funds Management Accounting Information

Funds Management Accounting Functionality [page 88]

Enabling the Funds Management Accounting Feature [page 88]

Prerequisites for the Funds Management Accounting Feature [page 89]

Configuring Funds Management Accounting [page 89]

Funds Management Accounting Functionality

The Funds Management accounting functionality enables you to include accounting information for Funds Management account assignment fields on approvable documents, such as requisitions and invoice reconciliation documents for encumbrance accounting and budget availability control on fund-based budgets.

This functionality is supported only on buying solutions integrated with SAP ERP.

When the Funds Management accounting functionality is enabled, you can perform the following tasks:

- Include Funds Management account assignment fields as standard Flexible Master Data (FMD) fields on approvable documents based on pre-configured FMD templates.
- Derive accounting values from the SAP ERP for Funds Management account assignment fields on requisitions based on other accounting information that is included in such documents.

When you perform procurement transactions, Funds Management processes run in the SAP ERP using the data sent by the buying solution to validate the Funds Management accounting information on the approvable documents.

Enabling the Funds Management Accounting Feature

The Funds Management accounting feature is not enabled on your site in the default configuration.

To enable this feature, have your Designated Support Contact log a service request, and an SAP Ariba Customer Support representative will contact you. Additionally, include a request for the following:

- Add visibility conditions for specific Funds Management account assignment fields that you do not want to include for your site.
- Add visibility conditions for specific approvable documents for which you do not want to include Funds Management account assignment fields.
- Configure control attributes, such as mandatory, optional, or read-only, for specific Funds Management account assignment fields.

> **Note**
>
> If visibility conditions are not configured, there is no action possible to remove the Funds Management account assignment fields that are already added to any approvable document.

Additional references

See the Purchasing Unit Administration Guide for more information about the visibility control feature.

Prerequisites for the Funds Management Accounting Feature

- Your site must be configured to support ERP Order or ERPCC as the ordering method.
- In multi-ERP configurations, enable the required functionality in all child sites.
- Funds Management accounting information in your SAP Ariba solution must be synchronized with the master data for the Funds Management application in SAP ERP.

> **Note**
>
> There is no synchronization with SAP ERP to support organic data growth for the Funds Management account assignment fields. As a result, Funds Management accounting master data in your SAP Ariba solution is not automatically updated if there are changes to the master data in SAP ERP. In addition, your SAP Ariba solution does not maintain or validate the relationship between Funds Management account assignment fields.

Configuring Funds Management Accounting

To configure Funds Management accounting functionality, you must run the related data import tasks and configure the end points for the web services used by this feature.

Procedure

1. Run the data import tasks to import Funds Management accounting information for your site. For more information, see Importing Flexible Master Data Templates for Funds Management Account Assignment Fields [page 90].
2. Configure and enable the end points for the communication channel used by the web services. For more information, see Enabling Web Services Integration Tasks for Funds Management Account Assignment Fields [page 90].

Enabling Web Services Integration Tasks for Funds Management Account Assignment Fields

You must configure the end points for the web services used by the Funds Management accounting functionality.

Context

For more information about integrating data using web services and configuring an end point, see Web Services Overview.

Procedure

1. Build the integration using an Enterprise Application Integration (EAI) tool to support web services-based integration of transactional data between the buying solution and the ERP system.

   > **Note**
   >
   > For SAP NetWeaver PI, the buying solution provides an end-to-end integration with SAP ERP using SAP NetWeaver PI.

2. Depending on your SAP Ariba solution, configure the end points for the following web services:
   - **Export Requisitions to Derive the Accounting Information for Funds Management**
   - **Export Invoice Reconciliation Documents to External System to Add Funds Management Accounting Information**
3. Enable the web services.

Importing Flexible Master Data Templates for Funds Management Account Assignment Fields

SAP Ariba provides pre-configured FMD templates to configure the Funds Management account assignment fields for use with SAP ERP-integrated sites. Depending on the requirements of your site, import the appropriate FMD templates to your buying solution.

Procedure

1. Go to **Manage** > **Core Administration** > **Site Manager** > **Data Import/Export**.
2. On the **Import** tab, perform a search for the Funds Management accounting data import tasks.

   You can use **Flex** as a keyword to perform a search for Funds Management accounting data import tasks.

3. Click **Import** for the appropriate task.
4. Click **Browse** and select the appropriate CSV file.
5. Click **Run** to enable the task.

Related Information

Sample CSV File Formats for FMD Templates [page 91]

Sample CSV File Formats for FMD Templates

The sample formats of the CSV files illustrate the column header definitions for the data import tasks used by the Funds Management feature.

The following table lists the simplified data import tasks and the corresponding CSV files for the Funds Management account assignment fields:

Data Import task	Name of CSV file
Import Flex Master Data EarmarkedFundsDocument	EarMarkExport.csv
Import Flex Master Data EarmarkedFundsLineItem	EarmarkedFundsDocumentExport.csv
Import Flex Master Data Fund	FundExport.csv
Import Flex Master Data BudgetPeriod	BudgetPeriodExport.csv
Import Flex Master Data FundsCenter	FundsCenterExport.csv
Import Flex Master Data CommitmentItem	CommitExport.csv
Import Flex Master Data FunctionalArea	FuncAreaExport.csv
Import Flex Master Data Grant	GranteeExport.csv
Import Flex Master Data FMArea	FMAreaExport.csv

Import Flex Master Data EarmarkedFundsDocument

The **Import Flex Master Data EarmarkedFundsDocument** data import task defines the **Earmarked Funds Document** field and related information and reads from the `EarMarkExport.csv` file.

Following is an example of the `EarMarkExport.csv` file:

Sample Code

```
UTF-8,,,,,,,
```

```
UniqueName,Name,CompanyCode,PurchasingUnit,cus_FMArea,cus_FMValidfromdate,cus_FMV
alidtodate,cus_Description
10,Item Nozzle,,,Atlanta,1/1/2015,1/1/2016,Nozzle for Rocket IVX
11,Item Oxidizer,,,Toronto,1/1/2015,1/1/2016,Oxidizer for Rocket IVX
12,Propulsion Pumps,,,columbia,1/1/2015,1/1/2016,Pumps for Rocket IVX
```

Import Flex Master Data EarmarkedFundsLineItem

The **Import Flex Master Data EarmarkedFundsLineItem** data import task defines the **Earmarked Funds Line Item #** field and related information and reads from the `EarmarkedFundsDocumentExport.csv` file.

Following is an example of the `EarmarkedFundsDocumentExport.csv` file:

Sample Code

```
UTF-8
UniqueName,Name,CompanyCode,PurchasingUnit,cus_DocumentItem,cus_ItemText,Amount,U
niqueName,ApproxAmountInBaseCurrency,ConversionDate,UniqueName
10,Roket Nozzle,,,10,Fund For Nozzle,2000,,,,
11,Oxidizer,,,11,Fund for Oxidizer,2001,,,,
12,Propulsion pumps,,,12,Fund for propulsion Systems,2002,,,,
```

Import Flex Master Data Fund

The **Import Flex Master Data Fund** data import task defines the **Fund** field and related information and reads from the `FundExport.csv` file.

Following is an example of the `FundExport.csv` file:

Sample Code

```
"UTF-8"
UniqueName,Name,CompanyCode,PurchasingUnit,cus_FundAREA,cus_ValidStartDate,cus_Va
lidEndDate,cus_Description
10,Space Discovery,,,Atlanta,01/01/2014,01/01/2016,Fund for Space Discovery
11,Space Station,,,Toronto,01/01/2015,01/01/2017,Fund for Space Station
12,Mars project,,,Columbia,01/01/2013,01/01/2018,Fund for Mars project
```

Import Flex Master Data BudgetPeriod

The **Import Flex Master Data BudgetPeriod** data import task defines the **Budget Period** field and related information and reads from the `BudgetPeriodExport.csv` file.

Following is an example of the `BudgetPeriodExport.csv` file:

Sample Code

```
"UTF-8"
UniqueName,Name,CompanyCode,PurchasingUnit,cus_BudgetPeriod,cus_FMValidfromdate,c
us_FMValidtodate,cus_Description
10,FiscalYear2015-2016,,,FIY15,01/01/2015,01/01/2016,FiscalYear2015-2016
11,FiscalYear2016-2017,,,FIY16,01/01/2016,01/01/2017,FiscalYear2016-2017
```

Import Flex Master Data FundsCenter

The **Import Flex Master Data FundsCenter** data import task defines the **Funds Center** field and related information and reads from the `FundsCenterExport.csv` file.

Following is an example of the `FundsCenterExport.csv` file:

Sample Code

```
"UTF-8"
UniqueName,Name,CompanyCode,PurchasingUnit,cus_FMArea,cus_FundsCenter,cus_FMValid
fromdate,cus_FMValidtodate,cus_Description
10,R&D,,,Atlanta,R&D,01/01/2015,01/01/2016,Research and Development
11,Space Exploration,,,Toronto,Space Exploration,01/01/2015,01/01/2016,Space
Exploration
12,Propulsion Systems,,,Columbia,Propulsion
Systems,01/01/2015,01/01/2016,Propulsion Systems
```

Import Flex Master Data CommitmentItem

The **Import Flex Master Data CommitmentItem** data import task defines the **Commitment** field and related information and reads from the `CommitExport.csv` file.

Following is an example of the `CommitExport.csv` file:

Sample Code

```
"UTF-8"
UniqueName,Name,CompanyCode,PurchasingUnit,cus_FMArea,cus_CommitmentItem,cus_FMVa
lidfromdate,cus_FMValidtodate,cus_SuperiorCommitmentItem,cus_Descriptionline1,cus
_Descriptionline2,cus_Descriptionline3
10,R&D,,,Atlanta,R&D,01/01/2015,01/01/2016,ReasearchOnMars,Research&Development,M
ars Project,Ongoing
11,SpaceShuttle,,,Toronto,SpaceShuttle,01/01/2015,01/01/2016,SpaceShuttle,SpaceSh
uttle,Shuttle Project,Final Stage
```

Import Flex Master Data FunctionalArea

The **Import Flex Master Data FunctionalArea** data import task defines the **Functional Area** field and related information and reads from the `FuncAreaExport.csv` file.

Following is an example of the `FuncAreaExport.csv` file:

Sample Code

```
"UTF-8"
UniqueName,Name,CompanyCode,PurchasingUnit,cus_FMValidfromdate,cus_FMValidtodate,
cus_Functionalareatext
10,Engineering,,,01/01/2015,01/01/2016,Engineering
11,Applied Science,,,01/01/2015,01/01/2016,Applied Science
12,Opearations,,,01/01/2015,01/01/2016,Opearations
```

Import Flex Master Data Grant

The **Import Flex Master Data Grant** data import task defines the **Grant** field and related information and reads from the `GranteeExport.csv` file.

Following is an example of the `GranteeExport.csv` file:

Sample Code

```
"UTF-8"
UniqueName,Name,CompanyCode,PurchasingUnit,cus_Sponsor,cus_Validfromdate,cus_Vali
dtodate,cus_GrantName,cus_GrantDescription,cus_GrantType,cus_GrantTypeDescription
10,COFAR,,,Financial Reform Dept,01/01/2015,01/01/2016,COFAR,Council on
Financial Assistance Reform,Finance,Nasa's Exisiting Fund
11,PMS,,,Payment Managemt System,01/01/2015,01/01/2016,PMS,Payment Management
System,Payment,USA Govt Allocation
```

Import Flex Master Data FMArea

The **Import Flex Master Data FMArea** data import task defines the **FM Area** field and related information and reads from the `FMAreaExport.csv` file.

Following is an example of the `FMAreaExport.csv` file:

Sample Code

```
"UTF-8"
UniqueName,Name,CompanyCode,PurchasingUnit,cus_FMAreatext,cus_FMAreaCurrency
10,Atlanta,,,Atlanta,USD
11,Toronto,,,Toronto,USD
12,Columbia,,,Columbia,USD
```

Configuring Messaging

The messaging feature allows users to initiate conversations from orders and invoices to interact with suppliers on SAP Business Network and other users in your organization to get information they require to process approvable documents.

To configure this feature, you must complete the following tasks:

1. Run the data import tasks (`Import Messaging Topics` and `Import Conversation Context Rules for Messaging`) to import conversation context rules and messaging topics.
2. Configure the messaging setup to map approvable documents with the appropriate conversation topics, users, user groups, and suppliers.

   Import Messaging Topics Task [page 95]

   Import Conversation Context Rules for Messaging Task [page 96]

   Configuring the Messaging Setup [page 104]

Related Information

Configuring the Messaging Setup [page 104]
Import Conversation Context Rules for Messaging Task [page 96]
Import Messaging Topics Task [page 95]

Import Messaging Topics Task

The **Import Messaging Topics** task imports predefined messaging topics used for conversation by the messaging service.

Users while initiating a conversation can choose one of the messaging topics to specify the subject of the conversation.

The **Import Messaging Topics** task uses the `MessagingTopicExport.csv` file. The following table describes the fields in the `MessagingTopicExport.csv` file:

Field name	Required	Type and description	Sample value
`UniqueName`	Yes	String. The unique internal identifier for a messaging topic.	QuantityIssue

Field name	Required	Type and description	Sample value
`DisplayName`	Yes	Boolean. The user-visible name for the messaging topic.	Incorrect quantity
`DefaultMessage`	No	String. The user-visible messaging topic that is displayed by default.	Incorrect quantity

Following is a sample of the `MessagingTopicExport.csv` file:

```
"UTF-8"
UniqueName,DisplayName,DefaultMessage
RequestForPartialCreditMemo,"Request for partial credit memo",
RequestForTotalCreditMemo,"Request for total credit memo",
PaymentIssue,"Payment issue",
PricingIssue,"Pricing issue",
TaxIssue,"Tax issue",
QuantityMismatch,"Quantity mismatch",
QualityIssue,"Quality issue",
MissingInvoiceData,"Missing invoice data",
ProductServiceQuestion,"Product/Service question",
RequestForSupportingDocuments,"Request for supporting documents",
Wrongquantity,"Wrong quantity",
Wrongdeliveryaddress,"Wrong delivery address",
Confirmation,"Confirmation",
PartialConfirmation,"Partial Confirmation",
Refuseddelivery,"Refused delivery",
```

Import Conversation Context Rules for Messaging Task

The **Import Conversation Context Rules for Messaging** task defines the approvable document types (for example, orders and invoices) for which messaging must to be enabled. In addition, this task maps each approvable document type with predefined topics, suppliers, user groups, and specific users who can participate in conversations related to such documents.

You can also customize the format of the subject line for invoices and orders. For example, you can set options that automatically append the order ID to a topic in the subject line for conversations related to orders.

The **Import Conversation Context Rules for Messaging** task uses the `CollaborationConfigurationDataPull.csv` file. The following table describes the fields in the `CollaborationConfigurationDataPull.csv` file:

Field name	Required	Type and description	Sample value
`UniqueName`	Yes	String. The unique internal identifier for the messaging context.	Invoice

Field name	Required	Type and description	Sample value
`Name`	Yes	Boolean. The user-visible name for the messaging context.	Invoice

Field name	Required	Type and description	Sample value
`ConfigData`	Yes	String. The configuration data specific to the messaging context. The value of this field contains five parts: 1. **\"id\"**: The unique internal identifier for the messaging context. Set it to the same value as `UniqueName`. 2. **\"contextName\"**: The user-visible name for the messaging context. Set it to the same value as `Name`. 3. **\"contacts\"**: The users or user groups who can participate in conversations related to the defined messaging contexts and topics. 4. **\"topics\"**: The topics to be mapped with the messaging context. You can map multiple topics to one messaging context. Each topic is included in a pair of braces and contains an ID, name, and description. The topics added here must be defined in the `MessagingTopics.csv` file using the **Import Messaging Topics** task. 5. **\"defaultMessage\"**: The default types of information included in a messaging topic. For example, if you add `\"defaultMessage\":\"PO#: {0},Supplier: {1}, Amount:`	`"{ \"id\": \"INVOICE\", \"contextName\": \"Invoice\", \"contacts\": [{ \"id\": \"Invoice Manager\", \"name\": \"Invoice Manager\" }], \"topics\": [{ \"id\": \"RequestForTotalCreditMemo\", \"name\": \"Request for total credit memo\", \"desc\": \"Request for total credit memo\" }, { \"id\": \"MissingInvoiceData\", \"name\": \"Missing invoice data\", \"desc\": \"Missing invoice data\" }], \"defaultMessage\" :\"Invoice# : {0},Supplier: {1}, Total Amount: {2}\" }"`

Field name	Required	Type and description	Sample value
		`{2}\"` for the messaging context `Purchase Order` in the `CollaborationConfigurationDataPull.csv` file, the messaging topics mapped to this messaging context will include three types of information: PO number, Supplier, and PO amount.	

Following is a sample of the `CollaborationConfigurationDataPull.csv` file:

```
"UTF-8"
UniqueName,Name,ConfigData
"INVOICE","Invoice","{
    \"id\": \"INVOICE\",
    \"contextName\": \"Invoice\",
    \"contacts\": [{
        \"id\": \"Invoice Manager\",
        \"name\": \"Invoice Manager\"
    },
    {
        \"id\": \"Purchasing Agent\",
        \"name\": \"Purchasing Agent\"
    }, {
        \"id\": \"Purchasing Manager\",
        \"name\": \"Purchasing Manager\"
    },
     {
        \"id\": \"Supplier\",
        \"name\": \"Supplier\"
    },{
        \"id\": \"Requester\",
        \"name\": \"Requester\"
    }],
    \"topics\": [{
        \"id\": \"RequestForTotalCreditMemo\",
        \"name\": \"Request for total credit memo\",
        \"desc\": \"Request for total credit memo\"
    },{
        \"id\": \"RequestForPartialCreditMemo\",
        \"name\": \"Request for partial credit memo\",
        \"desc\": \"Request for partial credit memo\"
    },{
        \"id\": \"Delay\",
        \"name\": \"Delay\",
        \"desc\": \"Delay\"
    },{
        \"id\": \"PaymentIssue\",
        \"name\": \"Payment Issue\",
        \"desc\": \"Payment issue\"
    },{
        \"id\": \"PricingIssue\",
        \"name\": \"Pricing issue\",
        \"desc\": \"Pricing issue\"
    }, {
        \"id\": \"QuantityMismatch\",
```

```
        \"name\": \"Quantity Mismatch\",
        \"desc\": \"Quantity Mismatch\"
    },  {
        \"id\": \"TaxIssue\",
        \"name\": \"Tax Issue\",
        \"desc\": \"Tax Issue\"
    }, {
        \"id\": \"QualityIssue\",
        \"name\": \"Quality issue\",
        \"desc\": \"Quality issue\"
    }, {
        \"id\": \"MissingInvoiceData\",
        \"name\": \"Missing invoice data\",
        \"desc\": \"Missing invoice data\"
    }, {
        \"id\": \"ProductServiceQuestion\",
        \"name\": \"Product/Service question\",
        \"desc\": \"Product/Service question\"
    },{
        \"id\": \"RequestForSupportingDocuments\",
        \"name\": \"Request for supporting documents\",
        \"desc\": \"Request for supporting documents\"
    }],
        \"defaultMessage\":\"Invoice# :{0},Supplier: {1}, Total Amount: {2}\"
}"
"INVOICE_LINE_ITEM","Invoice Line Item","{
    \"id\": \"INVOICE_LINE_ITEM\",
    \"contextName\": \"Invoice Line Item\",
    \"contacts\": [{
        \"id\": \"Purchasing Agent\",
        \"name\": \"Purchasing Agent\"
    }, {
        \"id\": \"Requester\",
        \"name\": \"Requester\"
    },{
        \"id\": \"Supplier\",
        \"name\": \"Supplier\"
    }],
    \"topics\": [{
        \"id\": \"PricingIssue\",
        \"name\": \"Pricing issue\",
        \"desc\": \"Pricing issue\"
    }, {
        \"id\": \"QuantityMismatch\",
        \"name\": \"Quantity Mismatch\",
        \"desc\": \"Quantity Mismatch\"
    },  {
        \"id\": \"TaxIssue\",
        \"name\": \"Tax Issue\",
        \"desc\": \"Tax Issue\"
    }, {
        \"id\": \"QualityIssue\",
        \"name\": \"Quality issue\",
        \"desc\": \"Quality issue\"
    }, {
        \"id\": \"MissingInvoiceData\",
        \"name\": \"Missing invoice data\",
        \"desc\": \"Missing invoice data\"
    }, {
        \"id\": \"ProductServiceQuestion\",
        \"name\": \"Product/Service question\",
        \"desc\": \"Product/Service question\"
    },{
        \"id\": \"RequestForSupportingDocuments\",
        \"name\": \"Request for supporting documents\",
        \"desc\": \"Request for supporting documents\"
    }],
```

```
     \"defaultMessage\":\"Invoice#:{0}, Supplier:{1}, Line Item: {2}, Qty: {3},
Price: {4}, Amount:{5}\"
}"
"INVOICE_LINE_ITEM_EXCEPTION","INVOICE Line Item Exception","{
    \"id\": \"INVOICE_LINE_ITEM_EXCEPTION\",
    \"contextName\": \"INVOICE Line Item Exception\",
    \"contacts\": [{
        \"id\": \"InvoiceExceptionGroup\",
        \"name\": \"Invoice Exception Group\"
    }, {
        \"id\": \"Invoice Manager\",
        \"name\": \"Invoice Manager\"
    }, {
        \"id\": \"Receiving Manager\",
        \"name\": \"Receiving Manager\"
    },
    {
        \"id\": \"Requester\",
        \"name\": \"Requester\"
    },{
        \"id\": \"Supplier\",
        \"name\": \"Supplier\"
    }],
    \"topics\": [{
        \"id\": \"PricingIssue\",
        \"name\": \"Pricing issue\",
        \"desc\": \"Pricing issue\"
    }, {
        \"id\": \"QuantityMismatch\",
        \"name\": \"Quantity Mismatch\",
        \"desc\": \"Quantity Mismatch\"
    },  {
        \"id\": \"TaxIssue\",
        \"name\": \"Tax Issue\",
        \"desc\": \"Tax Issue\"
    }, {
        \"id\": \"QualityIssue\",
        \"name\": \"Quality issue\",
        \"desc\": \"Quality issue\"
    }, {
        \"id\": \"MissingInvoiceData\",
        \"name\": \"Missing invoice data\",
        \"desc\": \"Missing invoice data\"
    }, {
        \"id\": \"ProductServiceQuestion\",
        \"name\": \"Product/Service question\",
        \"desc\": \"Product/Service question\"
    },{
        \"id\": \"RequestForSupportingDocuments\",
        \"name\": \"Request for supporting documents\",
        \"desc\": \"Request for supporting documents\"
    }],
    \"defaultMessage\":\"Invoice#: {0}, Supplier: {1}, Exception Type: {2}\"
}"
"IR","Invoice Reconciliation","{
    \"id\": \"IR\",
    \"contextName\": \"Invoice Reconciliation\",
    \"contacts\": [{
        \"id\": \"Invoice Manager\",
        \"name\": \"Invoice Manager\"
    },{
        \"id\": \"Purchasing Agent\",
        \"name\": \"Purchasing Agent\"
    }, {
        \"id\": \"Payment Manager\",
        \"name\": \"Payment Manager\"
    }, {
        \"id\": \"Requester\",
```

```
        \"name\": \"Requester\"
    }],
    \"topics\": [{
        \"id\": \"PricingIssue\",
        \"name\": \"Pricing issue\",
        \"desc\": \"Pricing issue\"
    }, {
        \"id\": \"QuantityMismatch\",
        \"name\": \"Quantity Mismatch\",
        \"desc\": \"Quantity Mismatch\"
    },  {
        \"id\": \"TaxIssue\",
        \"name\": \"Tax Issue\",
        \"desc\": \"Tax Issue\"
    }, {
        \"id\": \"QualityIssue\",
        \"name\": \"Quality issue\",
        \"desc\": \"Quality issue\"
    }, {
        \"id\": \"MissingInvoiceData\",
        \"name\": \"Missing invoice data\",
        \"desc\": \"Missing invoice data\"
    }, {
        \"id\": \"ProductServiceQuestion\",
        \"name\": \"Product/Service question\",
        \"desc\": \"Product/Service question\"
    },{
        \"id\": \"RequestForSupportingDocuments\",
        \"name\": \"Request for supporting documents\",
        \"desc\": \"Request for supporting documents\"
    }],
    \"desc\":\"Messaging configuration for Invoice Reconciliation\"
}"
"IR_LINE_ITEM","IR Line Item","{
    \"id\": \"IR_LINE_ITEM\",
    \"contextName\": \"IR Line Item\",
    \"contacts\": [{
        \"id\": \"Invoice Agent\",
        \"name\": \"Invoice Agent\"
    }, {
        \"id\": \"Purchasing Agent\",
        \"name\": \"Purchasing Agent\"
    }, {
        \"id\": \"Requester\",
        \"name\": \"Requester\"
    }],
    \"topics\": [{
        \"id\": \"PricingIssue\",
        \"name\": \"Pricing issue\",
        \"desc\": \"Pricing issue\"
    }, {
        \"id\": \"QuantityMismatch\",
        \"name\": \"Quantity Mismatch\",
        \"desc\": \"Quantity Mismatch\"
    },  {
        \"id\": \"TaxIssue\",
        \"name\": \"Tax Issue\",
        \"desc\": \"Tax Issue\"
    }, {
        \"id\": \"QualityIssue\",
        \"name\": \"Quality issue\",
        \"desc\": \"Quality issue\"
    }, {
        \"id\": \"MissingInvoiceData\",
        \"name\": \"Missing invoice data\",
        \"desc\": \"Missing invoice data\"
    }, {
        \"id\": \"ProductServiceQuestion\",
```

```
        \"name\": \"Product/Service question\",
        \"desc\": \"Product/Service question\"
    },{
        \"id\": \"RequestForSupportingDocuments\",
        \"name\": \"Request for supporting documents\",
        \"desc\": \"Request for supporting documents\"
    }],
    \"desc\":\"Messaging configuration for Invoice Reconciliation Line Item\"
}"
"PO","PO","{
    \"id\": \"PO\",
    \"contextName\": \"Purchase Order\",
    \"contacts\": [{
        \"id\": \"Purchasing  Manager\",
        \"name\": \"Purchasing Manager\"
    }, {
        \"id\": \"Supplier\",
        \"name\": \"Supplier\"
    }, {
        \"id\": \"Purchasing Agent\",
        \"name\": \"Purchasing Agent\"
    }, {
        \"id\": \"Requester\",
        \"name\": \"Requester\"
    }],
    \"topics\": [{
        \"id\": \"Confirmation\",
        \"name\": \"Confirmation\",
        \"desc\": \"Confirmation\"
    }, {
        \"id\": \"PartialConfirmation\",
        \"name\": \"Partial Confirmation\",
        \"desc\": \"Partial Confirmation\"
    }, {
        \"id\": \"RequestForSupportingDocuments\",
        \"name\": \"Request for supporting documents\",
        \"desc\": \"Request for supporting documents\"
    },{
        \"id\": \"Wrongquantity\",
        \"name\": \"Wrong quantity\",
        \"desc\": \"Wrong quantity\"
    }, {
        \"id\": \"Wrongdeliveryaddress\",
        \"name\": \"Wrong delivery address\",
        \"desc\": \"Wrong delivery address\"
    }, {
        \"id\": \"QualityIssue\",
        \"name\": \"Quality issue\",
        \"desc\": \"Quality issue\"
    }, {
        \"id\": \"Refuseddelivery\",
        \"name\": \"Refused delivery\",
        \"desc\": \"Refused delivery\"
    }],
    \"defaultMessage\":\"PO#:{0},Supplier: {1}, Amount: {2}\"
    }"
"PO_LINE_ITEM","PO Line Item","{
    \"id\": \"PO_LINE_ITEM\",
    \"contextName\": \"PO Line Item\",
    \"contacts\": [{
        \"id\": \"Purchasing  Manager\",
        \"name\": \"Purchasing Manager\"
    }, {
        \"id\": \"Supplier\",
        \"name\": \"Supplier\"
    }, {
        \"id\": \"Purchasing Agent\",
        \"name\": \"Purchasing Agent\"
```

```
    }, {
        \"id\": \"Receiving Manager\",
        \"name\": \"Receiving Manager\"
    },{
        \"id\": \"Requester\",
        \"name\": \"Requester\"
    }],
    \"topics\": [{
        \"id\": \"Confirmation\",
        \"name\": \"Confirmation\",
        \"desc\": \"Confirmation\"
    }, {
        \"id\": \"PartialConfirmation\",
        \"name\": \"Partial Confirmation\",
        \"desc\": \"Partial Confirmation\"
    }, {
        \"id\": \"RequestForSupportingDocuments\",
        \"name\": \"Request for supporting documents\",
        \"desc\": \"Request for supporting documents\"
    },{
        \"id\": \"Wrongquantity\",
        \"name\": \"Wrong quantity\",
        \"desc\": \"Wrong quantity\"
    }, {
        \"id\": \"Wrongdeliveryaddress\",
        \"name\": \"Wrong delivery address\",
        \"desc\": \"Wrong delivery address\"
    }, {
        \"id\": \"QualityIssue\",
        \"name\": \"Quality issue\",
        \"desc\": \"Quality issue\"
    }, {
        \"id\": \"Refuseddelivery\",
        \"name\": \"Refused delivery\",
        \"desc\": \"Refused delivery\"
    }],
    \"defaultMessage\": \"PO#:{0}, Supplier:{1}, Description: {2}, Qty: {3}, Amount:
{4}\"
}"
```


Configuring the Messaging Setup

You can configure the messaging setup in Ariba Administrator to enable users to participate in messaging conversations with suppliers and other users in the organization.

Procedure

1. On the dashboard, click **Manage** > **Core Administration**.
2. Depending on your SAP Ariba solution, perform one of the following actions in Ariba Administrator:
   - Click **Procure-to-Pay Manager** > **Messaging Setup**.
   - Click **Invoicing Manager** > **Messaging Setup**.

   The **Messaging Setup** page appears.

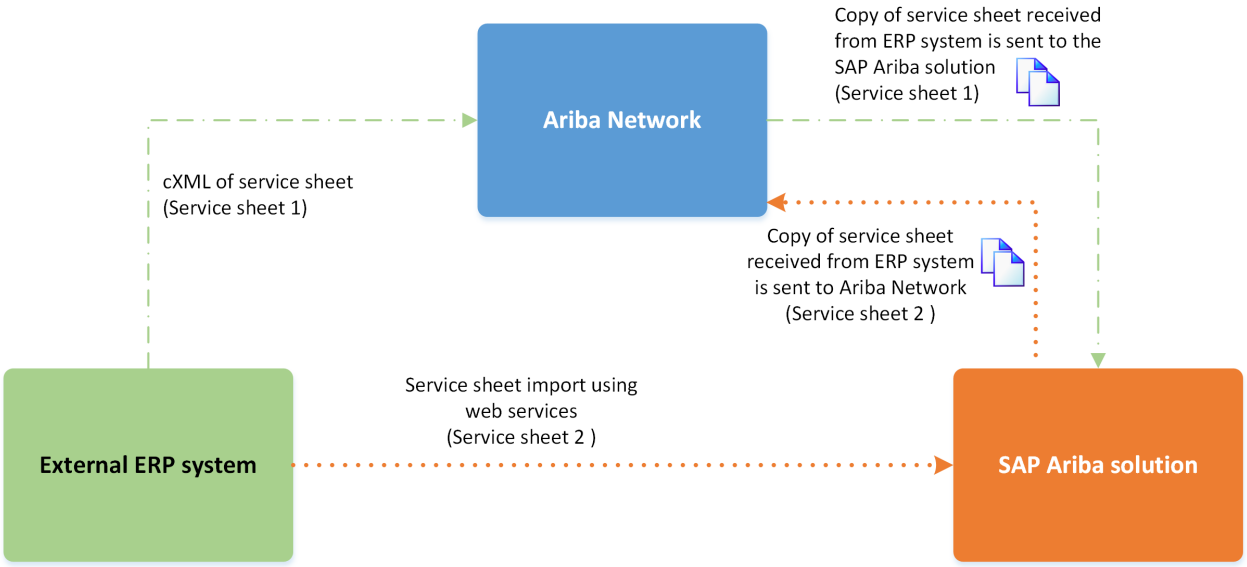
3. To configure users and user groups that can participate in messaging conversations for a document type (order or invoice), perform the following actions:
   a. Click the edit icon for the document type under the **Contacts** column.
   b. In the **Identify Contacts** window for the document type, choose the appropriate users and user groups and add them to the **Selected Contacts** list.
   c. Click **Save**.
   d. If required, repeat the steps to configure the contacts for the next document type.
4. To configure users and user groups that can participate in messaging conversations for a document type (order or invoice), perform the following actions:
   a. To configure topics that are applicable for a document type, click the edit icon for the document type under the **Topics** column.
   b. In the **Topics** window for the document type, choose the appropriate topics and add them to the **Selected Topics** list.
   c. Click **Save**.
   d. If required, repeat the steps on the **Messaging Setup** page to configure the topics for the next document type.

About Importing Service Sheets

You can import service sheets in the **Processed** state from external ERP systems using web services. In addition, you can also send processed service sheets from SAP Business Network to the SAP Ariba solution in cXML format.

> **Note**
>
> The SAP Ariba solution does not perform validations for maximum amounts on **Processed** service sheets based on the amount limits specified on the associated orders.

The following diagram illustrates the service sheet import workflow:



By default, service sheets that are successfully imported in the SAP Ariba solution display the **Processed** status. You can create PO-based invoices based on processed service sheets. After invoices based on those service sheets are fully reconciled, the status of those service sheets change to **Invoiced**.

Prerequisites

- To import service sheets using web services, your site must be configured to use web services for data integration.
- To import service sheets using cXML Posts, ensure that your site is configured to use XML for data integration.

  > **Note**
  >
  > The cXML channel applies only to SAP Business Network and not from the ERP system to your SAP Ariba solution.

- Ensure that copies of the associated purchase orders are available in your SAP Ariba solution to ensure that service sheet imports do not fail.

Importing Service Sheets Using Various Integration Channels [page 107]

Importing Service Sheets Using Various Integration Channels

You can import service sheets using Web Services or using cXML Posts.

Importing Service Sheets Using Web Services

You can import service sheets in the **Processed** state using the `ImportExternalServiceSheetPull` web services task.

If enabled for your site, you can import service sheets for SAP ERP-integrated sites through SAP Integration Suite, managed gateway for spend management and SAP Business Network, add-on for SAP ERP. See Importing Service Sheets from SAP ERP-Integrated Sites Using SAP Integration Suite, managed gateway for spend management and SAP Business Network [page 109].

Importing Service Sheets Using cXML Posts

You can import service sheets from SAP Business Network using cXML posts.

cXMLs of service sheets that have an `AribaNetwork.isECCOriginated` attribute indicate that a service sheet originated from an external system.

The following is a sample of a cXML file used to import service sheets from SAP Business Network:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE cXML SYSTEM "http://svcdev6.ariba.com/schemas/cXML/1.2.036/Fulfill.dtd">
<cXML payloadID="1519381191979-5213569330019126833@10.163.0.33"
timestamp="2018-02-23T02:19:51-08:00">
   <Header>
      <From>
         <Credential domain="NetworkID">
            <Identity>AN70000000004</Identity>
         </Credential>
         <Credential domain="PrivateID">
            <Identity>sid496___0000000100___0000000100</Identity>
         </Credential>
      </From>
      <To>
         <Credential domain="NetworkID">
            <Identity>AN71000004081</Identity>
         </Credential>
      </To>
      <Sender>
         <Credential domain="NetworkID">
            <Identity>AN70000000004</Identity>
            <SharedSecret>welcome1a</SharedSecret>
         </Credential>
```

```
          <UserAgent>Supplier</UserAgent>
      </Sender>

</Header>
<Request deploymentMode="production"><ServiceEntryRequest>
<ServiceEntryRequestHeader serviceEntryDate="2018-02-23T17:01:24+05:30"
serviceEntryID="s065">
<PartnerContact><Contact role="from"><Name xml:lang="en-US">JCN Technologies
- do NOT edit (sunita2- 1/26/12)</Name><PostalAddress><Street>An
der Welle 4</Street><Street>cathy edit 1/13/2012 test
org sync</Street><Street></Street><Street></Street><City>Paris</City><State>CA</
State><PostalCode>94089</PostalCode><Country isoCountryCode="US">United States</
Country></PostalAddress></Contact></PartnerContact>
<PartnerContact><Contact addressID="3000" role="to"><Name xml:lang="en-US">New
York</Name><PostalAddress><Street>691 Broadway</Street><Street></Street><Street></
Street><Street></Street><City>NEW YORK</City><State>NY</State><PostalCode>10001</
PostalCode><Country isoCountryCode="US">United States</Country></PostalAddress></
Contact></PartnerContact>
<PartnerContact><Contact role="requester"><Name xml:lang="en-
US">s</Name><Email>s@ariba.com</Email></Contact></PartnerContact><Extrinsic
name="AribaNetwork.isECCOriginated"/>
</ServiceEntryRequestHeader>
<ServiceEntryOrder>
<ServiceEntryOrderInfo>
<OrderReference orderID="PO14"><DocumentReference
payloadID="1520496871385.1261350791.000000007@l7IgrXp9Kz8knBIpFnwvBawCanQ=""></
DocumentReference></OrderReference>
</ServiceEntryOrderInfo>
<ServiceEntryItem quantity="1" serviceLineNumber="1" type="service"><ItemReference
lineNumber="2">
<ItemID><SupplierPartID>ADNM251</SupplierPartID></ItemID>
<Description xml:lang="en">Adapter Null Modem DB25F/F</Description>
</ItemReference><UnitOfMeasure>EA</UnitOfMeasure>
<UnitPrice><Money currency="USD">7.95</Money></UnitPrice>
<SubtotalAmount><Money currency="USD">7.95</Money></SubtotalAmount>
<Distribution>
<Accounting name="DistributionCharge">
<AccountingSegment id="0000001110"><Name xml:lang="en-US">Cost Center</Name>
<Description xml:lang="en-US">ID</Description>
</AccountingSegment>
<AccountingSegment id="0000404000"><Name xml:lang="en-US">GL Account</
Name><Description xml:lang="en-US">ID</Description></AccountingSegment>
<AccountingSegment id="100"><Name xml:lang="en-US">Percentage</Name><Description
xml:lang="en-US">Percentage</Description></AccountingSegment>
</Accounting>
<Charge><Money currency="USD">7.95</Money></Charge></Distribution>
<Extrinsic name="punchinItemFromCatalog">no</Extrinsic><Extrinsic
name="parentPOLineNumber">1</Extrinsic>
<Extrinsic name="isLineFromPO">yes</Extrinsic>
</ServiceEntryItem>
<ServiceEntryItem quantity="1" serviceLineNumber="2" type="service">
<ItemReference lineNumber="3"><ItemID><SupplierPartID>KYB52 10</SupplierPartID></
ItemID><Description xml:lang="en">Keyboard Din5M/M</Description></ItemReference>
<UnitOfMeasure>EA</UnitOfMeasure><UnitPrice><Money currency="USD">8.95</Money></
UnitPrice>
<SubtotalAmount><Money currency="USD">8.95</Money></
SubtotalAmount><Distribution><Accounting name="DistributionCharge">
<AccountingSegment id="0000001110"><Name xml:lang="en-US">Cost Center</
Name><Description xml:lang="en-US">ID</Description>
</AccountingSegment>
<AccountingSegment id="0000404000"><Name xml:lang="en-US">GL Account</Name>
<Description xml:lang="en-US">ID</Description></AccountingSegment>
<AccountingSegment id="100"><Name xml:lang="en-US">Percentage</Name><Description
xml:lang="en-US">Percentage</Description></AccountingSegment></Accounting>
<Charge><Money currency="USD">8.95</Money></Charge></Distribution>
<Extrinsic name="punchinItemFromCatalog">no</Extrinsic><Extrinsic
name="parentPOLineNumber">1</Extrinsic>
```

```
<Extrinsic name="isLineFromPO">yes</Extrinsic></ServiceEntryItem></
ServiceEntryOrder>
<ServiceEntrySummary><SubtotalAmount><Money currency="USD">16.90</Money></
SubtotalAmount></ServiceEntrySummary></ServiceEntryRequest>
    </Request>
</cXML>
```

Importing Service Sheets from SAP ERP-Integrated Sites Using SAP Integration Suite, managed gateway for spend management and SAP Business Network

Service sheets can be imported from SAP ERP-integrated sites through SAP Integration Suite, managed gateway for spend management and SAP Business Network, add-on for SAP ERP. This capability is enabled through two web services, **Import Service Sheets from External Application Asynchronously** and **Export Service Sheet Response to External System Asynchronously**.

When you successfully import approved service sheets from SAP ERP, the status of the service sheets is set to **Processed** in the SAP Ariba solution. The date of the import is set as the creation date of the service sheets.

Table 2: Web Services for Importing Service Sheets Using SAP Integration Suite, managed gateway for spend management and SAP Business Network, add-on for SAP ERP

Web Service	Description
Import Service Sheets from External Application Asynchronously	Used to import the approved service sheets from SAP ERP using SAP Integration Suite, managed gateway for spend management and SAP Business Network.
Export Service Sheet Response to External System Asynchronously	Used to send the import success or failure notification from the SAP Ariba solution to SAP ERP.

Prerequisites

- To enable this feature, have your Designated Support Contact (DSC) submit a case. SAP Support will follow up to complete the case.
- Ensure that copies of the associated purchase orders are available in your SAP Ariba solution to ensure that service sheet imports do not fail.

Restriction

The SAP Ariba solution does not perform validations for maximum amounts on **Processed** service sheets based on the amount limits specified on the associated orders.

How to Import External cXML Invoices to SAP Ariba Procurement solutions

Buyers who create cXML invoices can import the invoices directly into SAP Ariba Buying and Invoicing or SAP Ariba Invoice Management. An HTTP API lets buyers post cXML invoices to their SAP Ariba site.

Prerequisites

This task requires the following:

- A tool that lets you do an HTTP post.
- The URL for your SAP Ariba Buying and Invoicing or SAP Ariba Invoice Management site.
- Your organization's Ariba Network ID on SAP Business Network.
- The shared secret for your SAP Business Network account. To get the shared secret, contact the SAP Business Network administrator at your organization. They configured the shared secret as part of cXML setup for your SAP Business Network account.

Context

You post the cXML invoices to a URL that includes the HTTP Query parameter `ExternalCXMLRequest=true`.

The URL consists of:

- The URL for your SAP Ariba Buying and Invoicing or SAP Ariba Invoice Management site
- The import channel you're using, `cxmlchannel`
- For child sites in multi-ERP configurations, the SAP Business Network business application system ID
- Your organization's Ariba Network ID on SAP Business Network
- The query parameter `ExternalCXMLRequest=true`

For standalone sites, the syntax of the query is:

```
https://service-base/Buyer/cxmlchannel/AribaNetworkID?ExternalCXMLRequest=true
```

Examples for standalone sites:

```
https://s1.ariba.com/Buyer/cxmlchannel/AN000123456789?ExternalCXMLRequest=true
```

```
https://s1-eu.ariba.com/Buyer/cxmlchannel/AN000123456789?ExternalCXMLRequest=true
```

For child sites in multi-ERP configurations, the syntax of the query is:

```
https://service-base/Buyer/cxmlchannel/SystemID- AribaNetworkID?
ExternalCXMLRequest=true
```

Examples for child sites:

`https://s1.ariba.com/Buyer/cxmlchannel/CHILD1SAP-AN000123456789?ExternalCXMLRequest=true`

`https://s1-eu.ariba.com/Buyer/cxmlchannel/CHILD1SAP-AN000123456789?ExternalCXMLRequest=true`

Imported cXML invoices are treated the same way as cXML invoices from SAP Business Network, with the following exceptions:

- When an external invoice is imported, the SAP Ariba solution sends a CC invoice to SAP Business Network. External invoices support CC invoice quick enablement on sites that use it.
  The SAP Ariba solution also sends subsequent status update requests (SURs).
  In these ways, imported cXML invoices are treated the same way as invoices the buyer enters in the user interface.
- Because imported cXML invoices bypass SAP Business Network, they're not validated by SAP Business Network transaction rules.
- Your site can process up to 10 concurrent incoming external cXML invoices.
- You can't use this feature to import cancel invoice requests. Cancel invoice requests are applicable only for invoices sent by suppliers through SAP Business Network.
- You can't use this feature to import SAP Fieldglass or ERP CC invoices.

Procedure

1. Create properly structured cXML invoices that include the buyer shared secret and buyer Ariba Network ID in the `Sender` credential.
2. In the posting tool, specify a destination URL that identifies your SAP Ariba Buying and Invoicing or SAP Ariba Invoice Management site and ends with the HTTP query parameter `?ExternalCXMLRequest=true`.
3. For the body of the post, provide the cXML invoice.

Results

When the cXML invoice is posted to a URL, the receiving server returns a synchronous HTTP response. The body of the response contains a cXML document with a Response message that includes a status code and text describing the success or failure of the transmission.

Following is an example of a Response message:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE cXML SYSTEM "http://xml.cXML.org/schemas/cXML/1.2.019/cXML.dtd">
<cXML payloadID="9949494-189@ariba.com" xml:lang="en-US"
timestamp="2001-03-12T18:39:10-08:00">
<Response>
<Status code="200" text="OK"/>
```

The status code attribute contains a numeric value that falls within one of the following ranges:

- 200–299 indicates success.
- 400–499 indicates a permanent failure. Review the transmission to resolve the issue.
- 500–599 indicates a transient error. Retry the transmission.

The following table lists the most common status codes. For a complete list of the codes that can be returned, see the cXML Solutions Guide.

Code	Text	Description
200	OK	The server was able to execute the request or deliver it to the final recipient. The cXML Request itself generated no errors or warnings. This status does not reflect any errors or warnings that might be generated afterward by the application (in this case, SAP Ariba Buying and Invoicing). The returned Response might contain application warnings or errors. No further status updates are sent, unless an error occurs during later processing.
201	Accepted	The Request was accepted for forwarding by an intermediate hub, or was accepted by its ultimate destination and was not examined yet. The sender receives updates on the status of the Request if a mechanism to deliver the updates is available.
400	Bad Request	The Request was unacceptable to the server, although it parsed correctly.
401	Unauthorized	The credentials in the Sender element of the Request were not recognized by the server.
403	Forbidden	The user sending the Request has insufficient privileges to execute the Request.
406	Not Acceptable	The Request is unacceptable to the server, likely due to a parsing failure.
409	Conflict	The current state of the server or its internal data prevented the (update) operation request. An identical Request is unlikely to succeed in the future, but only after another operation has executed, if at all.
500	Internal Server Error	The server was unable to complete the Request.
551	Unable to Forward Request	The server was unable to forward the Request due to service provider misconfiguration. For example, an intermediate hub failed to authenticate itself to a service provider. Clients can't rectify this error, but this error might be resolved before the client retries.
560	Temporary Server Error	The server was unable to process the Request due to a temporary issue with the server. For example, a server might be down for maintenance. Try resending the Request.

cXML Guidelines for Importing External Invoices to SAP Ariba Procurement solutions [page 113]

Creating an Approval Rule for External cXML Invoices [page 150]

cXML Guidelines for Importing External Invoices to SAP Ariba Procurement solutions

There are guidelines for structuring cXML invoices for importing into SAP Ariba Buying and Invoicing or SAP Ariba Invoice Management.

cXML is an open language defined by public Document Type Definitions (DTDs). These DTDs define cXML so that it is extremely flexible. The cXML Solutions Guide provides cXML solution guidelines, recommendations, and examples to supplement the general description of cXML provided by the *cXML reference guide* at http://www.cxml.org .

cXML Requirements for Importing Invoices [page 113]

cXML Invoice Header for Importing External Invoices to SAP Ariba Procurement solutions [page 114]

cXML Invoice Request Element [page 116]

Invoice Submission Method for Importing External cXML Invoices [page 117]

Tax Information cXML [page 117]

Required and Optional Calculated Totals cXML [page 119]

Attaching the Digital Image of a Paper Invoice [page 119]

Invoice Data to Include When Importing External cXML Invoices [page 122]

Invoice-to-cXML Data Mapping [page 124]

Sample cXML for a PO-Based Invoice [page 138]

Sample cXML for a Non-PO Invoice [page 141]

Sample cXML for a Contract-Based Invoice [page 146]

Related Information

cXML solutions guide

cXML Requirements for Importing Invoices

Following are the cXML transmission requirements and high-level invoice requirements for importing invoices into SAP Ariba Buying and Invoicing or SAP Ariba Invoice Management.

- Use a posting utility that supports HTTP transactions.
- Use cXML DTD version 1.2.020 or later when defining cXML invoices.
- A cXML invoice must contain a `Header` element and a `Request` element.
  - The `Request` element for an invoice contains an `InvoiceDetailRequest`.
  - The `Header` element contains the credentials of the parties to the invoice transmission.

cXML Invoice Header for Importing External Invoices to SAP Ariba Procurement solutions

The cXML invoice header contains supplier and buyer credentials.

- A `From` credential identifies the supplier who sent the invoice.
- A `To` credentials identify the buyer to whom the supplier sent the invoice. In multi-ERP configurations, the `To` credentials also identify the child site.
- A `Sender` credential identifies the party providing the cXML invoice. In multi-ERP configurations, the `Sender` credential also identifies the child site.

Following is an example of a cXML document header for a buyer child site. The sections that follow explain the credentials in more detail.

```
<Header>
    <From>
        <Credential domain="privateid">
            <Identity>50000145224</Identity>
        </Credential>
        <Correspondent preferredLanguage="en-US">
            <Contact role="correspondent">
                <Name xml:lang="en-US">MySupplier2</Name>
                <PostalAddress>
                    <Street>1 Supplier Street</Street>
                    <City>Test Supplier City</City>
                    <State>NJ</State>
                    <PostalCode>08877</PostalCode>
                    <Country isoCountryCode="US"/>
                </PostalAddress>
            </Contact>
        </Correspondent>
    </From>
    <To>
        <Credential domain="SystemID"> <!-- Required if the buyer site has a
business application system ID. Generally used to identify child sites. -->
            <Identity>ERPCHILD1</Identity>
        </Credential>
        <Credential domain="networkid">
            <Identity>AN02003873343</Identity>
        </Credential>
    </To>
    <Sender>
        <Credential domain="networkid">
            <Identity>ERPCHILD1-AN02003873343</Identity> <!-- Buyer Ariba Network
ID preceded by the buyer site system ID, if there is one. If there were no system
ID, this would be AN02003873343. -->
            <SharedSecret>welcome3z@</SharedSecret>
        </Credential>
         <UserAgent>Buyer</UserAgent>
    </Sender>
</Header>
```

Identifying the Supplier in the From Credential

The supplier is identified by a supplier ID, which consists of a domain and a value. Examples of domains are `privateid` and `duns`. Use an ID assigned to the supplier in SAP Ariba Buying and Invoicing or SAP Ariba Invoice Management.

Following is an example of a `From` credential in which the supplier is identified using the `privateid` domain. The `Correspondent` element is optional.

```
<From>
    <Credential domain="
      privateid
     ">
        <Identity>50000145224</Identity>
    </Credential>
     <Correspondent preferredLanguage="en-US">
        <Contact role="correspondent">
            <Name xml:lang="en-US">My supplier2</Name>
            <PostalAddress>
                <Street>1 Supplier Street</Street>
                <City>Test Supplier City</City>
                <State>NJ</State>
                <PostalCode>08877</PostalCode>
                <Country isoCountryCode="US"/>
            </PostalAddress>
        </Contact>
    </Correspondent>
</From>
```

Identifying the Buyer in the To Credential

In the `To` credentials, the buying organization is identified by their Ariba Network ID on SAP Business Network.

For child sites in multi-ERP configurations, a system ID (the SAP Business Network business application system ID) identifies the specific buyer site.

Example for standalone site:

```
<To>
    <Credential domain="networkid">
        <Identity>AN02003873343</Identity>
    </Credential>
</To>
```

Example for multi-ERP child site:

```
<To>
    <Credential domain="SystemID">
        <Identity>ERPCHILD1</Identity>
    </Credential>
    <Credential domain="networkid">
        <Identity>AN02003873343</Identity>
    </Credential>
</To>
```

Identifying the Buyer in the Sender Credential

The buyer is considered to be the sender of the invoice because the buyer is the entity posting the invoice. In the `Sender` credential, the buyer is identified by the buyer's Ariba Network ID on SAP Business Network.

For child sites in multi-ERP configurations, include the business application system ID from SAP Business Network.

The `Sender` credential also has authentication information in the form of a shared secret.

The `UserAgent` element identifies the originating application.

Example for a standalone site:

```
<Sender>
    <Credential domain="networkid">
        <Identity>AN02003873343</Identity>
        <SharedSecret>welcome3z@</SharedSecret>
    </Credential>
    <UserAgent>Buyer</UserAgent>
</Sender>
```

Example for a standalone site:

```
<Sender>
    <Credential domain="networkid">
        <Identity>ERPCHILD1-AN02003873343</Identity>
        <SharedSecret>welcome3z@</SharedSecret>
    </Credential>
    <UserAgent>Buyer</UserAgent>
</Sender>
```

cXML Invoice Request Element

The `Request` element of a cXML invoice contains a single `InvoiceDetailRequest`, the content of which represents the invoice data on the paper invoice from the supplier.

A cXML invoice can represent a standard invoice, a credit memo, or a debit memo. The invoice detail can be at either the invoice header level or line level. The cXML invoice `Request` element includes three main sections:

- The invoice header element (`InvoiceDetailRequestHeader`) contains information about the parties involved in the invoice transaction (`BillTo`, `RemittTo`, `From`, `SoldTo`, and so on) as well as banking information, shipping information, and comments, extrinsic data. The header also contains information about referenced documents, such as a PO or contract. The header element can also include the `InvoiceSubmissionMethod` extrinsic element to indicate how the invoice was submitted, for example, `PaperViaICS` or `cXML`. The `Comments` element can include information about the attached image file, when sent.
- A summary element (`InvoiceDetailRequestSummary`) is required. It must contain a subtotal plus the sum of the tax and shipping amounts from the lines or the summary tax and shipping amount. A gross amount is also required.
- The invoice line element (`InvoiceDetailItem` or `InvoiceDetailServiceItem`) includes details about the lines. Use `InvoiceDetailServiceitem` to indicate service lines, which might have additional data specific to the service. Invoice lines can contain information about shipping, tax, and special handling. If included, they must also be summarized in the summary section. Each line must contain a subtotal with an optional gross amount and net amount.

This section does not cover all the details of `InvoiceDetailRequest`. It provides some general guidelines for transforming data extracted from a paper invoice into cXML.

> → Tip
>
> It's good practice for the invoice converter to go through a mapping exercise, mapping the data from their standard extract format to the cXML invoice document.

Related Information

Invoice-to-cXML Data Mapping [page 124]
cXML Invoice Requirements
cXML Document Header
Invoice Submission Method cXML
cXML Requirements for Importing Invoices [page 113]
cXML Invoice Header for Importing External Invoices to SAP Ariba Procurement solutions [page 114]
Invoice Submission Method for Importing External cXML Invoices [page 117]
Tax Information cXML [page 117]
VAT ID cXML [page 118]
Required and Optional Calculated Totals cXML [page 119]

Invoice Submission Method for Importing External cXML Invoices

The `invoiceSubmissionMethod` extrinsic indicates how the invoice was submitted. For invoices you're going to import directly into an SAP Ariba solution, set the value to `cXML`.

Following is an example of the `invoiceSubmissionMethod` extrinsic:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE cXML SYSTEM "http://xml.cxml.org/schemas/cXML/1.2.020/InvoiceDetail.dtd">
...
<InvoiceDetailRequestHeader invoiceDate="2019-08-11T01:34:42+01:00"
invoiceID="280064482"
operation="new" purpose="standard" ...>
<Extrinsic name="invoiceSubmissionMethod">cXML</Extrinsic>
</InvoiceDetailRequestHeader>
...
```

When an invoice is imported directly into SAP Ariba Buying and Invoicing or SAP Ariba Invoice Management, the **Invoice Submission Method** field on the **Summary** tab of the invoice and the `invoiceSubmissionMethod` element in the cXML file contain the value `cXML`.

Tax Information cXML

Tax information in an invoice can be included in the summary or at the line level.

To indicate that tax is included at the line level, set the following attribute under `InvoiceDetailRequestHeader/InvoiceDetailLineIndicator` (in `InvoiceDetailItem` and/or `InvoiceDetailServiceItem`):

```
isTaxInLine="yes"
```

If tax information is provided only in the `Summary` element, do not include the `isTaxInLine` attribute.

The `Tax` element in `InvoiceDetailSummary` must include the total tax for the invoice. If tax is included at the line level, the `InvoiceDetailSummary/Tax/Money` value must reflect the sum of the `Tax/Money` elements at the line level.

Tax information specified in the summary or at the line level can include details (`TaxDetail`) describing the percentage rate, category, amount, and taxable amount of the component taxes. (Additional attributes are available.) `TaxDetail`, if included, must include at least the `category` attribute and `TaxAmount` element. `TaxDetail` can occur multiple times within the `Tax` element, to provide for details of multiple tax categories. When provided, the `TaxAmount` values in `TaxDetail` must total to the value specified in the `Tax/Money` element.

Following is an example of a `Tax` element with a single `TaxDetail` element:

```
<Tax>
<Money currency="USD">13.00</Money>
<Description xml:lang="en-US">tax </Description>
<TaxDetail category="sales" percentageRate="6.5">
<TaxableAmount>
    <Money currency="USD">200.00</Money>
</TaxableAmount>
<TaxAmount>
    <Money currency="USD">13.00</Money>
</TaxAmount>
</TaxDetail>
</Tax>
```

The person constructing the cXML invoice determines the level of tax detail to include based on the requirements of the buyer and level of detail available on the paper invoice.

Related Information

Invoice-to-cXML Data Mapping [page 124]

VAT ID cXML

The supplier VAT ID must be included as an extrinsic in cXML invoices.

The following cXML excerpt shows the VAT ID of the buyer and the supplier in extrinsic elements:

```
<InvoiceDetailRequestHeader>
...
<Extrinsic name="buyerVatID">NL001605768B01</Extrinsic>
<Extrinsic name="supplierVatID">NL007369633B01</Extrinsic>
</InvoiceDetailRequestHeader>
```

Related Information

cXML Solutions Guide

Required and Optional Calculated Totals cXML

In cXML invoices, the `InvoiceDetailSummary` element must include values for the `SubTotal`, `Tax`, and `NetAmount` calculated totals. The `GrossAmount` and `DueAmount` calculated totals are optional.

- `SubTotal`: The sum of all the line level `SubTotal` elements
- `Tax`: Total tax amount for the line.
- `NetAmount`: The gross amount minus any discount applied (`InvoiceDetailDiscount`)

The following calculated totals are optional at the summary level:

- `GrossAmount`: The subtotal plus any shipping, special handling, and tax
- `DueAmount`: The net amount minus any prepaid amounts

At the line level, `SubTotal`, `Tax`, and `NetAmount` are optional. It is a good idea to provide a subtotal at the line level for clarity. The `SubTotal` at the line level is the individual line quantity multiplied by unit price.

`GrossAmount` and `DueAmount` are not applicable to the line level.

Related Information

Invoice-to-cXML Data Mapping [page 124]

Attaching the Digital Image of a Paper Invoice

cXML invoices can include an attachment that contains a digital image of the paper invoice. Attaching the digital image provides the buyer with a copy of the paper invoice, which assists in resolving issues.

In some cases, it's required to include a digital image of a paper invoice.

- With SAP Ariba ICS, service providers are required to include an attachment of the digital image. We recommend that SAP Ariba Open ICS providers include it as well.
- A digital image isn't required for buyers importing invoices into SAP Ariba Procurement solutions, regardless of whether they're using the Open ICS technology to import them.

> **Note**
>
> cXML invoices sent inside a `ProviderDataRequest` rather than as an `InvoiceDetailRequest` can't include an invoice image attachment. (ICS providers use a `ProviderDataRequest` to send an incomplete or image-only invoice.)

Attachment Guidelines

The following guidelines apply to attachments to cXML invoices:

- Attachments in one cXML document can total up to 10 MB. (This limit is for the total size of all attachments associated with the invoice, but it doesn't include the invoice itself. The invoice can be up to 4 MB or 3000 line items. The total size limit for invoices with attachments is 14 MB.)
- The process to attach the image file involves a multi-part Multipurpose Internet Mail Extensions (MIME) message to capture the transmission. The first part of the multi-part MIME message is the cXML invoice, and the second part is the attached image.
- The attached image can be encoded with any valid content-Transfer-encoding (for example, base64 or binary). The MIME part header indicates the type of encoding used.
- For an attachment to be included in a cXML invoice, one of the `Comments` elements in the cXML invoice must contain an `Attachment` element. The `Attachment` element contains a `URL` element that indicates the content ID (`Content-ID`) of the MIME part. The `URL` element specifies the content ID with a `cid:` prefix. In the MIME part header of the attachment part, the content ID is enclosed in angle brackets.
- In multi-part MIME messages, specify the content type according to the format of the image. For example:
  `Content-Type: image/tiff`
  `Content-Type: application/pdf`
- For ICS invoices: SAP Business Network stores invoice attachments for retrieval by buyers and suppliers. Users can retrieve the files by logging in to their SAP Business Network accounts. Attachments expire 18 months after SAP Business Network receives them. Expired attachments aren't available online.

The following example shows a skeleton of a multi-part MIME message for sending a cXML invoice with an attachment. Each part is set off by a boundary, which is defined in the second line.

```
POST /cXML HTTP/1.0
Content-type: multipart/related; boundary=unique boundary name;
type="text/xml"; start=<uniqueMainCID@sender.com>
--unique boundary name
Content-type: text/xml; charset="UTF-8"
Content-ID: <uniqueMainCID@sender.com>
<?xml version="1.0" encoding="UTF-8"?>
…
<Attachment>
<URL>cid:uniqueAttachmentCID@sender.com</URL>
</Attachment>
…
--unique boundary name
Content-type: image/jpeg
Content-ID: <uniqueAttachmentCID@sender.com>
…
--unique boundary name--
```

The following example of a multi-part MIME message includes a bit more detail than the preceding example. In the following example, there are two parts. The name of the boundary separating the parts is =_Part_2_7059338.1312213970506. The example includes `[...]` in place of actual invoice details.

```
------=_Part_2_7059338.1312213970506
Content-Type: text/xml; charset=UTF-8
Content-Disposition: attachment; filename=WNSMessageMultiAttach080811.xml
Content-ID:
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE cXML SYSTEM "http://xml.cXML.org/schemas/cXML/1.2.021/InvoiceDetail.dtd">
<cXML timestamp="2008-04-23T16:37:23-8:00" payloadID="Apr232008_04372337PMRTL13">
<Header>
```

```
[...]
</Header>
<Request deploymentMode="test">
<InvoiceDetailRequest>
[...]
    <InvoiceDetailRequestHeader purpose="standard" invoiceID="ARBA_Mime_2"
invoiceDate="2010-02-09T19:20:17-9:00">
        <InvoiceDetailHeaderIndicator /><InvoiceDetailLineIndicator />
[...]
        <Comments>
            <Attachment><URL>cid:TestImageURL</URL></Attachment>
        </Comments>
    </InvoiceDetailRequestHeader>
[...]
</InvoiceDetailRequest>
[...]
</Request>
</cXML>
------=_Part_2_7059338.1312213970506
Content-Type: image/tiff
Content-transfer-encoding: 7bit
Content-ID:
      <TestImageURL>
       <!--
      The angle brackets are required. Content-ID must match the cid in the URL in
the Attachment element.
     -->
Content-Disposition: attachment; filename=TestImageURL.tif
Content-Length: 23618
[...]
```

Including Attachments in PDF Format

Some buyers using SAP Ariba Procurement solutions have enabled a feature that lets them view the image side-by-side with the invoice data. For these buyers, use PDF format instead of TIF in `InvoiceDetailRequest` documents.

For buyers who haven't requested PDF attachments, and for all test buyers, send TIF files.

Following is an example of an `InvoiceDetailRequestHeader` specifying a PDF invoice attachment:

```
<InvoiceDetailRequestHeader>
<Extrinsic name="
      invoicePDF
     "><Attachment>
<URL>cid:18040725.1344960046396@cxml.org</URL>
</Attachment></Extrinsic>
</InvoiceDetailRequestHeader>
```

In multi-part MIME messages, specify the following content type when including a PDF image attachment:

```
Content-Type: application/pdf
```

Related Information

cXML Solutions Guide

Invoice Data to Include When Importing External cXML Invoices

When you create a cXML invoice from a paper invoice, you use data from the invoice header, the invoice line, the vendor master file, and Ship To master data.

Fields from the Invoice Header

Required fields are noted in parentheses:

- Language
- Invoice Type
- Invoice Number (required)
- Original PO Number, or Contract ID, or Supplier Sales Order ID for the PO (required—the invoice must contain one of these three fields)
- Invoice Date (required)
- Currency (required)
- Subtotal (required)
- Invoice Total (required)
- Tax Description (required)
- Tax Amount (required)
- Tax Rate
- Freight Amount
- Supplier (Vendor) ID
- Supplier Name
- Supplier Address
- Payment Terms
- Comments
- Requester Email Address

Fields from the Invoice Line

- Item Description
- Invoice Line Item Number (required)
- PO Line Number (required for PO based invoices)
- Supplier Part Number
- Quantity (required)

- Unit of Measure (required)
- Unit Price (required)
- Line Item Amount
- Currency
- Tax Description
- Tax Rate
- Tax Amount
- Shipping and Handling Amount (can be at the header or line level)
- Line Subtotal
- Line Total

Fields to Fill in Based on the Vendor Master File

- Remit To ID
- Remit To Name
- Remit To Address
- Remit To City
- Remit To State
- Remit To Country
- Remit To Post Code

Fields to Fill in Based on the Ship To Master Data

> **Note**
>
> Ship To information for PO-based invoices comes from the purchase order.

- Ship To Name
- Ship To Address
- Ship To City
- Ship To State
- Ship To Country
- Ship To Post Code

VAT-Related Invoice Information

VAT-related information captured includes:

- Line item tax/VAT percent

- Line item taxable amount
- Line item tax/VAT amount
- Line–tax tax amount in local currency
- Line item tax description
- Date of supply

Other legal fields can be captured for an additional charge.

Fields Added as Part of the Conversion Process

- ImageID
- CustomerID (identifies the buyer)

Invoice-to-cXML Data Mapping

Service providers must be able to map invoice data from standard extract format to the fields in the request element, header element, summary element, and invoice line fields of a cXML invoice (`InvoiceDetailRequest`).

Request Element Fields

In the request element fields table, the Field Name column represents data that appears in a paper invoice. Service providers can adjust the values to their format as needed.

Field Name	cXML Field	Description and Notes
Deployment mode	`InvoiceDetailRequest/@deploymentMode`	Optional. Possible values: • `"test"` • `"production"`

Fields in the Header Element of the Request

In the Header element fields table, the Field Name column represents data that appears in a paper invoice. Service providers can adjust the values to their format as needed.

Field Name	cXML Field (XPath)	Description and Notes
Invoice purpose, Invoice operation (Invoice type)	`InvoiceDetailRequestHeader/@purpose` `InvoiceDetailRequestHeader/@operation`	Optional. • For a standard invoice, specify `purpose="standard"` and `operation="new"`. • For a credit memo, which specifies credit to a buyer, specify `purpose="creditMemo"` and `operation="new"`. This must be a header invoice, and amounts must be negative. • For a standard invoice with line item level credit, the quantity amount is negative at the item level.
Description and notes	`InvoiceDetailRequestHeader/@invoiceID`	Required. The unique invoice number. Case sensitive. For ICS invoices, the recommended best practice is to set this value using the scan date plus the account number (or just the scan date if the paper invoice does not include an account number).
Invoice date	`InvoiceDetailRequestHeader/@invoiceDate`	Required. The invoice date on the paper invoice.
Invoice origin	`InvoiceDetailRequestHeader/@invoiceOrigin`	Optional. This field is a constant, always set to `"supplier"`.
Original PO	`InvoiceDetailOrder/InvoiceDetailOrderInfo/OrderIDInfo/orderID` Example: `<InvoiceDetailOrderInfo><OrderIDInfo orderID="2" /></InvoiceDetailOrderInfo>`	Conditionally required. The header must include either this element or `MasterAgreementIDInfo` or `SupplierOrderInfo` ID. The PO number as shown on the paper invoice. In this case, the PO number is a PO ID known to the buyer.
Supplier order	`InvoiceDetailOrder/InvoiceDetailOrderInfo/SupplierOrderInfo/orderID`	Conditionally required. The header must include either this element or `OrderIDInfo` or `MasterAgreementIDInfo`. The PO number as shown on the paper invoice. In this case, it is the supplier sales order ID of the PO.

Field Name	cXML Field (XPath)	Description and Notes
Contract	`InvoiceDetailOrder/InvoiceDetailOrderInfo/MasterAgreementIDInfo/agreementID`	Conditionally required. The header must include either this element or `OrderIDInfo` or `SupplierOrderInfo`. The buyer's ID of a corresponding master agreement.
Payment terms	`InvoiceDetailRequestHeader/PaymentTerm`	Optional. Payment terms are numeric only. Map net and discount terms if included on the paper invoice.

Indicator for header/line information	`InvoiceDetailHeaderIndicator`
	Example:
	`<InvoiceDetailHeaderIndicator></InvoiceDetailHeaderIndicator>`
	If empty:
	`<InvoiceDetailHeaderIndicator/>`
		Required.
		Required even if empty.

Invoice line indicator	`InvoiceDetailLineIndicator/@isTaxInLine`
	Example:
	`<InvoiceDetailLineIndicatorisTaxInLine="yes">`
	`</InvoiceDetailLineIndicator>`
		Optional.
		Set to `"yes"` if tax is at the line item level.

Sold to/customer contact	`InvoiceDetailRequestHeader/InvoicePartner/Contact/[@role='soldTo']`	Optional. (This information must be mapped if the supplier provides it.) Contact role type where `role="soldTo"` Buyer contact.
	`InvoiceDetailRequestHeader/InvoicePartner/Contact/[@role='soldTo' /Name` Example: `<Name xml:lang="en-US">SupplierName</Name>`	Conditionally required. Required for invoices that include soldTo information. Must include the language attribute.
	`InvoiceDetailRequestHeader/InvoicePartner/Contact[@role='soldTo'/PostalAddress/Street`	Conditionally required. Required for invoices that include `soldTo` information. Up to three occurrences of `Street` are supported.
	`InvoiceDetailRequestHeader/InvoicePartner/Contact[@role='soldTo' /PostalAddress/City`	Conditionally required. Required for invoices that include `soldTo` information.
	`InvoiceDetailRequestHeader/InvoicePartner/Contact[@role='soldTo' /PostalAddress/State`	Optional. State, province, or region.

Field Name	cXML Field (XPath)	Description and Notes
	`InvoiceDetailRequestHeader/InvoicePartner/Contact[@role='soldTo'/PostalAddress/PostalCode`	Optional.
	`InvoiceDetailRequestHeader/InvoicePartner/Contact[@role='soldTo'/Email`	Optional. The email address of the requester.
	`InvoiceDetailRequestHeader/InvoicePartner/Contact[@role='soldTo'/PostalAddress/Country` Example: `<Country isoCountryCode="US">UnitedStates</Country>`	Conditionally required. Required for invoices that include `soldTo` information. Must include the ISO country code.
Remit to	`InvoiceDetailRequestHeader/InvoicePartner/Contact/[@role='remitTo'/]`	Optional. (This information must be mapped if the supplier provides it.) Contact role type where `role="remitTo"`
Remit to ID	`InvoiceDetailRequestHeader/InvoicePartner/Contact[@role='remitTo' @addressID]`	Optional. An ID from the buyer that uniquely identifies the remittance address of the supplier. This field enables the buyer to match the remittance address in the invoice with a remittance address in their accounting system.
Remit to	`InvoiceDetailRequestHeader/InvoicePartner/Contact[@role='remitTo'/Name` Example: `<Name xml:lang="en-US">SupplierName</Name>`	Conditionally required. Required for invoices that include `remitTo` information. Must include the language attribute.
	`InvoiceDetailRequestHeader/InvoicePartner/Contact[@role='remitTo'/PostalAddress/Street`	Conditionally required. Required for invoices that include `remitTo` information. Up to three occurrences of `Street` are supported.
	`InvoiceDetailRequestHeader/InvoicePartner/Contact[@role='remitTo'/PostalAddress/City`	Conditionally required. Required for invoices that include `remitTo` information.
	`InvoiceDetailRequestHeader/InvoicePartner/Contact[@role='remitTo'/PostalAddress/State`	Optional. State, province, or region.
	`InvoiceDetailRequestHeader/InvoicePartner/Contact[@role='remitTo'/PostalAddress/PostalCode`	Optional.

Field Name	cXML Field (XPath)	Description and Notes
	`InvoiceDetailRequestHeader/InvoicePartner/Contact[@role='remitTo'/PostalAddress/Country` Example: `<Country isoCountryCode="US">UnitedStates</Country>`	Conditionally required. Required for invoices that include `remitTo` information. Must include the ISO country code.
	`IDReference@type="ach/eft/routingcode"`	Optional.
Bill to	`InvoiceDetailRequestHeader/InvoicePartner/Contact/[@role='billTo']`	Optional. (This information must be mapped if the supplier provides it.) Contact role type where `role="billTo"`.
	`InvoiceDetailRequestHeader/InvoicePartner/Contact[@role='billTo']/Name` Example: `<Name xml:lang="en-US">SupplierName</Name>`	Conditionally required. Required for invoices that include `billTo` information. Must include the language attribute.
	`InvoiceDetailRequestHeader/InvoicePartner/Contact[@role='billTo']PostalAddress/Street`	Conditionally required. Required for invoices that include `billTo` information. Up to three occurrences of `Street` are supported.
	`InvoiceDetailRequestHeader/InvoicePartner/Contact[@role='billTo']PostalAddress/City`	Conditionally required. Required for invoices that include `billTo` information.
	`InvoiceDetailRequestHeader/InvoicePartner/Contact[@role='billTo']/PostalAddress/State`	Optional. State, province, or region.
	`InvoiceDetailRequestHeader/InvoicePartner/Contact[@role='billTo']/PostalAddress/PostalCode`	Optional. Required for invoices that include `billTo` information.
	`InvoiceDetailRequestHeader/InvoicePartner/Contact[@role='billTo']/PostalAddress/Country` Example: `<Country isoCountryCode="US">UnitedStates</Country>`	Conditionally required. Required for invoices that include `billTo` information. Must include ISO country code.
From	`InvoiceDetailRequestHeader/InvoicePartner/Contact/[@role='from']`	Optional. (This information must be mapped if the supplier provides it.) Contact role type where `role="from"`. Generally, the address from which goods are sent.
	`InvoiceDetailRequestHeader/InvoicePartner/Contact[@role='from']/Name` Example: `<Name xml:lang="en-US">SupplierName</Name>`	Conditionally required. Required for invoices that include `from` information. Must include the language attribute.

Field Name	cXML Field (XPath)	Description and Notes
	`InvoiceDetailRequestHeader/InvoicePartner/Contact[@role='from']PostalAddress/Street`	Conditionally required. Required for invoices that include `from` information. Up to three occurrences of `Street` are supported.
	`InvoiceDetailRequestHeader/InvoicePartner/Contact[@role='from']PostalAddress/City`	Conditionally required. Required for invoices that include `from` information.
	`InvoiceDetailRequestHeader/InvoicePartner/Contact[@role='from']PostalAddress/State`	Optional. State, province, or region.
	`InvoiceDetailRequestHeader/InvoicePartner/Contact[@role='from']PostalAddress/PostalCode`	Optional.
	`InvoiceDetailRequestHeader/InvoicePartner/Contact[@role='from']PostalAddress/Country` Example: `<Country isoCountryCode="US">UnitedStates</Country>`	Conditionally required. Required for invoices that include `from` information. Must include the ISO country code.
Shipping Detail (Ship To)	`InvoiceDetailRequestHeader/InvoiceDetailShipping/Contact/[@role='shipTo']`	Optional. (This information must be mapped if the supplier provides it.) Contact role type where `role="shipTo"` shipTo must be sent with shipFrom. If only one is available, neither can be sent. `shipTo` can be specified at the summary level or the line level but not both. If the paper invoice includes header tax only, map `shipTo` at the summary level. If the paper invoice includes line tax information, map `shipTo` at the line level (see the Shipping Detail fields in the Invoice line fields table).
	`InvoiceDetailRequestHeader/InvoiceDetailShipping[@role='shipTo']/Name` Example: `<Name xml:lang="en-US">SupplierName</Name>`	Conditionally required. Required for invoices that include `shipTo` information. Must include the language attribute.

Field Name	cXML Field (XPath)	Description and Notes
	`InvoiceDetailRequestHeader/InvoiceDetailShipping[@role='shipTo']PostalAddress/Street`	Conditionally required. Required for invoices that include `shipTo` information. Up to three occurrences of `Street` are supported.
	`InvoiceDetailRequestHeader/InvoiceDetailShipping[@role='shipTo']PostalAddress/City`	Conditionally required. Required for invoices that include `shipTo` information.
	`InvoiceDetailRequestHeader/InvoiceDetailShipping[@role='shipTo']PostalAddress/State`	Optional. State, province, or region.
	`InvoiceDetailRequestHeader/InvoiceDetailShipping[@role='shipTo']PostalAddress/PostalCode`	Optional
	`InvoiceDetailRequestHeader/InvoiceDetailShipping[@role='shipTo']PostalAddress/Country` Example: `<Country isoCountryCode="US">UnitedStates</Country>`	Conditionally required. Required for invoices that include `shipTo` information. Must include the ISO country code.
	`InvoiceDetailRequestHeader/InvoiceDetailShipping[@role='shipTo']/Email`	Optional
Shipping Detail (Ship From)	`InvoiceDetailRequestHeader/InvoiceDetailShipping/Contact/[@role='shipFrom']`	Optional. (This information must be mapped if the supplier provides it.) Contact role type where `role="shipFrom"`. `shipFrom` is supported at the header level only. shipFrom must be sent with shipTo. If only one is available, neither can be sent.
	`InvoiceDetailRequestHeader/InvoiceDetailShipping[@role='shipFrom']/Name` Example: `<Name xml:lang="en-US">SupplierName</Name>`	Conditionally required. Required for invoices that include `shipFrom` information. Must include the language attribute.
	`InvoiceDetailRequestHeader/InvoiceDetailShipping[@role='shipFrom']PostalAddress/Street`	Conditionally required. Required for invoices that include `shipFrom` information. Up to three occurrences of `Street` are supported.

Field Name	cXML Field (XPath)	Description and Notes
	`InvoiceDetailRequestHeader/InvoiceDetailShipping[@role='shipFrom']PostalAddress/City`	Conditionally required. Required for invoices that include `shipFrom` information.
	`InvoiceDetailRequestHeader/InvoiceDetailShipping[@role='shipFrom']PostalAddress/State`	Optional. State, province, or region.
	`InvoiceDetailRequestHeader/InvoiceDetailShipping[@role='shipFrom']PostalAddress/PostalCode`	Optional.
	`InvoiceDetailRequestHeader/InvoiceDetailShipping[@role='shipFrom']PostalAddress/Country` Example: `<Country isoCountryCode="US">UnitedStates</Country`	Conditionally required. Required for invoices that include `shipFrom` information. Must include the ISO country code.
Comments	`InvoiceDetailRequestHeader/Comment`	Optional. This is for additional information from the supplier. For example, the field can provide details about the status of a back ordered item or inform the buyer that the invoice differs from the related PO in some ways.

Attachments	`InvoiceDetailRequestHeader/Comment/Attachment/URL`
	Example:
	`<Comments><Attachment><URL>cid:TestImageURL</URL></Attachment><Attachment><URL>cid:b9991630-7b0e-1000-999b-0a0e06110001</URL></Attachment></Comments>`
		Optional. (Map this field to include the scanned image in the cXML invoice so it can be sent to the ERP system.)
		With ICS, this is the location of the ID used for the scanned image attachment.

Field Name	cXML Field (XPath)	Description and Notes

Extrinsic	ICS invoices:
	`InvoiceDetailRequestHeader/Extrinsic/[name="invoiceSubmissionMethod"]="PaperViaICS"`
	Example: `<Extrinsic name="invoiceSumissionMethod">PaperViaICS</Extrinsic>`
	cXML invoices that you're importing directly into an SAP Ariba solution:
	`InvoiceDetailRequestHeader/Extrinsic/[name="invoiceSubmissionMethod"]="cXML"`
	Example: `<Extrinsic name="invoiceSumissionMethod">cXML</Extrinsic>`
		Optional but recommended.
		Indicates how the invoice was submitted.
		ICS note: It is a best practice to set `invoiceSubmissionMethod` to `PaperViaICS` to prevent the invoice from being digitally signed.

Extrinsic (legacy)	`InvoiceDetailRequestHeader/Extrinsic/[name="convertedInvoiceData"]` Example: `<Extrinsic name="convertedInvoiceData"/>`	Legacy, replaced by `invoiceSubmissionMethod`. Indicates conversion method. This field is supported for legacy interfaces. Including the `convertedInvoiceData` extrinsic prevents the invoice from being digitally signed.

Fields in the Summary Element of the Request

In the Summary element fields table, the Field Name column represents data that appears in a paper invoice. Service providers can adjust the values to their format as needed.

Field Name	cXML Field	Description and Notes
Subtotal	`InvoiceDetailSummary/SubtotalAmount/Money` Example: `<Moneycurrency="USD">3.05</Money>`	Required. Sum of line item quantities multiplied by unit price. Must include the currency type.

Field Name	cXML Field	Description and Notes

Tax	`InvoiceDetailSummary/Tax/TaxableAmount/Money`
	
	Following is a complete Tax example. See the following rows for details on tax-related fields.
	
	`<Tax><TaxDetail category="sales" percentageRate="7.75">`
	`<TaxableAmount>`
	`<Money currency="USD">6.95</Money>`
	`</TaxableAmount>`
	`<TaxAmount>`
	`<Money alternateAmount="0.54" alternateCurrency="USD" currency="USD">0.54`
	`</Money>`
	`</TaxAmount>`
	`<Description xml:lang="en-US"></Description>`
	`</TaxDetail></Tax>`
		Required.
		
		Amount. Must include the currency type:
		`<Moneycurrency="USD">3.05</Money>`

Tax	`InvoiceDetailSummary/Tax/Description`	Required. Description of the tax.
Tax detail	`InvoiceDetailSummary/Tax/TaxDetail`	Required. Detail on the tax amount. Multiple occurrences.
Tax category	`InvoiceDetailSummary/Tax/TaxDetail/@category`	Conditionally required. Required for invoices that include `TaxDetail` information. The type of tax. Sample values are" • `"sales"` • `"gst"` • `"vat"`
Tax summary rate	`InvoiceDetailSummary/Tax/TaxDetail/@percentageRate`	Optional. Percentage rate of the tax. Percentage rate multiplied by taxable amount equals tax amount.
Tax summary taxable amount	`InvoiceDetailSummary/Tax/TaxDetail/TaxableAmount`	Conditionally required.

Field Name	cXML Field	Description and Notes
Tax summary location	`InvoiceDetailSummary/Tax/TaxDetail/TaxLocation`	Optional. The location or description of the tax authority.
Tax summary description	`InvoiceDetailSummary/Tax/TaxDetail/Description` Example: `<Description xml:lang="en-US">tax desc</Description>`	Optional. Map if the information is provided. Must include language attribute.
Total tax	`InvoiceDetailSummary/Tax/TaxDetail/TaxAmount`	Conditionally mandatory. Total tax.

Total special handling	`InvoiceDetailSummary/SpecialHandlingAmount/Money`
	and
	`/Description`
		Optional.
		Total special handling charge. Supplier can optionally add a `Description` element to explain the charge.

Total shipping	`InvoiceDetailSummary/ShippingAmount/Money`	Optional. Total shipping charge.
Total gross amount	`InvoiceDetailSummary/GrossAmount/Money`	Optional. Sum of subtotal, taxes, special handling charges, and shipping charges, before discounts.

Total discount amount	`InvoiceDetailSummary/InvoiceDetailDiscount/@percentageRate`
	and
	`/Money`
		Optional.
		The total discount or penalty applied in the invoice.

Total net amount	`InvoiceDetailSummary/NetAmount/Money`	Required. Total gross amount minus discounts.
Amount due	`InvoiceDetailSummary/DueAmount/Money`	Optional. Total amount due and payable.

Invoice Line Fields

In the Invoice Line fields table, the Field Name column represents data that appears in a paper invoice. Service providers can adjust the values to their format as needed.

Field Name	cXML Field	Description and Notes
Invoice line item number	`InvoiceDetailItem/[@invoiceLineNumber]`	Conditionally required. Invoice line number. ICS note: Required for `invoiceDetailItem` method.
Invoice line number	`InvoiceDetailItem/ InvoiceDetailItemReference/lineNumber`	Conditionally required. The line reference number for each line item, corresponding to the PO line number. ICS note: Required when referencing a PO.
Part number	`InvoiceDetailItem/ InvoiceDetailItemReference/ItemID/ SupplierPartID`	Conditionally required. The service provider must include this field in cXML invoice if the paper invoice has the supplier part ID. The complete part number. With non-PO based invoices, or if the buyer allows changes to the part number for PO based invoices, the part number is displayed in editable format.
Description	`InvoiceDetailItem/ InvoiceDetailItemReference/Description`	Optional. A text description of the line item. Must include a language attribute.
Quantity	`InvoiceDetailItem/quantity`	Conditionally required. The quantity of the item, as a float. Fractional amounts are supported with weights and lengths (for example, 6 or 6.43). ICS note: Required for `invoiceDetailItem` method.
Unit of measure	`InvoiceDetailItem/UnitOfMeasure`	Conditionally required. UN/CEFACT unit of measure code, for example, `"EA"`

Field Name	cXML Field	Description and Notes
Unit price	`InvoiceDetailItem/UnitPrice/Money`	Conditionally required. The price per unit of measure. This is a negative number if the invoice is intended to provide credit to the buyer. ICS note: Required for `invoiceDetailItem` method.
Subtotal	`InvoiceDetailItem/SubtotalAmount/Money` Example: `<SubtotalAmount><Money currency="USD">36.85</Money>`	Optional. A calculated field showing the quantity multiplied by unit price. SAP Business Network calculates the subtotal for online invoice entry. For general service items, the subtotal can be entered directly rather than calculated.
Gross amount	`InvoiceDetailItem/GrossAmount/Money` Example: `<GrossAmount><Money currency="USD">800.66</Money></GrossAmount>`	Optional. (Map if the paper invoice includes the information.)
Net amount	`InvoiceDetailItem/NetAmount/Money` Example: `<NetAmount><Money currency="USD">836.85</Money></NetAmount>`	Optional. (Map if the paper invoice includes the information.)
Tax details	`InvoiceDetailItem/Tax`	Optional. Tax can be specified at the header level or the line level.
Line–tax rate	`InvoiceDetailItem/Tax/TaxDetail/percentageRate`	Optional. The tax rate as a numeric value, with a decimal point if appropriate.
Line–tax taxable amount	`InvoiceDetailItem/Tax/TaxDetail/TaxableAmount`	Optional. The amount that is taxable.
Line–tax tax amount	`InvoiceDetailItem/Tax/TaxDetail/TaxAmount`	Optional. The calculated tax.

Field Name	cXML Field	Description and Notes
Line–tax location	`InvoiceDetailItem/Tax/TaxDetail/TaxLocation`	Optional. The name of the locale in which this tax is to be paid.
Line-tax description	`InvoiceDetailItem/Tax/TaxDetail/Description`	Optional. A brief description of the tax. Sample values: • `"California sales tax"` • `"exempt"` • `“zero tax”` May be required for some countries.
Special handling description	`InvoiceDetailItem/ InvoiceDetailLineSpecialHandling/ Description`	Optional.
Special handing amount	`InvoiceDetailItem/ InvoiceDetailLineSpecialHandling/Money`	Optional.
Shipping detail	`InvoiceDetailItem/ InvoiceDetailLineShipping/ InvoiceDetailShipping/shippingDate` Example: `<InvoiceDetailShipping shippingDate="2008-04-11T00:00:00">`	Optional.
Line-level Ship To	`InvoiceDetailItem/ InvoiceDetailLineShipping/ InvoiceDetailShipping/Contact/ [@role='shipTo']`	Optional. (Map if the supplier includes item-level shipping information.) Contact role type where `role = "shipTo"`
	`InvoiceDetailItem/ InvoiceDetailLineShipping/ InvoiceDetailShipping/[@role='shipTo']/Name` Example: `<Name xml:lang="en-US">Supplier Name</Name>`	Optional. Must include the language attribute.
	`InvoiceDetailItem/ InvoiceDetailLineShipping/ InvoiceDetailShipping/[@role='shipTo']/ PostalAddress/Street`	Optional. Up to three occurrences of `Street` are supported.

Field Name	cXML Field	Description and Notes
	`InvoiceDetailItem/ InvoiceDetailLineShipping/ InvoiceDetailShipping/[@role='shipTo']/ PostalAddress/State`	Optional.
	`InvoiceDetailItem/ InvoiceDetailLineShipping/ InvoiceDetailShipping/[@role='shipTo']/ PostalAddress/PostalCode`	Optional.
	`InvoiceDetailItem/ InvoiceDetailLineShipping/ InvoiceDetailShipping/ [@role='shipTo']PostalAddress/Country` Example: `<Country isoCountryCode="US">United States</Country>`	Optional. Must include the ISO country code.
	`InvoiceDetailItem/ InvoiceDetailLineShipping/ InvoiceDetailShipping/[@role='shipTo']/ Email`	Optional.

Sample cXML for a PO-Based Invoice

Here's a cXML invoice example for buyers importing external PO-based cXML invoices to SAP Ariba Buying and Invoicing or SAP Ariba Invoice Management.

> **Note**
>
> This example is for a standalone buyer site. In multi-ERP configurations, include both the Ariba Network ID and the business application system ID from SAP Business Network in the `To` and `Sender` credentials, as in `ERPCHILD1-AN71000003970`.

Sample Code

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE cXML SYSTEM "http://xml.cxml.org/schemas/cXML/1.2.035/
InvoiceDetail.dtd">
<cXML payloadID="15077886091-903867768609504416.109.111.112"
timestamp="2017-10-09T04:18:06-07:00" version="1.2.035">
<Header>
    <From>
        <Credential domain="privateid">
            <Identity>1234567</Identity>
        </Credential>
    </From>
    <To>
        <Credential domain="NetworkID">
            <Identity>AN71000003983</Identity>
        </Credential>
```

```
    </To>
    <Sender>
        <Credential domain="NetworkID">
            <Identity>AN71000003983</Identity>
            <SharedSecret>welcome1a</SharedSecret>
        </Credential>
        <UserAgent>Buyer</UserAgent>
    </Sender>
</Header>
<Request deploymentMode="test">
<InvoiceDetailRequest>
<InvoiceDetailRequestHeader invoiceDate="2017-10-09T16:48:05+05:30"
invoiceID="INV31" invoiceOrigin="supplier" operation="new" purpose="standard">
    <InvoiceDetailHeaderIndicator/>
    <InvoiceDetailLineIndicator isAccountingInLine="yes" isShippingInLine="yes"
isSpecialHandlingInLine="yes"/>
    <InvoicePartner>
        <Contact addressID="2050000348" role="remitTo">
            <Name xml:lang="en-US">Any Name Inc</Name>
            <PostalAddress>
                <Street>300 West Avenue Street</Street><Street/><Street/
><Street/>
                <City>Syracuse</City>
                <State>NY</State>
                <PostalCode>13202</PostalCode>
                <Country isoCountryCode="US">United States</Country>
            </PostalAddress>
        </Contact>
    </InvoicePartner>
    <InvoicePartner>
        <Contact addressID="2026" role="billTo">
            <Name xml:lang="en">Algun nombre</Name>
            <PostalAddress>
                <Street>PASEO DE TAMARINDOS</Street>
                <City>MEXICO CITY</City>
                <State>DIF</State>
                <PostalCode>05110</PostalCode>
                <Country isoCountryCode="MX"/>
            </PostalAddress>
        </Contact>
    </InvoicePartner>
    <InvoicePartner>
        <Contact role="from">
            <Name xml:lang="en-US">Any Name Inc</Name>
            <PostalAddress>
                <Street>300 West Avenue Street</Street><Street/><Street/
><Street/>
                <City>Syracuse</City>
                <State>NY</State>
                <PostalCode>13202</PostalCode>
                <Country isoCountryCode="US">United States</Country>
            </PostalAddress>
        </Contact>
    </InvoicePartner>
    <InvoicePartner>
        <Contact role="billFrom">
            <Name xml:lang="en-US">Any Name Inc</Name>
            <PostalAddress>
                <Street>300 West Avenue Street</Street><Street/><Street/
><Street/>
                <City>Syracuse</City>
                <State>NY</State>
                <PostalCode>13202</PostalCode>
                <Country isoCountryCode="US">United States</Country>
            </PostalAddress>
        </Contact>
    </InvoicePartner>
    <InvoicePartner>
```

```
        <Contact role="soldTo">
            <Name xml:lang="en-US">Another Contact Name</Name>
            <PostalAddress>
                <Street>19001 S WESTERN AVE</Street><Street/><Street/><Street/>
                <City>CACITY</City>
                <State>CA</State>
                <PostalCode>90501</PostalCode>
                <Country isoCountryCode="US">United States</Country>
            </PostalAddress>
        </Contact>
    </InvoicePartner>
    <PaymentTerm payInNumberOfDays="1"/>
    <Extrinsic name="invoiceSourceDocument">PurchaseOrder</Extrinsic>
    <Extrinsic name="invoiceSubmissionMethod">cXML</Extrinsic>
    <Extrinsic
name="invoicePDF"><Attachment><URL>cid:1361870635.1507547895432@cxml.org</URL></
Attachment></Extrinsic>
</InvoiceDetailRequestHeader>
<InvoiceDetailOrder>
    <InvoiceDetailOrderInfo>
        <OrderReference orderID="4501731">
                <DocumentReference
payloadID="1507547809043_711429418@10.59.35.140"></DocumentReference>
        </OrderReference>
    </InvoiceDetailOrderInfo>
    <InvoiceDetailItem invoiceLineNumber="1" quantity="10">
        <UnitOfMeasure>EA</UnitOfMeasure>
        <UnitPrice><Money currency="USD">10.00</Money></UnitPrice>
        <PriceBasisQuantity conversionFactor="1" quantity="1">
            <UnitOfMeasure>EA</UnitOfMeasure>
            <Description xml:lang="en-US"></Description>
        </PriceBasisQuantity>
        <InvoiceDetailItemReference lineNumber="1">
            <ItemID>
                <SupplierPartID>Non Catalog Item</SupplierPartID>
            </ItemID>
            <Description xml:lang="en">Green thing</Description>
        </InvoiceDetailItemReference>
        <SubtotalAmount><Money currency="USD">100.00</Money></SubtotalAmount>
        <InvoiceDetailLineSpecialHandling>
            <Description xml:lang="en-US"></Description><Money
currency="USD">40.00</Money>
        </InvoiceDetailLineSpecialHandling>
        <InvoiceDetailLineShipping>
            <InvoiceDetailShipping>
                <Contact role="shipFrom">
                    <Name xml:lang="en-US">Any Name Inc</Name>
                    <PostalAddress>
                        <Street>300 West Avenue Street</Street><Street></
Street><Street></Street><Street></Street>
                        <City>Syracuse</City>
                        <State>NY</State>
                        <PostalCode>13202</PostalCode>
                        <Country isoCountryCode="US">United States</Country>
                    </PostalAddress>
                </Contact>
                <Contact addressID="21AH" role="shipTo">
                    <Name xml:lang="EN">MFG MX</Name>
                    <PostalAddress><DeliverTo></DeliverTo><DeliverTo></DeliverTo>
                        <Street>PASEO DE TAMARINDOS, BOSQUES 642-H</Street>
                        <City>MEXICO CITY</City>
                        <PostalCode>05110</PostalCode>
                        <Country isoCountryCode="MX">></Country>
                    </PostalAddress>
                </Contact>
            </InvoiceDetailShipping>
            <Money currency="USD">50.00</Money>
        </InvoiceDetailLineShipping>
```

```
        <GrossAmount><Money currency="USD">190.00</Money></GrossAmount>
        <NetAmount><Money currency="USD">190.00</Money></NetAmount>
        <Distribution>
            <Accounting name="DistributionCharge">
                <AccountingSegment id="6171720002">
                    <Name xml:lang="EN">GeneralLedger</Name>
                    <Description xml:lang="EN"></Description>
                </AccountingSegment>
                <AccountingSegment id="50F000">
                    <Name xml:lang="EN">CostCenter</Name>
                    <Description xml:lang="EN"></Description>
                </AccountingSegment>
                <AccountingSegment id="100.00">
                    <Name xml:lang="EN">Percentage</Name>
                    <Description xml:lang="EN"></Description>
                </AccountingSegment>
            </Accounting>
            <Charge><Money currency="USD">100.00</Money></Charge>
        </Distribution>
        <Extrinsic name="punchinItemFromCatalog">no</Extrinsic>
    </InvoiceDetailItem>
</InvoiceDetailOrder>
    <InvoiceDetailSummary>
        <SubtotalAmount><Money currency="USD">100.00</Money></SubtotalAmount>
        <Tax>
            <Money currency="USD">0</Money>
            <Description xml:lang="en-US"></Description>
        </Tax>
        <SpecialHandlingAmount><Money currency="USD">40.00</Money></
SpecialHandlingAmount>
        <ShippingAmount><Money currency="USD">50.00</Money></ShippingAmount>
        <GrossAmount><Money currency="USD">190.00</Money></GrossAmount>
        <NetAmount><Money currency="USD">190.00</Money></NetAmount>
        <DueAmount><Money currency="USD">190.00</Money></DueAmount>
    </InvoiceDetailSummary>
</InvoiceDetailRequest>
</Request>
</cXML>
```

Sample cXML for a Non-PO Invoice

Here's a cXML invoice example for buyers importing external non-PO cXML invoices to SAP Ariba Buying and Invoicing or SAP Ariba Invoice Management.

Note

This example is for a standalone buyer site. In multi-ERP configurations, include both the Ariba Network ID and the business application system ID from SAP Business Network in the `To` and `Sender` credentials, as in `ERPCHILD1-AN71000003970`.

Sample Code

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE cXML SYSTEM "http://svcdev6.ariba.com/schemas/cXML/1.2.042/
InvoiceDetail.dtd">
<cXML payloadID="1566287213497-3191938426863.1.124"
timestamp="2019-08-20T00:46:53-07:00">
    <Header>
        <From>
```

```
            <Credential domain="privateid">
                <Identity>1234567</Identity>
            </Credential>
        </From>
        <To>
            <Credential domain="NetworkId">
                <Identity>AN71000003970</Identity>
            </Credential>
        </To>
        <Sender>
            <Credential domain="NetworkId">
                <Identity>AN71000003970</Identity>
                <SharedSecret>welcome1a</SharedSecret>
            </Credential>
            <UserAgent>Buyer</UserAgent>
        </Sender>
    </Header>
    <Request>
        <InvoiceDetailRequest>
            <InvoiceDetailRequestHeader
                invoiceOrigin="supplier" invoiceDate="2019-11-06T13:47:50-08:00"
                operation="new" purpose="standard"
                invoiceID="_88">
                <InvoiceDetailHeaderIndicator />
                <InvoiceDetailLineIndicator
                    isAccountingInLine="yes" />
                <InvoicePartner>
                    <Contact addressID="US006" role="billTo">
                        <Name xml:lang="en">New York Sales</Name>
                        <PostalAddress name="default">
                            <Street>89 That Street</Street>
                            <City>City</City>
                            <State>NY</State>
                            <PostalCode>18037</PostalCode>
                            <Country isoCountryCode="US">United States</Country>
                        </PostalAddress>
                        <Phone name="work">
                            <TelephoneNumber>
                                <CountryCode isoCountryCode="US">1</CountryCode>
                                <AreaOrCityCode>212</AreaOrCityCode>
                                <Number>5551212</Number>
                            </TelephoneNumber>
                        </Phone>
                        <Fax name="work">
                            <TelephoneNumber>
                                <CountryCode isoCountryCode="US" />
                                <AreaOrCityCode />
                                <Number />
                            </TelephoneNumber>
                        </Fax>
                    </Contact>
                </InvoicePartner>
                <InvoicePartner>
                    <Contact addressID="RA11-1" role="remitTo">
                        <Name xml:lang="en">Any Name</Name>
                        <PostalAddress name="default">
                            <Street>89 That Street</Street>
                            <City>City</City>
                            <State>TX</State>
                            <PostalCode>78701</PostalCode>
                            <Country isoCountryCode="US">United States</Country>
                        </PostalAddress>
                    </Contact>
                </InvoicePartner>
                <InvoicePartner>
                    <Contact addressID="RA11-1" role="soldTo">
                        <Name xml:lang="en">Any Name</Name>
                        <PostalAddress name="default">
```

```
                <Street>89 That Street</Street>
                <City>City</City>
                <State>TX</State>
                <PostalCode>78701</PostalCode>
                <Country isoCountryCode="US">United States</Country>
            </PostalAddress>
            <Email></Email>
        </Contact>
    </InvoicePartner>
    <InvoiceDetailShipping>
        <Contact addressID="144" role="shipFrom">
            <Name xml:lang="de">jeder Name</Name>
            <PostalAddress name="Standardwert">
                <Street>Schelmenwasenstrasse 00</Street>
                <City>Stuttgart</City>
                <PostalCode>7000</PostalCode>
                <Country isoCountryCode="DE">Germany</Country>
            </PostalAddress>
            <Phone name="work">
                <TelephoneNumber>
                    <CountryCode isoCountryCode="DE">49</CountryCode>
                    <AreaOrCityCode>1</AreaOrCityCode>
                    <Number>1720000000</Number>
                </TelephoneNumber>
            </Phone>
        </Contact>
        <Contact addressID="US0806" role="shipTo">
            <Name xml:lang="en">New York buyer</Name>
            <PostalAddress name="default">
                <Street>89 That Street</Street>
                <City>City</City>
                <State>NY</State>
                <PostalCode>10937</PostalCode>
                <Country isoCountryCode="US">United States</Country>
            </PostalAddress>
            <Phone name="work">
                <TelephoneNumber>
                    <CountryCode isoCountryCode="US">1</CountryCode>
                    <AreaOrCityCode>212</AreaOrCityCode>
                    <Number>5551212</Number>
                </TelephoneNumber>
            </Phone>
            <Fax name="work">
                <TelephoneNumber>
                    <CountryCode isoCountryCode="US" />
                    <AreaOrCityCode />
                    <Number />
                </TelephoneNumber>
            </Fax>
        </Contact>
    </InvoiceDetailShipping>
    <PaymentTerm payInNumberOfDays="45"></PaymentTerm>
    <PaymentTerm payInNumberOfDays="30">
        <Discount>
            <DiscountPercent percent="2" />
        </Discount>
    </PaymentTerm>
    <PaymentTerm payInNumberOfDays="20">
        <Discount>
            <DiscountPercent percent="3" />
        </Discount>
    </PaymentTerm>
    <Comments>
        <Attachment>
            <URL>cid:testpdf</URL>
        </Attachment>
    </Comments>
```

```
                <Extrinsic name="invoiceSourceDocument">NoOrderInformation</
Extrinsic>
                <Extrinsic name="invoiceSubmissionMethod">cXML</Extrinsic>
                <Extrinsic name="PurchasingUnit">US100</Extrinsic>
                <Extrinsic name="Custom(H)" />
            </InvoiceDetailRequestHeader>
            <InvoiceDetailOrder>
                <InvoiceDetailOrderInfo>
                    <SupplierOrderInfo orderID="Unknown" />
                </InvoiceDetailOrderInfo>
                <InvoiceDetailItem quantity="1"
                    invoiceLineNumber="1">
                    <UnitOfMeasure>EA</UnitOfMeasure>
                    <UnitPrice>
                        <Money alternateCurrency="" alternateAmount=""
currency="USD">5.95</Money>
                    </UnitPrice>
                    <InvoiceDetailItemReference lineNumber="0">
                        <ItemID>
                            <SupplierPartID> NEWHEIRARCHYabc156</SupplierPartID>
                        </ItemID>
                        <Description xml:lang="en-US">Adapter Din5M/MD6F
Keyboard</Description>
                        <ManufacturerPartID>abc156</ManufacturerPartID>
                        <ManufacturerName>All Cables</ManufacturerName>
                        <Country isoCountryCode="DE">Germany</Country>
                    </InvoiceDetailItemReference>
                    <SubtotalAmount>
                        <Money alternateCurrency="" alternateAmount=""
currency="USD">5.95</Money>
                    </SubtotalAmount>
                    <GrossAmount>
                        <Money alternateCurrency="" alternateAmount=""
currency="USD">5.95</Money>
                    </GrossAmount>
                    <NetAmount>
                        <Money alternateCurrency="" alternateAmount=""
currency="USD">5.95</Money>
                    </NetAmount>
                    <Distribution>
                        <Accounting name="DistributionCharge">
                            <AccountingSegment id="301">
                                <Name xml:lang="en">AccountingCombinationID</
Name>
                                <Description
xml:lang="en">AccountingCombinationID</Description>
                            </AccountingSegment>
                            <AccountingSegment id="GD">
                                <Name xml:lang="en">Company</Name>
                                <Description xml:lang="en">ID</Description>
                            </AccountingSegment>
                            <AccountingSegment id="">
                                <Name xml:lang="en">ERPSplitValue</Name>
                                <Description xml:lang="en">ERPSplitValue</
Description>
                            </AccountingSegment>
                            <AccountingSegment id="100">
                                <Name xml:lang="en">Percentage</Name>
                                <Description xml:lang="en">Percentage</
Description>
                            </AccountingSegment>
                            <AccountingSegment id="US100">
                                <Name xml:lang="en">BusinessUnit</Name>
                                <Description xml:lang="en">ID</Description>
                            </AccountingSegment>
                            <AccountingSegment id="3100">
                                <Name xml:lang="en">CostCenter</Name>
                                <Description xml:lang="en">ID</Description>
```

```
                            </AccountingSegment>
                            <AccountingSegment id="US100">
                                <Name xml:lang="en">ProcurementUnit</Name>
                                <Description xml:lang="en">ID</Description>
                            </AccountingSegment>
                            <AccountingSegment id="5008">
                                <Name xml:lang="en">SubAccount</Name>
                                <Description xml:lang="en">ID</Description>
                            </AccountingSegment>
                            <AccountingSegment id="7752">
                                <Name xml:lang="en">Account</Name>
                                <Description xml:lang="en">ID</Description>
                            </AccountingSegment>
                        </Accounting>
                        <Charge>
                            <Money alternateCurrency="" alternateAmount=""
                                currency="USD">5.95</Money>
                        </Charge>
                    </Distribution>
                    <Comments>
                        [Ariba added comment: Created in the name and on behalf
of the supplier]
                        <Attachment>
                            <URL>cid:test.pdf</URL>
                        </Attachment>
                    </Comments>
                    <Extrinsic name="OriginatingSystemLineNumber">0</Extrinsic>
                    <Extrinsic name="Custom(L)" />
                </InvoiceDetailItem>
            </InvoiceDetailOrder>
            <InvoiceDetailSummary>
                <SubtotalAmount>
                    <Money alternateCurrency="" alternateAmount=""
currency="USD">5.95</Money>
                </SubtotalAmount>
                <Tax>
                    <Money alternateCurrency="" alternateAmount=""
currency="USD">0.00</Money>
                    <Description xml:lang="en">TotalTax                    </
Description>
                </Tax>
                <SpecialHandlingAmount>
                    <Money alternateCurrency="" alternateAmount=""
currency="USD">0.00</Money>
                </SpecialHandlingAmount>
                <ShippingAmount>
                    <Money alternateCurrency="" alternateAmount=""
currency="USD">0.00</Money>
                </ShippingAmount>
                <GrossAmount>
                    <Money alternateCurrency="" alternateAmount=""
currency="USD">5.95</Money>
                </GrossAmount>
                <InvoiceDetailDiscount>
                    <Money alternateCurrency="" alternateAmount=""
currency="USD">0.00</Money>
                </InvoiceDetailDiscount>
                <NetAmount>
                    <Money alternateCurrency="" alternateAmount=""
currency="USD">5.95</Money>
                </NetAmount>
                <DueAmount>
                    <Money alternateCurrency="" alternateAmount=""
currency="USD">5.95</Money>
                </DueAmount>
            </InvoiceDetailSummary>
        </InvoiceDetailRequest>
    </Request>
```

```
</cXML>
```

Sample cXML for a Contract-Based Invoice

Here's a cXML invoice example for buyers importing external contract-based cXML invoices to SAP Ariba Buying and Invoicing or SAP Ariba Invoice Management.

> **Note**
>
> This example is for a standalone buyer site. In multi-ERP configurations, include both the Ariba Network ID and the business application system ID from SAP Business Network in the `To` and `Sender` credentials, as in `ERPCHILD1-AN71000003970`.

Sample Code

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE cXML SYSTEM "http://svcdev6.ariba.com/schemas/cXML/1.2.043/
InvoiceDetail.dtd">
<cXML payloadID="1567054577535-48042314483318910.163.0.90"
timestamp="2019-08-28T21:56:17-07:00">
<Header>
    <From>
        <Credential domain="privateid">
            <Identity>1234567</Identity>
        </Credential>
    </From>
    <To>
        <Credential domain="NetworkID">
            <Identity>AN71000003970</Identity>
        </Credential>
    </To>
    <Sender>
        <Credential domain="NetworkID">
            <Identity>AN71000003970</Identity>
            <SharedSecret>welcome1a</SharedSecret>
        </Credential>
        <UserAgent>Buyer</UserAgent>
    </Sender>
</Header>
<Request deploymentMode="production">
<InvoiceDetailRequest>
<InvoiceDetailRequestHeader invoiceDate="2019-06-14T07:41:40-07:00"
invoiceID="34343433333" invoiceOrigin="supplier" operation="delete"
purpose="standard">
    <InvoiceDetailHeaderIndicator/>
    <InvoiceDetailLineIndicator isAccountingInLine="yes"/>
    <InvoicePartner>
        <Contact addressID="US006" role="billTo">
            <Name xml:lang="en">New York Sales</Name>
            <PostalAddress name="default">
                <Street>4500 Avenue Street</Street>
                <City>New York</City>
                <State>NY</State>
                <PostalCode>10037</PostalCode>
                <Country isoCountryCode="US">United States</Country>
            </PostalAddress>
            <Phone name="work">
                <TelephoneNumber>
                    <CountryCode isoCountryCode="US">1</CountryCode>
```

```
                    <AreaOrCityCode>212</AreaOrCityCode>
                    <Number>5551212</Number>
                </TelephoneNumber>
            </Phone>
            <Fax name="work">
                <TelephoneNumber>
                    <CountryCode isoCountryCode="US"/>
                    <AreaOrCityCode/>
                    <Number/>
                </TelephoneNumber>
            </Fax>
        </Contact>
    </InvoicePartner>
    <InvoicePartner>
        <Contact addressID="RA11-1" role="remitTo">
            <Name xml:lang="en">My Contact 1</Name>
            <PostalAddress name="default">
                <Street>3450 Park Road</Street>
                <City>Austin</City>
                <State>TX</State>
                <PostalCode>78701</PostalCode>
                <Country isoCountryCode="US">United States</Country>
            </PostalAddress>
        </Contact>
    </InvoicePartner>
    <InvoicePartner>
        <Contact addressID="14" role="from">
            <Name xml:lang="de">jeder Name</Name>
            <PostalAddress name="Standardwert">
                <Street>Schelmenwasenstrasse 00</Street>
                <City>Stuttgart</City>
                <PostalCode>7000</PostalCode>
                <Country isoCountryCode="DE">Deutschland</Country>
            </PostalAddress>
        </Contact>
    </InvoicePartner>
    <InvoiceDetailShipping>
        <Contact addressID="14" role="shipFrom">
            <Name xml:lang="de">jeder Name</Name>
            <PostalAddress name="Standardwert">
                <Street>Schelmenwasenstrasse 00</Street>
                <City>Stuttgart</City>
                <PostalCode>7000</PostalCode>
                <Country isoCountryCode="DE">Deutschland</Country>
            </PostalAddress>
            <Phone name="work">
                <TelephoneNumber>
                    <CountryCode isoCountryCode="DE">49</CountryCode>
                    <AreaOrCityCode>1</AreaOrCityCode>
                    <Number>1720000000</Number>
                </TelephoneNumber>
            </Phone>
        </Contact>
        <Contact addressID="US006" role="shipTo">
            <Name xml:lang="en">New York Sales</Name>
            <PostalAddress name="default">
                <Street>4500 Street Avenue</Street>
                <City>New York</City>
                <State>NY</State>
                <PostalCode>10037</PostalCode>
                <Country isoCountryCode="US">United States</Country>
            </PostalAddress>
            <Phone name="work">
                <TelephoneNumber>
                    <CountryCode isoCountryCode="US">1</CountryCode>
                    <AreaOrCityCode>212</AreaOrCityCode>
                    <Number>5551212</Number>
                </TelephoneNumber>
```

```
            </Phone>
            <Fax name="work">
                <TelephoneNumber>
                    <CountryCode isoCountryCode="US"/>
                    <AreaOrCityCode/>
                    <Number/>
                </TelephoneNumber>
            </Fax>
        </Contact>
    </InvoiceDetailShipping>
    <PaymentTerm payInNumberOfDays="30">
        <Discount><DiscountPercent percent="2"/></Discount>
    </PaymentTerm>
    <PaymentTerm payInNumberOfDays="45"></PaymentTerm>
    <PaymentTerm payInNumberOfDays="20">
        <Discount><DiscountPercent percent="3"/></Discount>
    </PaymentTerm>
    <Extrinsic name="invoiceSourceDocument">Contract</Extrinsic>
    <Extrinsic name="invoiceSubmissionMethod">cXML</Extrinsic>
    <Extrinsic name="paymentMethod"/>
    <Extrinsic name="PurchasingUnit">US100</Extrinsic>
    <Extrinsic name="Custom(H)"/>
</InvoiceDetailRequestHeader>
<InvoiceDetailOrder>
    <InvoiceDetailOrderInfo>
        <MasterAgreementReference agreementDate="2019-05-31T11:30:00-07:00"
agreementID="C1557309735069">
            <DocumentReference payloadID=""></DocumentReference>
        </MasterAgreementReference>
        <SupplierOrderInfo orderID="Unknown"></
SupplierOrderInfo>
    </InvoiceDetailOrderInfo>
    <InvoiceDetailItem quantity="10" invoiceLineNumber="1">
        <UnitOfMeasure>EA</UnitOfMeasure>
        <UnitPrice>
            <Money alternateCurrency="" alternateAmount="" currency="EUR">29.21</
Money>
        </UnitPrice>
        <InvoiceDetailItemReference lineNumber="1">
            <ItemID>
                <SupplierPartID>MON152X 6</SupplierPartID>
            </ItemID>
            <Description xml:lang="en-US">Monitor HD15M/M Extension Coax</
Description>
            <Classification domain="unspsc">432118</
Classification>
            <ManufacturerPartID>MON152X 6</ManufacturerPartID>
            <ManufacturerName>All Cables</ManufacturerName>
            <Country isoCountryCode="DE">Germany</Country>
        </InvoiceDetailItemReference>
        <SubtotalAmount>
            <Money alternateCurrency="" alternateAmount=""
currency="EUR">292.10</Money>
        </SubtotalAmount>
        <GrossAmount>
            <Money alternateCurrency="" alternateAmount=""
currency="EUR">292.10</Money>
        </GrossAmount>
        <NetAmount>
            <Money alternateCurrency="" alternateAmount=""
currency="EUR">292.10</Money>
        </NetAmount>
        <Distribution>
            <Accounting name="DistributionCharge">
                <AccountingSegment id="298">
                    <Name xml:lang="en">AccountingCombinationID</Name>
                    <Description xml:lang="en">AccountingCombinationID</
Description>
```

```
                </AccountingSegment>
                <AccountingSegment id="GD">
                    <Name xml:lang="en">Company</Name>
                    <Description xml:lang="en">ID</Description>
                </AccountingSegment>
                <AccountingSegment id="">
                    <Name xml:lang="en">ERPSplitValue</Name>
                    <Description xml:lang="en">ERPSplitValue</Description>
                </AccountingSegment>
                <AccountingSegment id="100">
                    <Name xml:lang="en">Percentage</Name>
                    <Description xml:lang="en">Percentage</Description>
                </AccountingSegment>
                <AccountingSegment id="US100">
                    <Name xml:lang="en">BusinessUnit</Name>
                    <Description xml:lang="en">ID</Description>
                </AccountingSegment>
                <AccountingSegment id="3100">
                    <Name xml:lang="en">CostCenter</Name>
                    <Description xml:lang="en">ID</Description>
                </AccountingSegment>
                <AccountingSegment id="US100">
                    <Name xml:lang="en">ProcurementUnit</Name>
                    <Description xml:lang="en">ID</Description>
                </AccountingSegment>
                <AccountingSegment id="5008">
                    <Name xml:lang="en">SubAccount</Name>
                    <Description xml:lang="en">ID</Description>
                </AccountingSegment>
                <AccountingSegment id="7520">
                    <Name xml:lang="en">Account</Name>
                    <Description xml:lang="en">ID</Description>
                </AccountingSegment>
            </Accounting>
            <Charge>
                <Money alternateCurrency="" alternateAmount=""
currency="EUR">292.10</Money>
            </Charge>
        </Distribution>
            <Comments></Comments>
            <Extrinsic name="GR Based Invoice">No</Extrinsic>
            <Extrinsic name="OriginatingSystemLineNumber">0</Extrinsic>
            <Extrinsic name="receiptID"></Extrinsic>
            <Extrinsic name="ReceiptLineNumber"></Extrinsic>
            <Extrinsic name="Custom(L)"></Extrinsic>
            <Extrinsic name="parentPOLineNumber">unknown</Extrinsic>
        </InvoiceDetailItem>
    </InvoiceDetailOrder>
    <InvoiceDetailSummary>
        <SubtotalAmount>
            <Money alternateCurrency="" alternateAmount=""
currency="EUR">292.10</Money>
        </SubtotalAmount>
        <Tax>
            <Money alternateCurrency="" alternateAmount="" currency="EUR">0.00</
Money>
            <Description xml:lang="en">TotalTax</Description>
        </Tax>
        <SpecialHandlingAmount>
            <Money alternateCurrency="" alternateAmount="" currency="EUR">0.00</
Money>
        </SpecialHandlingAmount>
        <ShippingAmount>
            <Money alternateCurrency="" alternateAmount="" currency="EUR">0.00</
Money>
        </ShippingAmount>
        <GrossAmount>
```

```
            <Money alternateCurrency="" alternateAmount=""
currency="EUR">292.10</Money>
        </GrossAmount>
        <InvoiceDetailDiscount>
            <Money alternateCurrency="" alternateAmount="" currency="EUR">0.00</
Money>
        </InvoiceDetailDiscount>
        <TotalCharges>
            <Money alternateCurrency="" alternateAmount="" currency="EUR">0.00</
Money>
        </TotalCharges>
        <NetAmount>
            <Money alternateCurrency="" alternateAmount=""
currency="EUR">292.10</Money>
        </NetAmount>
        <DueAmount>
            <Money alternateCurrency="" alternateAmount=""
currency="EUR">292.10</Money>
        </DueAmount>
    </InvoiceDetailSummary>
</InvoiceDetailRequest>
</Request>
</cXML>
```

Creating an Approval Rule for External cXML Invoices

Use this procedure to create approval rules for external cXML invoices.

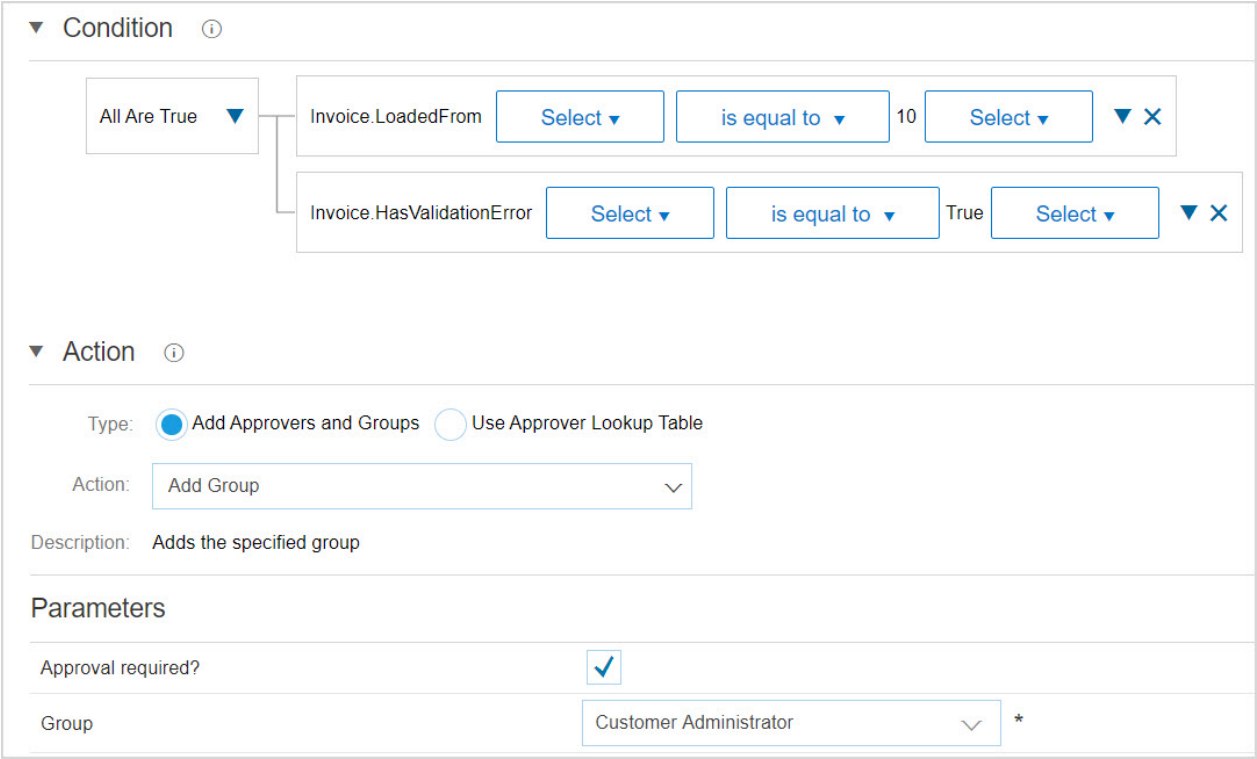
Prerequisites

Members of the **Customer Administrator** or **Invoicing Administrator** group can manage approval processes for invoicing.

This procedure assumes you're familiar with managing approval processes.

Context

Your approval process for invoices and invoice reconciliation (IR) documents can include approvers specifically for external cXML invoices posted directly to SAP Ariba Buying and Invoicing or SAP Ariba Invoice Management.
To identify imported cXML invoices, you create an approval rule condition using the `LoadedFrom` field.
To identify invoices that were submitted with errors, you create another approval rule condition using the `HasValidationError` field.



Procedure

1. Choose **Manage** > **Approval Processes**.
2. Find and open the approval process you want to edit, and choose **Edit** in the approval process editor.
3. In the **Approval Process Diagram** section, add an approval rule, filling in a rule title and description.
4. In the **Condition** section, add a **Field Match** condition.
5. Choose **Select** > **Select Field**.

   The **Select Field** popup opens.
6. Scroll down and select the `LoadedFrom` field.
7. From the second **Select** button, choose **Select Value**.
8. In the **Value** field, enter 10.

   The value 10 represents external cXML invoices that were imported.
9. In the **Condition** section, add a **Field Match** condition.
   a. Select the `HasValidationError` field.
   b. From the second **Select** button, choose **Select Value**.
   c. In the **Value** window, select **True**.
10. Add an action for the approval rule.

About Exporting Invoices

Invoices that have been transmitted to SAP Ariba Buying and Invoicing solutions configured for the Generic, SAP, and PeopleSoft variants and integrated with your ERP through the Ariba Integration Toolkit, but have not yet had reconciliations approved are processing invoices.

Processing invoices are preliminary and are subject to a number of changes during reconciliation. For example, the reconciliation process might adjust an invoice total downwards because some invoiced goods were not actually received. Once the reconciliations are approved, the invoices are then "OK to Pay."

The `Export Invoices` task exports header, detail, and split accounting data for all invoices that are processing but have not yet had reconciliations approved. You can use this preliminary invoice data in a number of different accounting processes. For example, some companies want to account for, or book, all outstanding invoices preliminarily so that they have accurate data on all potential accounts payable at a given time. They distinguish these preliminary bookings from final reconciled, approved invoice bookings.

Each invoice that SAP Ariba Buying and Invoicing is currently processing generates a record for invoice export. When a processing invoice is rejected or cancelled, that event generates another record for invoice export. The `Status` and `StatusDate` fields in the exported data file show when an invoice is marked as processing, rejected or canceled. After an invoice reconciliation is approved, it is no longer included in the invoice export. Instead, it is included in the payment request export.

The following table describes how invoices with different statuses are included in data exports:

Status	Export
Processing	Data for the invoice is added to the invoice export with a status of Processing. In most sites, invoices automatically go straight into reconciliation. However, some sites are configured to require manual approval of all processing invoices before they can be reconciled; the invoice is not added to the export until it has been manually or automatically approved for reconciliation.
Rejected or Cancelled	Data for the invoice is added to the invoice export with a status of Rejected or Cancelled. This data is only added if the invoice was already Processing. Data for invoices that are rejected or cancelled before entering Processing status are not added to the export.
Approved	Data for approved invoice reconciliations are not included in the invoice export; instead, they are added to the payment request (OK-to-Pay) export.

Note

- Invoice export is an optional feature and is disabled by default; SAP Ariba Customer Support must enable it in your site by setting the `Application.Invoicing.InvoiceExport.Enabled` parameter to `Yes`. Once the export is enabled, the exported invoice data only includes new invoices. It does not include any invoices that are in the process of or have completed reconciliation, approval of invoice reconciliation, or payment at the time of enablement.
- SAP Ariba recommends that you configure your integrations to pull invoice export data before payment request data so that it can be used for preliminary invoice bookings.

Invoice Data Export Task [page 153]

Invoice Data in Generic Variants [page 154]
Invoice Data in SAP Variants [page 157]
Invoice Data in PeopleSoft Variants [page 161]

Related Information

Managing Payment Methods [page 68]

Invoice Data Export Task

You can export invoices by running the `Export Invoices` data export task in Ariba Administrator.

Exporting Invoices

Use this procedure to export invoices by running the `Export Invoices` data export task in Ariba Administrator.

Procedure

1. Choose **Site Manager** > **Data Import/Export** and click the **Export** tab.
2. Find the `Export Invoices` task and click **Export**.
3. Select a date range for the export by choosing dates in the **From** and **To** fields. Ariba Administrator exports all the invoices that register a status of Processing, Rejected, or Cancelled within that time range. Invoices that are reconciled and approved during the specified date range are not included in the export.
4. Click **Run** to export the data.

Next Steps

After the data has finished downloading, click **Open** to view the contents of the ZIP file or **Save** to save the file to your local hard drive.

Invoice Data in Generic Variants

When you run the `Export Invoices` data export task in Ariba Administrator on Generic variants, it creates a ZIP file containing three CSV files with the invoice data.

The `Export Invoices` data export task exports a ZIP file containing the following CSV files:

- `InvoiceHeaderExport.csv`
- `InvoiceDetailExport.csv`
- `InvoiceSplitAccountingExport.csv`

About Invoice Header Data in Generic Variants

`InvoiceHeaderExport.csv` contains general information about each invoice.

The following table describes the columns in `InvoiceHeaderExport.csv`.:

Field	Description	Format
`InvoiceRequester`	The unique name of the user who requested the invoice. This field does not appear on the invoice in SAP Ariba Buying and Invoicing.	String (255)
`StatusDate`	The date on which the invoice is received/approved, rejected, or cancelled. This field does not appear on the invoice in SAP Ariba Buying and Invoicing.	String (100)
`PaymentTerms`	The invoice's payment terms.	String (50)
`InvoiceType`	The type of the invoice: standard invoice, header-level credit memo, line-level credit memo, or debit memo. This field does not appear on the invoice.	String (32)
`TaxExchangeRate`	The exchange rate captured when a cXML invoice comes in from SAP Business Network and has tax lines in different currency than the item lines. The alternate amount/currency for the tax lines is calculated using this exchange rate. This field does not appear on the invoice in SAP Ariba Buying and Invoicing.	Decimal
`SupplierLocationContactID`	The ID of the contact person at the specified supplier location.	String (50)
`RemittanceLocation`	The remittance address for the invoice.	String (50)
`SupplierLocation`	The unique internal identifier of the supplier location to which the payment is made.	String (50)
`Supplier`	The unique identifier for the supplier that generated the invoice.	String (50)
`GrossAmount`	The gross total of the invoice.	Decimal
`GrossAmountCurrency`	The currency code for the gross amount of the invoice.	String (50)
`InvoiceID`	The unique identifier for the invoice.	String (50)

Field	Description	Format
`Status`	The status of the invoice: Processing for received/approved (but not yet reconciled) invoices, Rejected for rejected invoices, and Cancelled for cancelled invoices. This field does not appear on the invoice in SAP Ariba Buying and Invoicing.	String (100)
`InvoiceNumber`	The number associated with the invoice.	String (50)
`InvoiceDate`	The date on which the invoice was created.	String (100)
`PurchasingUnit`	The unique identifier for the purchasing unit that placed the order.	String (50)

Following is an example of `InvoiceHeaderExport.csv`:

```
InvoiceId,PurchasingUnit,InvoiceNumber,InvoiceDate,InvoiceType,Status,StatusDate,Sup
plier,
GrossAmount,GrossAmountCurrency,PaymentTerms,TaxExchangeRate,SupplierLocation,
SupplierLocationContactID,InvoiceRequester,RemittanceLocation
"INVINV159-159","US100","INV159","02/20/2013 11:08:48-0800","standard","Processing",
"02/20/2013 11:13:44-0800","11","6.9500","USD","PT1","0","14","63","ghalas","RA11-1"
"INVINV159-159","US100","INV159","02/20/2013 11:09:48-0800","standard","Cancelled",
"02/20/2013 11:13:44
-0800","11","6.9500","USD","PT1","0","14","63","ghalas","RA11-1"
```

About Invoice Detail Data in Generic Variants

`InvoiceDetailExport.csv` contains information about invoice line items.

The following table describes the columns in `InvoiceDetailExport.csv`.

Field	Description	Format
`ERPOrderLineNumber`	The purchase order line number associated with the invoice reconciliation line number.	Integer
`LineTypeCategory`	The category of the invoice line item. Line categories separate procurement line items from various special handling categories such as tax and freight. The line categories are as follows: • 1 — Line Item (for all line items) • 2 — Tax charge (for all tax charges) • 4 — Freight charge (for all freight-related charges) • 8 — Handling charge (for all handling-related charges) • 16 — Charge added using a user-defined charge type • 32 — Discount (for all discounts related to invoicing)	Integer
`Description`	The description of the invoice line item.	String (2000)
`Amount`	The amount of the invoice line item, such as the amount for a commodity or a tax charge.	Decimal
`AmountCurrency`	The currency code for the amount.	String (50)
`InvoiceId`	The unique identifier for the invoice.	String (50)

Field	Description	Format
`UnitPrice`	The unit price for the invoice line item or, for tax line items, the tax charge.	Decimal
`LineNumber`	The unique identifier for an individual or grouped invoice line.	Integer
`ERPCommodityCode`	The ERP commodity code for the invoice line item. If your site does not use ERP commodity codes, this field is blank.	String (50)
`ParentLineNumber`	The unique identifier for an invoice item group that contains child grouped items.	Integer
`TaxCode`	The tax code for the invoice line item.	String (50)
`TaxType`	The tax type for the invoice line item; for example, `SalesTax`.	String (50)
`ERPOrderID`	The purchase order ID associated with the invoice reconciliation line number.	String (50)
`Purchasing Unit`	The unique identifier for the purchasing unit that generated the order.	String (50)

Following is an example of `InvoiceDetailExport.csv`:

```
InvoiceId,PurchasingUnit,LineNumber,ParentLineNumber,LineTypeCategory,TaxCode,TaxTyp
e,Description,Amount,UnitPrice,AmountCurrency,ERPCommodityCode,ERPOrderID,ERPOrderLi
neNumber
"INVINV159-159","US100","1",,"1",,,"Mac Angled Keyboard Adapter MD4M/
F","6.9500","6.9500","USD",,,
"INVINV159-159","US100","1",,"1",,,"Mac Angled Keyboard Adapter MD4M/
F","6.9500","6.9500","USD",,,
```

About Invoice Split Accounting Data in Generic Variants

`InvoiceSplitAccountingExport.csv` contains information about invoices where accounting is split among different subaccounts.

The following table describes the columns in `InvoiceSplitAccountingExport.csv`:

Field	Description	Format
`Company`	The unique identifier for the company.	String (50)
`AccountingCombinationID`	The unique identifier for the accounting combination defined in SAP Ariba Buying and Invoicing.	String (50)
`SubAccount`	The unique identifier for the sub-account.	String (50)
`Project`	The unique identifier for the project.	String (50)
`Account`	The unique identifier for the account; for example, `Expense`.	String (50)
`Quantity`	The line item quantity.	Decimal
`CostCenter`	The unique identifier for the cost center.	String (50)
`Product`	The unique identifier for the product.	String (50)

Field	Description	Format
`Region`	The unique identifier for the accounting region.	String (50)
`InvoiceId`	The unique identifier for the invoice.	String (50)
LineNumber	The unique identifier for an individual or grouped invoice line item.	Integer
BusinessUnit	The unique identifier for the general ledger business unit.	String (50)
SplitLineNumber	The unique identifier for the invoice line.	Integer
Purchasing Unit	The unique identifier for the purchasing unit that generated the order.	String (50)
Amount	The split amount.	Decimal

Following is an example of `InvoiceSplitAccountingExport.csv`:

```
InvoiceId,PurchasingUnit,LineNumber,SplitLineNumber,Company,BusinessUnit,CostCenter,
Account,
SubAccount,Product,Project,Quantity,Amount,Region,AccountingCombinationID
"INVINV159-159","US100","1","1","GD","US100","3100","7520","5008",,,"1.0000","6.9500
",,"298"
"INVINV159-159","US100","1","1","GD","US100","3100","7520","5008",,,"1.0000","6.9500
",,"298"
```

Invoice Data in SAP Variants

When you run the `Export Invoices` data export task in Ariba Administrator on SAP variants, it creates a ZIP file containing three CSV files with the invoice data.

In SAP variants, the `Export Invoices` task exports an `InvoiceExport.zip` file containing the following invoice data CSV files:

- `InvoiceHeaderExport.csv`
- `InvoiceDetailExport.csv`
- `InvoiceAccountDetails.csv`

About Payment Header Data in SAP Variants

`InvoiceHeaderExport.csv` contains general information about each invoice.

The following table describes the columns in `InvoiceHeaderExport.csv`.

Field	Description	Format
GROSS_AMT	The total cost of the invoice.	BigDecimal
PMNTTRMS	A unique code of payment terms.	String (50)

Field	Description	Format
StatusDate	The date on which the invoice was approved, canceled, or rejected. This field does not appear on the invoice.	String (100)
COMP_CODE	The company code for invoice.	String (50)
InvoiceType	The type of the invoice: standard invoice, header-level credit memo, line-level credit memo, or debit memo. This field does not appear on the invoice.	String (32)
DOC_DATE	The date on the invoice.	Date
DIFF_INV	The supplier ID.	String (50)
InvoiceId	The unique identifier for the invoice.	String (50)
Status	The status of the invoice for export. This field does not appear on the invoice.	String (100)
REF_DOC_NO	The invoice number as reference document number. The maximum permissible length for this field is 16 characters. This field does not appear on the invoice.	String (16)
InvoiceDate	The date on which the invoice was created.	String (100)
CURRENCY	The type of reporting currency of the invoice.	String (50)
URL	URL to access the site. This field does not appear on the invoice.	String (100)

Following is an example of `InvoiceHeaderExport.csv`:

```
InvoiceId,REF_DOC_NO,COMP_CODE,CURRENCY,DOC_DATE,PMNTTRMS,InvoiceDate,DIFF_INV,Invoi
ceType,Status,StatusDate,URL,GROSS_AMT
“INVINV-4853-INV856566678-20”,”INV-4853-INV8565”,”3000”, “USD”, “20140601”,”PT1”,
“6/1/2014”, “6:20”, “100”, “standard”, “Processing”, URL, “695”
```

About Invoice Detail Data in SAP Variants

`InvoiceDetailExport.csv` contains information about invoice line items.

The following table describes the columns in `InvoiceDetailExport.csv`:

Field	Description	Format
ORDERID	The unique value of internal order account assignment category for that line item.	String (50)
LINE_TYPE_LOOKUP_CODE	The category of line item. This field does not appear on the invoice.	Integer
COSTCENTER	The unique value of the cost center account assignment category for each line item.	String (50)
GL_ACCOUNT	The unique value of the general ledger accounting type for each line item.	String (50)
LINEITEM_AMOUNT	The amount of the line item.	BigDecimal
ITEM_AMOUNT	The amount for the line item.	BigDecimal

Field	Description	Format
INVOICE_DOC_ITEM	The line number of the line item.	Integer
PLANT	The unique code of the ship-to location for each line item.	String (50)
QUANTITY	The quantity for the line item.	BigDecimal
CHARGE_FLAG	The line number of the line item.	Integer
PBQ_DESC	The price basis quantity fields for each item in the requisition line items.	String (2000)
ITEM_AMOUNT_ROUNDED	The amount for the line item in ERP precision.	BigDecimal
IS_COMPOSITE	The flag indicating if the line is composite or not. Composite lines may consist of child lines for a regular line item. This field does not appear on the invoice.	Boolean
COMP_CODE	The company code of the invoice.	String(50)
PO_UNIT	A unit of measure for the line item.	String(50)
ITEM_TEXT	The description of a reference line.	String(2000)
LINE_COMP_CODE	The unique value of the Company Code for the line item. This field will be populated for non-purchase order invoices and `Application.Invoicing.AllowLegalEntityCodeAtSplitLevel` parameter is set to Yes.	String(50)
HDR_LEVEL	This field determines whether a line item is at the header level. This field does not appear on the invoice.	Boolean
PRICE_UNIT	The quantity on which the price is quoted.	BigDecimal
PO_PR_UOM	The quantity on which the price is quoted.	String(50)
INVOICEID	The unique ID for the invoice.	String(50)
TAX_CODE	The tax code.	String(50)
CONV_UNIT	The quantity on which the price is quoted.	BigDecimal
ORDER	The ERP purchase order number.	String(50)
PO_NUMBER	The unique ID of every purchase order the invoice belongs to. In case of an ERP order, the field provides a reference to the ERP order ID.	String(50)
WBS_ELEM	The unique code of the WBS element account assignment category for that line item.	String(50)
PO_ITEM	The line number of the corresponding line item.	Integer
ASSET_NO	The unique value of the Asset account assignment category for each line item.	String(50)

Following is an example of `InvoiceDetailExport.csv:`

```
InvoiceID, Item_Amount, Item_Amount_Rounded, LineItem_Amount, WBS_Elem, Quantity,
Asset_No, CostCenter, PO_Unit, Invoice_Doc_Item, OrderID, GL_Account, Comp_Code,
Line_Comp_Code, Tax_Code, Plant, PO_Item, Item_Text, Order, Line_Type_Lookup_Code,
HDR_Level, PO_Number, Charge_Flag, Price_Unit, PO_PR_Uom, Conv_Unit, PBQ_Desc,
Is_Composite
```

```
“INVINV-4853-
INV856566678-20”,”695”,”695”,”695”,,”100”,,”2100”,”EA”,”1”,,”405000”,”3000”,”3000”,,
 ”3300”,”0”,”Adapter DB9F/MD8M Mac Serial”,”1”,”No”,,,,,,”No”
```

About Invoice Split Accounting Data in SAP Variants

`InvoiceSplitAccountingExport.csv` contains information about invoices where accounting is split among different subaccounts.

The following table describes the columns in `InvoiceAccountDetails.csv`:

Field	Description	Format
ORDERID	The unique value of internal order account assignment category for the line item.	String(50)
SERIAL_NO	The serial number of the line item.	String(100)
QUANTITY	The quantity for the line item.	BigDecimal
PO_UNIT	The unique unit of measure value.	String(50)
CATEGORY	The category of the line item. This field does not appear on the invoice.	Integer(50)
COSTCENTER	The unique value of the cost center account assignment category for each line item.	String(4)
SUB_NUMBER	The asset SubNumber for the asset account assignment category for each line item. This field does not appear on the invoice.	String(50)
LINE_COMP_CODE	The unique value of the company code for the line item. This field will be populated for non-purchase order invoices and `Application.Invoicing.AllowLegalEntityCodeAtSplitLevel` parameter is set to Yes.	String(50)
PO_PR_UOM	The unit of measure for the quantity defined by price, basis, and quantity.	String(50)
GL_ACCOUNT	The unique value of the general ledger accounting type for each line item.	String(50)
INVOICEID	The unique ID of the invoice.	String(50)
INVOICE_DOC_ITEM	The line number of the line item.	Integer
ITEM_AMOUNT	The amount for the line item.	BigDecimal
TAX_CODE	The unique tax type code.	String(50)
PO_NUMBER	The unique ID of every purchase order the invoice belongs to. In case of an ERP Order, the field provides a reference to the ERP Order ID.	String(50)
ORDER	The ERP purchase order number.	String(50)
WBS_ELEM	The unique code of the WBS element account assignment category for that line item.	String(50)
ITEM_AMOUNT_ROUNDED	The amount for the line item in ERP precision.	BigDecimal

Field	Description	Format
ASSET_NO	The unique value of the Asset account assignment category for each line item.	String(50)

Following is an example of `InvoiceAccountDetails.csv`:

```
Item_Amount,Item_Amount_Rounded,WBS_Elem,Asset_No,CostCenter,Quantity,PO_Unit,Serial
_
No,Invoice_Doc_Item,OrderID,GL_Account,Tax_Code,InvoiceID,Category,Order,Line_Comp_C
ode,Sub_Number,PO_PR_UOM,
            PO_Number
         “695”,”695”,,,”2100”,”100”,”EA”,,”1”,,”405000”,,”INVINV-4853-
INV856566678-20”,”1”,,”3000”,,,
```

Invoice Data in PeopleSoft Variants

When you run the `Export Invoices` data export task in Ariba Administrator on PeopleSoft variants, it creates a ZIP file containing three CSV files with the invoice data.

In PeopleSoft variants, the `Export Invoices` task exports an `InvoiceExport.zip` file that contains the following invoice data CSV files:

- `InvoiceHeaderExport.csv`
- `InvoiceDetailExport.csv`
- `InvoiceSplitAccountingExport.csv`

About Payment Header Data in PeopleSoft Variants

`InvoiceHeaderExport.csv` contains general information about each invoice.

The following table describes the columns in `InvoiceHeaderExport.csv`:

Field	Description	Format
INV_ID	The unique invoice ID.	String(50)
BUSINESS_UNIT	The business unit for the invoice.	String(50)
INVOICE_ID	The unique invoice number/ID for the invoice/voucher.	String(50)
INVOICE_DT	The invoice/voucher date.	String(100)
VENDOR_SETID	The set ID of the supplier. This field does not appear on the invoice.	String(50)

Field	Description	Format
InvoiceType	The type of invoice: • Standard invoice • Header-level credit memo • Line-level credit memo • Debit memo This field does not appear on the invoice.	String(32)
VENDOR_ID	The vendor ID of the supplier. This field does not appear on the invoice.	String(10)
Status	The status of the invoice for export.	String(100)
StatusDate	The date on which the invoice was approved, canceled, or rejected.	String(100)
GROSS_AMT	The total cost of the invoice.	BigDecimal
TXN_CURRENCY_CD	The currency of the invoice.	String(50)
PAYMENT_TERMS	The payment term.	String(50)
VENDOR_LOC	The location of the supplier.	String(20)
VENDOR_ADDR_SEQ_NUM	The sequence number of the supplier's location. This field does not appear on the invoice.	String(10)

Following is an example of `InvoiceHeaderExport.csv`:

```
        INV_ID, BUSINESS_UNIT, INVOICE_ID, INVOICE_DT, VENDOR_SETID, InvoiceType,
                    VENDOR_ID, Status, StatusDate, GROSS_AMT, TXN_CURRENCY_CD,
PAYMENT_TERMS,
                    VENDOR_LOC, VENDOR_ADDR_SEQ_NUM
        'INVINV-4853-INV159566299-10”,”US003”,”INV-4853-
INV159566299”,,”SHARE”,”standard”,”USA0000006”,”Processing”,”6/1/2014
5:10”,”695”,’USD”,”PT1”,”HQ”,”1”,,
```

About Invoice Detail Data in PeopleSoft Variants

`InvoiceDetailExport.csv` contains information about invoice line items.

The following table describes the columns in `InvoiceDetailExport.csv`:

Field	Description	Format
INV_ID	Ariba Procure-to-Pay Professional invoice ID.	String(50)
BUSINESS_UNIT	The business unit for the invoice.	String(50)
VOUCHER_LINE_NUM	The voucher line number. This field does not appear on the invoice.	Integer
PARENT_LINE_NUM	The parent line number for each line in the invoice. This field does not appear on the invoice.	Integer

Field	Description	Format
LINE_TYPE_LOOKUP_CODE	The line type lookup code that distinguishes each line with tax codes. This field does not appear on the invoice.	Integer
TAX_CODE	The tax code. This field does not appear on the invoice.	String(25)
TAX_TYPE	The tax type.	String(50)
DESCR	The description of the line item.	String(2000)
UNIT_PRICE	The unit price of each line item.	BigDecimal
CURRENCY_CD	The currency of each line item.	String(50)
PO_ID	The ERP PO Number of the purchase order-based invoice. In case of an ERP Order, the field provides an ERP Order ID.	String(50)
SHIPTO	The ShipTo address of the invoice line item.	String(50)
UNIT_OF_MEASURE	The unit of measure of each line item.	String(50)
LINE_NBR	The ERP purchase order line number associated with the invoice line number.	Integer
BUSINESS_UNIT_PO	The business unit of a purchase order-based invoice.	String(50)
PriceBasisQuantity	The quantity on which the price is quoted.	BigDecimal
PriceBasisQuantityUOM	The quantity on which the price is quoted.	String(50)
ConversionFactor	The quantity on which the price is quoted. This field does not appear on the invoice.	BigDecimal
PriceBasisQuantityDesc	The price basis quantity fields for each item in requisition line items.	String(2000)
IsComposite	Indicates whether this line is composite or not. Composite lines may consist of child lines for a regular line. This field does not appear on the invoice.	Boolean

Following is an example of `InvoiceDetailExport.csv:`

```
INV_ID,BUSINESS_UNIT,VOUCHER_LINE_NUM,PARENT_LINE_NUM,LINE_TYPE_LOOKUP_CODE,TAX_CODE
,TAX_TYPE,DESCR,UNIT_PRICECURRENCY_CD,PO_ID,SHIPTO,UNIT_OF_MEASURE,LINE_NBR,BUSINESS
_UNIT_PO,PriceBasisQuantity,PriceBasisQuantityUOM,ConversionFactor,PriceBasisQuantit
yDesc,IsComposite
“INVINV-4853-INV159566299-10”,”US003”,”1”,”1”,,”Adapter DB9F/MD8M Mac
Serial”,”6.95”,”USD”,,”US013”,”EA”,”0”,,,,,”No”
```

About Invoice Split Accounting Data in PeopleSoft Variants

`InvoiceSplitAccountingExport.csv` contains information about invoices where accounting is split among different subaccounts.

The following table describes the columns in `InvoiceSplitAccountingExport.csv`:

Field	Description	Format
INV_ID	Ariba Procure-to-Pay Professional invoice ID.	String(50)

Field	Description	Format
BUSINESS_UNIT	The business unit for an invoice.	String(50)
VOUCHER_LINE_NUM	The voucher line number of each line item. This field does not appear on the invoice.	Integer
DISTRIB_LINE_NUM	The distribution line number of each line item. This field does not appear on the invoice.	Integer
BUSINESS_UNIT_GL	The general ledger business unit.	String(50)
ACCOUNT	The account number.	String(20)
DEPTID	The department ID.	String(20)
QTY_VCHR	The quantity of each line item.	BigDecimal
MERCHANDISE_AMT	The merchandise amount of each line item. This field does not appear on the invoice.	BigDecimal
SPLIT_TYPE	The split type.	String(50)
PO_ID	The purchase order number of the purchase order-based invoice. In case of ERP Order, the field provides a reference to the ERP order ID.	String(50)

Following is an example of `InvoiceSplitAccountingExport.csv:`

```
INV_ID,BUSINESS_UNIT,VOUCHER_LINE_NUM,DISTRIB_LINE_NUM,BUSINESS_UNIT_GL,ACCOUNT,DEPT
ID,QTY_VCHR,MERCHANDISE_AMT,SPLIT_TYPE,PO_ID
“INVINV-4853-
INV159566299-10”,”US003”,”1”,”1”,”US003”,”131000”,”10200”,”100”,”695”,”_Percentage”
```

Exporting Payment Requests in SAP Ariba Buying and Invoicing

About Exporting Payment Requests in SAP Ariba Buying and Invoicing [page 165]

About Payment Request Data Export Tasks [page 166]

PeopleSoft Payment Request Data [page 168]

SAP Payment Request Data [page 174]

Generic Payment Request Data [page 181]

Amount Precision in Payment Request Export Files [page 189]

About Exporting Payment Requests in SAP Ariba Buying and Invoicing

After an invoice has been approved in SAP Ariba Buying and Invoicing, the final step is to send a payment to the supplier. SAP Ariba Buying and Invoicing creates a document called a payment request to track the payment process.

SAP Ariba Buying and Invoicing creates the payment request and the IR document at the same time.

Most payment requests represent amounts due to a supplier, such as for an unpaid invoice. SAP Ariba Buying and Invoicing also provides support for payment requests associated with credit memos. For example, if your company purchased equipment that was later returned, the supplier might first issue an invoice and then (after the return) a credit memo, which represents the amount due to be refunded to you. SAP Ariba Buying and Invoicing creates credit payment requests to represent credit memos.

The payment request includes a payment schedule based on the payment terms in the invoice reconciliation (IR) document. Payment requests also include the amounts for indirect and withholding tax lines and charges.

> **Note**
>
> SAP Ariba Buying and Invoicing exports payment requests that use the External System payment model only. The External System payment model specifies that payments are handled in an external system, such as an ERP system.

> **Note**
>
> This information applies to SAP Ariba Buying and Invoicing only. In SAP Ariba Invoice Management, reconciled and approved invoices are transmitted from SAP Business Network to your ERP through an SAP Business Network adapter.

Related Information

Payment Request Documents

About Payment Request Data Export Tasks

Your site can export payment requests (also referred to as OK-to-Pay invoice data) using one of several data export tasks in Ariba Administrator. These export tasks support the CSV file channel and the Web services channel.

For information about exporting payment requests to an SAP ERP system using SAP Integration Suite, managed gateway for spend management and SAP Business Network, see Invoice in SAP Integration Suite, Managed Gateway for Spend Management and SAP Business Network Overview Guide.

Table 3: Payment Request Data Export Tasks

Task Name	Description
Export Payment Requests	Exports payment request data to 5 CSV files. This task can be used with SAP ERP and PeopleSoft as well as generic ERP systems that can handle line item details and split accounting information in separate files.
Export Payment Requests with Advance Payments	This task is for sites configured to export approved invoices for payment directly from the SAP Ariba Buying solutions to the ERP system in CSV file format, and not from SAP Business Network. This task exports payment request data, including details of advance payments and Funds Management accounting information (if enabled for SAP ERP), to the external ERP system. See About Exporting Payment Requests with Advance Payments [page 209].
Export Payment Requests with Advance Payments and Taxes	This task is for SAP Ariba sites that use third-party tax integration and are configured as PeopleSoft or Generic variants. This task exports payment request details, including details of advance payments, Funds Management accounting information (if enabled), and taxes to the ERP system. This task is similar to the task **Export Payment Requests with Advance Payments**, except it includes tax details in a separate CSV file (`PaymentTaxExport.csv`) or Web services element (`&ltPayment_PaymentTaxExport_Item>`). For more information, see Tax Info in Payment Requests with Enhanced third-party tax calculation for invoicing.

SAP Ariba Buying and Invoicing supports export of payment requests to the SAP ERP in real time (triggered when the transactions are ready to be sent to the ERP) using the Web services channel and SAP Netweaver.

> **Note**
>
> SAP Ariba Buying and Invoicing exports payment requests that use the External System payment model only. SAP Ariba Buying and Invoicing doesn't export partial payments, even if you import a partial remittance from your SAP system.

Using Web Services to Export Payment Requests

When you use Web services to expot payment requests, the following parameter controls which payment request export task your site uses:

Application.Invoicing.PayablePushEvent (set by SAP Ariba Support)

Specifies which payment export task the site uses. The setting determines which data is included payment request export. Following are the valid values:

- `PaymentRealTimeExport`: Enables the **Export Payment Requests** task. This task exports payment request details to the external system.
- `PaymentRealTimeExport_v2`: Enables the **Export Payment Requests with Advance Payments** task. This task exports payment request details including details of advance payments and Funds Management accounting information (if enabled) to the external system.
- `PaymentRealTimeExport_v3`: Enables the **Export Payment Requests with Advance Payments and Taxes** task. This task exports payment request details including details of advance payments and Funds Management accounting information (if enabled) and taxes to the external system.

After SAP Ariba sets the parameter, a user who belongs to the **Integration Admin** or **Customer Administrator** group configures the export task, which involves setting the end point, enabling the task, and disabling the task previously used. For more information, see Web Services Overview.

For ERP Systems That Require Line Item and Split Accounting Information in 1 File

If your ERP system can't handle line item details and split accounting information in separate files, you can use the task **Export Payment in 3 file Format**. This task exports data to 3 CSV files. It's available only for Generic variants. It doesn't include tax information. Customers who use the Generic variant can configure their ERP to export tax data as needed.

Exporting Payment Requests Manually

Use this procedure to export payment requests to CSV files manually using the data export task appropriate for your configuration.

Procedure

1. Choose **Site Manager** > **Data Import/Export**.
2. On the **Export** tab, find the data export task you want to use.

   The tasks are described under About Payment Request Data Export Tasks [page 166],
3. Choose **Export** for the task.
4. Select a date range for the export by choosing dates in the **From** and **To** fields. Ariba Administrator exports all the payment requests that were marked "OK-to-Pay" during the specified date range. A payment request is marked "OK-to-Pay" when the associated invoice reconciliation document has been fully approved and the invoice is reconciled.
5. Choose **Run** to export the data.

   SAP Ariba Buying and Invoicing does not export partial payments, even if you import a partial remittance from your ERP system.

Next Steps

After the data has finished downloading, choose **Open** to view the contents of the ZIP file or **Save** to save the file to your local hard drive.

PeopleSoft Payment Request Data

The payment request data for sites integrated with PeopleSoft ERP systems is exported in a ZIP file containing the following five CSV files:

- `PaymentHeaderExport.csv`
- `PaymentDetailExport.csv`
- `PaymentBankExport.csv`
- `PaymentSplitAccountingExport.csv`
- `AdvancePaymentDetailExport.csv`

Payment Header Data in the PeopleSoft Variant

`PaymentHeaderExport.csv` contains general information about payments.

The following table describes the columns in `PaymentHeaderExport.csv`:

Column	Description	Format
PMNT_ID	The unique reference ID of the payment.	String (50)
ECQUEUEINSTANCE	The unique name of the electronic document in your ERP system.	String (50)
BUSINESS_UNIT	The PeopleSoft business unit.	String (50)
INVOICE_ID	The invoice number.	String (50)
INVOICE_DT	The invoice date.	Date
VENDOR_SETID	The vendor Set ID.	String (50)
VENDOR_ID	The vendor ID.	String (10)
GROSS_AMT	The gross invoice amount.	Numeric
TXN_CURRENCY_CD	The invoice transaction currency code.	String (50)
PAYMENT_TERMS	The unique internal identifier of the payment terms selected for the invoice. Payment terms specify the time period in which a supplier requests payment and any discounts that the supplier chooses to make available for early payment.	String (50)
TAX_ACCRUAL_PMNT	A value that indicates whether the payment is an accrual payment.	Boolean
VENDOR_LOC	The vendor location.	String (20)
REMIT_ADDR_SEQ_NUM	The unique internal identifier of the remittance location. When sending a payment to a given supplier location, SAP Ariba Buying and Invoicing sends to the remittance location associated with that supplier location.	String (50)
VENDOR_ADDR_SEQ_NUM	The vendor address sequence number.	String (10)
INVOICE_TYPE	The type, or purpose, of the invoice. For price adjustments, indicates line-item credit memo or line-item debit memo.	String
IS_PRICE_ADJUSTMENT	Indicates whether the invoice is a price adjustment. Applies to line-item credit memos and line-item debit memos only.	Boolean
REF_INVOICE_ID	Reference number of the original invoice line. Applies to line-item credit memos and line-item debit memos only.	String

Following is an example of `PaymentHeaderExport.csv`:

```
PMNT_ID,ECQUEUEINSTANCE,BUSINESS_UNIT,INVOICE_ID,INVOICE_DT,VENDOR_SETID,VENDOR_ID,G
ROSS_AMT,TXN_CURRENCY_CD,PAYMENT_TERMS,TAX_ACCRUAL_PMNT,VENDOR_LOC,REMIT_ADDR_SEQ_NU
M,VENDOR_ADDR_SEQ_NUM,INVOICE_TYPE,IS_PRICE_ADJUSTMENT,REF_INVOICE_ID

"PAYTue Aug 01 03:27:42 2006-18","IRTue Aug 01 03:27:42 2006-18","US003","Tue
Aug 01 03:27:42 2006","08/01/2006 00:00:00 -0700","SHARE","USA0000006","24.9500",
"USD","2D","No","HQ","sid496-an","1",,,
PAYTue Aug 01 05:34:04 2006-21","IRTue Aug 01 05:34:04
2006-21","US003","Tue Aug 01 05:34:04 2006","08/01/2006 00:00:00
-0700","SHARE","USA0000064","7","6.6400","CAD","00","No","HQ","sid815-hq","1",,,
```

Payment Detail Data in the PeopleSoft Variant

`PaymentDetailExport.csv` provides line level details for invoices.

The following table describes the columns in `PaymentDetailExport.csv`:

Column	Description	Format
`PMNT_ID`	The unique reference ID of the payment. This value matches a payment defined in `PaymentHeaderExport.csv`.	String (50)
`ECQUEUEINSTANCE`	The unique name of the electronic document in your ERP system.	String (50)
`BUSINESS_UNIT`	The PeopleSoft business unit.	String (50)
`VOUCHER_LINE_NUM`	The line number on the voucher.	Numeric
`PARENT_LINE_NUM`	The voucher line number of the parent for a charge line on the invoice.	Numeric
`LINE_TYPE_LOOKUP_CODE`	The line category. Line categories separate procurement line items from the various special handling categories, such as tax and freight. The line categories are as follows: • `1` - Line Item (for all line items, including non-catalog, customer catalog and PunchOut) • `2` - Tax Charge (for all tax-related charges) • `4` - Freight Charge (for all freight-related charges) • `8` - Handling Charge (for all handling-related charges) • `32` - Discount (for discounts related to invoicing) • `64` - Withholding Tax	Numeric
`TAX_CODE`	The line item tax code. SAP Ariba Buying and Invoicing initially sets the tax code based on invoice information and the tax code it obtains from the tax code lookup table. However, if a business user modifies the tax code during invoice reconciliation, the modified tax code, not the initial tax code, appears in this column. In sites using enhanced third-party tax calculation for invoices, this is the tax code from the posting call.	String (25)
`TAX_TYPE`	The tax type, for example, `SUTTax` (Sales/Use tax).	String (50)
`TAX_ACCRUAL`	A value on the invoice tax line that indicates a tax accrual line. • If `TAX_ACCRUAL` is No and the `MERCHANDISE_AMT` for the tax line is 0 or greater, the supplier is paid for the tax amount on the invoice and taxes are not accrued in PeopleSoft. • If `TAX_ACCRUAL` is Yes and the `MERCHANDISE_AMT` for the tax line is 0, the supplier is not paid for the tax amount (even though the tax appears on the invoice). and the complete tax amount as expected by the system is accrued in PeopleSoft. • If `TAX_ACCRUAL` is Yes and the `MERCHANDISE_AMT` for the tax line is greater than 0, a partial accrual is indicated. In this case, the supplier is paid for the tax amount on the invoice and the remaining tax amount (the amount expected by the system less the amount charged by the supplier) is accrued in PeopleSoft.	Boolean

Column	Description	Format
DESCR	A description of the invoice line item.	String (2000)
MERCHANDISE_AMT	For a line item, the value in this column is the amount charged for the commodity. For a tax line item, the value in this column is the tax amount to pay to the supplier. This column can also include the values of withholding taxes.	Numeric
UNIT_PRICE	The unit price of the merchandise.	Numeric
CURRENCY_CD	The currency code for the invoice line item amount.	String (50)
PRICEBASISQUANTITY	The quantity corresponding to the unit price of the item specified by the supplier. The unit price is based on the price unit quantity of the item.	Big Decimal
CONVERSIONFACTOR	The ratio used to convert the ordered unit to the price unit. The unit price is based on the price unit quantity and the price unit.	Big Decimal
PRICEBASISQUANTITYDESC	Any information for the advanced pricing details.	String (2000)
PRICEBASISQUANTITYUOM	The unit of measure for the unit price.	String (50)

Following is an example of `PaymentDetailExport.csv`:

```
PMNT_ID,ECQUEUEINSTANCE,BUSINESS_UNIT,VOUCHER_LINE_NUM,PARENT_LINE_NUM,LINE_TYPE_LOO
KUP_CODE,
TAX_CODE,TAX_TYPE,TAX_ACCRUAL,DESCR,MERCHANDISE_AMT,UNIT_PRICE,CURRENCY_CD,PRICEBASI
SQUANTITY,PRICEBASISQUANTITYUOM,CONVERSIONFACTOR,PRICEBASISQUANTITYDESC

PAYNPOINv-67, IRNPOINv-67, US003, 1, , 1, CA, , No, Composite Monitor Ext. RCA M/F,
6950.0000, 6.9500, USD,"2.0000","GAL","1","Gallon price"
PAYNPOINv-67, IRNPOINv-67, US003, 2, 1, 2, CA, SUTTax, No, Sales/Use tax (auto-
generated), 0.0000, 0.0000, USDPayment Bank Data
```

Payment Bank Data in the PeopleSoft Variant

`PaymentBankExport.csv` contains information about bank data associated with invoices.

The following table describes the columns in `PaymentBankExport.csv`:

Column	Description	Format
PMNT_ID	The unique reference ID of the payment. This value matches a payment defined in `PaymentHeaderExport.csv`.	String (50)
ECQUEUEINSTANCE	The unique name of the electronic document in your ERP system.	String (50)
BUSINESS_UNIT	The PeopleSoft business unit.	String (50)
PYMNT_METHOD	The unique internal identifier of the payment method. A payment method defines how a buying organization makes a payment to a supplier.	String (50)
BANK_ACCT_KEY	The bank account key.	String (50)

Column	Description	Format
COMPOSITE_ACCT_KEY	The unique internal identifier of the SAP Ariba Buying and Invoicing buyer bank payment location.	String (50)
SCHEDULED_PAY_DT	The date of the scheduled payment.	Date
BANK_CD	The unique code number of the bank.	String (25)
PAYMENT_AMT	The payment amount.	Numeric
PAYMENT_GROSS_AMT	The payment gross amount.	Numeric
PAYMENT_DISC_AMT	The payment discount amount.	Numeric
PAYMENT_CURRENCY_CD	The transaction currency code for the payment.	String (50)

Following is an example of `PaymentBankExport`:

```
     PMNT_ID,ECQUEUEINSTANCE,BUSINESS_UNIT,PYMNT_METHOD,BANK_ACCT_KEY,
COMPOSITE_ACCT_KEY,
SCHEDULED_PAY_DT,BANK_CD,PAYMENT_AMT,PAYMENT_GROSS_AMT,PAYMENT_DISC_AMT,PAYMENT_CURR
ENCY_CD

PAYNPOINv-67, IRNPOINv-67, US003, ach, chec, BuyerBankLocation-1, 04/24/2006
12:43:48 -0700, 77777, 6950.0000, 6950.0000, 0.0000, USD
```

Payment Split Accounting Data in the PeopleSoft Variant

`PaymentSplitAccountingExport.csv` contains information about split accounting associated with an invoice.

The following table describes the columns in `PaymentSplitAccountingExport.csv`:

Column	Description	Format
PMNT_ID	The unique reference ID of the payment. This value matches a payment defined in `PaymentHeaderExport.csv`.	String (50)
ECQUEUEINSTANCE	The unique name of the electronic document in your ERP system.	String (50)
BUSINESS_UNIT	The PeopleSoft business unit.	String (50)
VOUCHER_LINE_NUM	The line number of the voucher.	Numeric
DISTRIB_LINE_NUM	The distribution line number.	Numeric
BUSINESS_UNIT_GL	The PeopleSoft general ledger business unit.	String (50)
ACCOUNT	The account ID.	String (20)
DEPTID	The PeopleSoft department ID.	String (20)
QTY_VCHR	The quantity of the invoice line item.	Numeric
MERCHANDISE_AMT	The currency amount of the merchandise.	Numeric

Following is an example of `PaymentSplitAccountingExport.csv`:

```
PMNT_ID,ECQUEUEINSTANCE,BUSINESS_UNIT,VOUCHER_LINE_NUM,DISTRIB_LINE_NUM,BUSINESS_UNI
T_GL,ACCOUNT,DEPTID,QTY_VCHR,MERCHANDISE_AMT

PAYNPOINv-67, IRNPOINv-67, US003, 1, 1, USBC3, 156000, 105000, 1000.0000, 6950.0000
PAYNPOINv-67, IRNPOINv-67, US003, 2, 1, USBC3, 156000, 105000, 1.0000, 0.0000
```

Advance Payment Detail Data in the PeopleSoft Variant

`AdvancePaymentDetailExport.csv` contains the details of the advance payments adjusted against the invoice for which the payment request is generated.

The following table describes the columns in the `AdvancePaymentDetailExport.csv` file.

Column	Description	Format
`PMNT_ID`	The unique identifier of the payment request created for the invoice	String (50)
`PAY101ECQUEUEINSTANCE`	The unique identifier of the invoice reconciliation document for the invoice	String (50)
`ORDER_ID`	The unique identifier of the purchase order associated with the invoice and advance payment documents	String (50)
`ADVANCEPAYMENT_ID`	The unique identifier of the advance payment adjusted against the invoice	String (50)
`ERPADVANCEPAYMENT_ID`	The unique identifier of the payment request created in the ERP for the advance payment	String (50)
`CONSUMEDAMOUNT_AMOUNT`	The advance payment amount adjusted against the invoice, in the user's currency	Numeric
`CONSUMEDAMOUNT_CURRENCY`	The unique name of the invoice currency	String (50)
`CONSUMEDAMOUNT_INBASECURRENCY`	The advance payment amount adjusted against the invoice, in the site base currency	Numeric

Following is an example of `AdvancePaymentDetailExport.csv`:

```
ADVANCEPAYMENT_ID,ECQUEUEINSTANCE,PMNT_ID,ORDER_ID,ERPADVANCEPAYMENT_ID,CONSUMEDAMOU
NT_AMOUNT,CONSUMEDAMOUNT_CURRENCY,CONSUMEDAMOUNT_INBASECURRENCY
"APMT10","INV-IR101","PAY101","PO101","ERPAPMT10",10.00,"USD",10.00
```

SAP Payment Request Data

SAP Ariba Buying and Invoicing supports export of payment requests to the SAP ERP in real time (triggered when the transactions are ready to be sent to the ERP) using the Web Services channel and SAP Netweaver.

> **Note**
>
> SAP Ariba Buying and Invoicing exports payment requests that use the External System payment model only, and does not export partial payments, even if you import a partial remittance from your SAP system.

This section describes the SAP data that Ariba Administrator exports when you run the Export Payment Requests data export task. The data is stored in a ZIP file containing the following CSV files:

- `PaymentHeaderExport.csv`
- `PaymentLineItemDet.csv`
- `PaymentAccountDet.csv`
- `PaymentDetailExport.csv`
- `PaymentTaxExport.csv`
- `AdvancePaymentDetailExport.csv`

You can run the SAP function module `ZARIBA_SSP_INVOICELOAD` to transfer payment information from CSV files into the SAP database.

Payment Header Data in the SAP Variant

`PaymentHeaderExport.csv` contains general information about payments.

The following table describes the columns in `PaymentHeaderExport.csv`:

Column	Description	Format
`PAYMENTID`	The unique reference ID of the payment.	String (50)
`REF_DOC_NO`	The unique name of the invoice. Invoice numbers exceeding 16 characters are truncated to 16 characters during export.	String (16)
`COMP_CODE`	The unique internal identifier of the SAP company code.	String (4)
`GROSS_AMOUNT`	The invoice's gross amount to be paid to the supplier. For credit memos and negative amounts, this value is positive.	Numeric
`CURRENCY`	The payment amount currency code.	String (3)
`DOC_DATE`	The invoice date.	Date
`PMNTTRMS`	The unique internal identifier of the payment terms selected for the invoice. Payment terms specify the time period in which a supplier requests payment and any discounts that the supplier chooses to make available for early payment.	String (4)
`DIFF_INV`	The invoice's vendor ID.	String (50)
`INVOICE_IND`	The invoice ID.	String (50)

Column	Description	Format
PYMT_METH	The unique internal identifier of the payment method. A payment method defines how a buying organization makes a payment to a supplier.	String (50)
ITEM_TEXT	The unique name of the invoice reconciliation document.	String (50)
BLINE_DATE	The invoice baseline date.	String (50)
URL	A direct action URL. When external users click this URL, they are authenticated and taken directly to the invoice reconciliation document in SAP Ariba Buying and Invoicing.	String (100)
ACCRUAL_PMNT_AMT	The accrual payment amount determined by SAP Ariba Buying and Invoicing to be paid separately.	Numeric
INVOICE_TYPE	The type, or purpose, of the invoice. For price adjustments, indicates line-item credit memo or line-item debit memo.	String
IS_PRICE_ADJUSTMENT	Indicates whether the invoice is a price adjustment. Applies to line-item credit memos and line-item debit memos only.	Boolean
REF_INVOICE_NUM	Reference number of the original invoice line. Applies to line-item credit memos and line-item debit memos only.	String

Following is an example of `PaymentHeaderExport.csv`:

```
PAYMENTID,REF_DOC_NO,COMP_CODE,GROSS_AMOUNT,CURRENCY,DOC_DATE,PAYMENT_TERMS,DIFF_INV
,INVOICE_IND,PYMT_METH,ITEM_TEXT,BLINE_DATE,URL,ACCRUAL_PMNT_AMT,INVOICE_TYPE,IS_PRI
CE_ADJUSTMENT,REF_INVOICE_NUM

PAYPO10Invoice-LineLevel-10,PO10Invoice-
Line,3000,8920,USD,20060726,NT60,1000,X,IRPO10Invoice-LineLevel-10,20060726,https://
sampleurl,125,,,
```

Payment Line Item Data in the SAP Variant

`PaymentLineItemDet.csv` provides line level details for invoices.

The following table describes the columns in `PaymentLineItemDet.csv`:

Column	Description	Format
PAYMENTID	The unique reference ID of the payment.	String (50)
TAX_CODE	The unique tax type code.	String (50)
POLINERECEIVINGTYPE	The purchase order receiving type.	String (50)
ITEM_AMOUNT	The amount for each line item. For credit memos and negative line item amounts, this value is negative.	Numeric
QUANTITY	The quantity for each line item.	Numeric
PO_UNIT	The unique unit of measure value.	String (50)

Column	Description	Format
`INVOICE_DOC_ITEM`	The line number of every line item.	Numeric
`PO_NUMBER`	The unique ID of every purchase order for the ERP it belongs to.	String (50)
`PO_ITEM`	The line number of the corresponding line item.	String (50)
`ITEM_TEXT`	The line item description.	String (2000)
`LINE_TYPE`	The unique internal identifier of the line type of the line item. This field holds the tax type for indirect and withholding tax lines. For charges, this field holds the values `FreightCharge`, `HandlingCharge`, or `Charge`. For discount lines, this field holds the value `Discount`. For other line items, this field holds the values `CatalogItem` or `NonCatalogItem`.	String (50)
`CATEGORY`	The category of each line item.	Numeric
`ORDER`	The purchase order number in SAP.	String (50)
PRICEBASISQUANTITY	The quantity corresponding to the unit price of the item specified by the supplier. The unit price is based on the price unit quantity of the item.	Big Decimal
CONVERSIONFACTOR	The ratio used to convert the ordered unit to the price unit. The unit price is based on the price unit quantity and the price unit.	Big Decimal
PRICEBASISQUANTITY-DESC	Any information for the advanced pricing details.	String (2000)
PRICEBASISQUANTI-TYUOM	The unit of measure for the unit price.	String (50)

Following is an example of `PaymentLineItemDet.csv`:

```
PAYMENTID,TAX_CODE,POLINERECEIVINGTYPE,ITEM_AMOUNT,QUANTITY,PO_UNIT,INVOICE_DOC_ITEM
,PO_NUMBER,
PO_ITEM,ITEM_TEXT,CATEGORY,ORDER,


     PRICEBASISQUANTITY,PRICEBASISQUANTITYUOM,CONVERSIONFACTOR,



PRICEBASISQUANTITYDESC

PAYsap01-20,I1,2,25.79,3,EA,1,4700000860,10,Sales tax (auto-
generated),1,EPCK24,"2.0000","GAL","1","Gallon price"
```

> **Note**
>
> Adding a custom flex field to an IR results in pre-pending the label **InvoiceReconciliation_LineItems_** to the custom field label when exported to `PaymentLineItemDet.csv`. For example, if you create a column named **cat.IRLI.inv2343,** the resulting column heading is displayed as **InvoiceReconciliation_LineItems_cus_catIRLIinv2343**.

Payment Account Data in the SAP Variant

`PaymentAccountDet.csv` contains line item payment information.

The following table describes the columns in `PaymentAccountDet.csv`:

Column	Description	Format
`ITEM_AMOUNT`	The amount for each line item. For credit memos and negative line item amounts, this value is negative.	Numeric
`WBS_ELEM`	The unique code of the WBS element account assignment category for that line item.	String (50)
`ASSET_NO`	The unique value of the asset account assignment category for each line item.	String (50)
`COSTCENTER`	The unique value of the cost center account assignment category for each line item.	String (50)
`POLINERECEIVINGTYPE`	The purchase order receiving type.	String (50)
`QUANTITY`	The quantity for each line item.	Numeric
`PO_UNIT`	The unique unit of measure value.	String (50)
`SERIAL_NO`	The serial number of each line item.	Numeric
`INVOICE_DOC_ITEM`	The line number of each line item.	Numeric
`ORDERID`	The unique value of the internal order account assignment category for the line item.	String (50)
`GL_ACCOUNT`	The unique value of the general ledger accounting type for each line item.	Numeric
`LINE_TYPE`	The unique internal identifier of the line type of the line item. This field holds the tax type for indirect and withholding tax lines. For charges, this field holds the values `FreightCharge`, `HandlingCharge`, or `Charge`. For discount lines, this field holds the value `Discount`. For other line items, this field holds the values `CatalogItem` or `NonCatalogItem`.	String (50)
`TAX_CODE`	The unique tax type code.	String (50)
`PAYMENTID`	The unique ID of each payment that has a status displayed.	String (50)
`CATEGORY`	The category of every line item.	Numeric
`ORDER`	An SAP purchase order number.	String (50)
`PO_NUMBER`	The unique ID of each purchase order.	String (50)
`ACTN`	The unique value of the network account assignment category for each line item.	String (50)
`ACTY`	The unique value of the activity number account assignment category for each line item.	Numeric

Following is an example of `PaymentAccountDet.csv`:

```
ITEM_AMOUNT,WBS_ELEM,ASSET_NO,COSTCENTER,POLINERECEIVINGTYPE,QUANTITY,PO_UNIT,SERIAL
_NO,
INVOICE_DOC_ITEM,ORDERID,GL_ACCOUNT,TAX_CODE,PAYMENTID,CATEGORY,ORDER,PO_NUMBER,ACTN
,ACTY
```

```
12.50,3-1200/31,1005,2100,2,3,EA,1,1,400046,404000,I0,PAYsap01-20,1,EPCK24,470000086
0,Network1,4001
```

Payment Detail Data in the SAP Variant

`PaymentDetailExport.csv` provides line level details for invoices.

The following table describes the columns in `PaymentDetailExport.csv`:

Column	Description	Format
PAYMENTID	The unique reference ID of the payment. This value matches a payment defined in `PaymentHeaderExport.csv`.	String (50)
ITEM_AMOUNT	The item amount. For credit memos and negative line item amounts, this value is positive. This column is also applicable for withholding taxes.	Numeric
WBS_ELEM	The unique internal identifier of the SAP work breakdown structure (WBS) element.	String (50)
ASSET_NO	The unique internal identifier of the SAP asset number.	String (50)
ITEM_AMOUNT_ROUNDED	The amount for the item in ERP precision.	BigDecimal
COSTCENTER	The unique internal identifier of the SAP cost center.	String (50)
INVOICE_DOC_ITEM	The invoice line number.	Numeric
ORDERID	The internal order ID.	String (50)
GL_ACCOUNT	The unique internal identifier of the SAP general ledger account.	String (50)
COMP_CODE	The unique internal identifier of the SAP company code.	String (50)
LINE_TYPE	The unique internal identifier of the line type of the line item. This field holds the tax type for indirect and withholding tax lines. For charges, this field holds the values `FreightCharge`, `HandlingCharge`, or `Charge`. For discount lines, this field holds the value `Discount`. For other line items, this field holds the values `CatalogItem` or `NonCatalogItem`.	String (50)
TAX_CODE	The line item tax code. SAP Ariba Buying and Invoicing initially sets the tax code based on invoice information and the tax code it obtains from the tax code lookup table. However, if a business user modifies the tax code during invoice reconciliation, the modified tax code, not the initial tax code, appears in this column.	String (50)
PLANT	The unique internal identifier of the SAP plant, which is the ship-to address.	String (50)
ITEM_TEXT	The item's description.	String (2000)
DB_CR_IND	A value that specifies whether the item is a debit or a credit as follows: • S - debit • H - credit	String (10)

Column	Description	Format
LINE_TYPE_LOOKUP_CODE	The line category. Line categories separate procurement line items from the various special handling categories, such as tax and freight. The line categories are as follows: • 1 - Line Item (for all line items, including non-catalog, customer catalog and PunchOut) • 2 - Tax Charge (for all tax-related charges) • 4 - Freight Charge (for all freight-related charges) • 8 - Handling Charge (for all handling-related charges) • 32 - Discount (for discounts related to invoicing) • 64 - Withholding Tax	Numeric
HDR_LEVEL	A value that specifies whether the item is a header-level or line-level item as follows: • Yes - header level • No - line level	Boolean
PRICEBASISQUANTITY	The quantity corresponding to the unit price of the item specified by the supplier. The unit price is based on the price unit quantity of the item.	Big Decimal
CONVERSIONFACTOR	The ratio used to convert the ordered unit to the price unit. The unit price is based on the price unit quantity and the price unit.	Big Decimal
PRICEBASISQUANTITYDESC	Any information for the advanced pricing details.	String (2000)
PRICEBASISQUANTITYUOM	The unit of measure for the unit price.	String (50)
ACTN	The unique value of the network account assignment category for each line item.	String (50)
ACTY	The unique value of the activity number account assignment category for each line item.	Numeric

Following is an example of `PaymentDetailExport.csv`:

```
PAYMENTID,ITEM_AMOUNT,WBS_ELEM,ASSET_NO,COSTCENTER,INVOICE_DOC_ITEM,ORDERID,
GL_ACCOUNT,COMP_CODE,TAX_CODE,PLANT,ITEM_TEXT,DB_CR_

IND,LINE_TYPE_LOOKUP_CODE,HDR_LEVEL,PRICEBASISQUANTITY,PRICEBASISQUANTITYUOM,CONVERS
IONFACTOR,PRICEBASISQUANTITYDESC,ACTN,ACTY

PAYPO10Invoice-LineLevel-10,8800,,,2100,1,,445000,3000,I1,3000,Adapter SUN Monitor
4-BNCF/13W3M,S,1,No,"2.0000","GAL","1","Gallon price",Network1,4001
```

Payment Tax Export Data in the SAP Variant

`PaymentTaxExport.csv` contains tax data about an invoice.

The following table describes the columns in `PaymentTaxExport.csv`:

Column	Description	Format
`PAYMENTID`	The payment's unique ID. This value matches a payment defined in `PaymentHeaderExport.csv`.	String (50)
`TAX_AMOUNT`	The invoice line item tax amount. For credit memos and negative line item amounts, this value is positive.	Numeric
`TAX_CODE`	The line item tax code. SAP Ariba Buying and Invoicing initially sets the tax code based on invoice information and the tax code it obtains from the tax code lookup table. However, if a business user modifies the tax code during invoice reconciliation, the modified tax code, not the initial tax code, appears in this column. In sites using enhanced third-party tax calculation for invoices, this is the tax code from the posting call.	String (50)
`REF_LINE_NUM`	The reference line number on the invoice. For header taxes, this is the item line. For line-level taxes, this is the actual tax line.	Numeric
`TAX_AMOUNT_ROUNDED`	The amount for the tax item in ERP precision.	BigDecimal
`ITEM_TEXT`	The tax line description.	String (255)
`DB_CR_IND`	The indicator for debit or credit.	String (10)

Following is an example of `PaymentTaxExport.csv`:

```
     PAYMENTID,TAX_AMOUNT,TAX_CODE,REF_LINE_NUM,ITEM_TEXT
PAYPO10Invoice-LineLevel-10,30,I1,2,Sales tax - line level
```

Export of Header Tax When Invoice Line Amount Is Zero

By default, header tax isn't exported in the `PaymentTaxExport.csv` file if the invoice line amount is zero.

To have the invoice header tax exported even when all line items have a zero amount, have your Designated Support Contact (DSC) log a Service Request (SR). SAP Ariba Support will follow up by enabling feature SINV-11089, **Export header tax amount in SAP variant even when all lines have zero amount**.

- When feature SINV-11089 is enabled, if an invoice has a header tax, and all line items have a zero amount, the header tax is still exported. In the `PaymentTaxExport.csv` file, the header tax is applied to the first line.
- If feature SINV-11089 is disabled, the header tax is not exported if all line items have a zero amount.

Advance Payment Detail Data in the SAP Variant

`AdvancePaymentDetailExport.csv` contains the details of the advance payments adjusted against the invoice for which the payment request is generated.

The following table describes the columns in the `AdvancePaymentDetailExport.csv` file.

Column	Description	Format
PAYMENTID	The unique identifier of the payment request created for the invoice	String (50)
INVOICERECONCILIATIONID	The unique identifier of the invoice reconciliation document for the invoice	String (50)
ORDERID	The unique identifier of the purchase order associated with the invoice and advance payment documents	String (50)
ADVANCEPAYMENTID	The unique identifier of the advance payment adjusted against the invoice	String (50)
ERPADVANCEPAYMENTID	The unique identifier of the payment request created in the ERP for the advance payment	String (50)
CONSUMEDAMOUNT_AMOUNT	The advance payment amount adjusted against the invoice, in the user's currency	Numeric
CONSUMEDAMOUNT_CURRENCY	The unique name of the invoice currency	String (50)
CONSUMEDAMOUNT_INBASECURRENCY	The advance payment amount adjusted against the invoice, in the site base currency	Numeric

Following is an example of `AdvancePaymentDetailExport.csv`:

```
ADVANCEPAYMENTID,INVOICERECONCILIATIONID,PAYMENTID,ORDERID,ERPADVANCEPAYMENTID,CONSU
MEDAMOUNT_AMOUNT,CONSUMEDAMOUNT_CURRENCY,CONSUMEDAMOUNT_INBASECURRENCY
"APMT10","INV-IR101","PAY101","PO101","ERPAPMT10",10.00,"USD",10.00
```

Generic Payment Request Data

Ariba Administrator exports Generic data when you run the `Export Payment Requests` or `Export Payment Requests in 3 File Format` data export tasks.

The `Export Payment Requests` data export task exports a ZIP file containing the following CSV files:

- `PaymentHeaderExport.csv`
- `PaymentBankExport.csv`
- `PaymentDetailExport.csv`
- `PaymentSplitAccountingExport.csv`
- `PaymentTaxExport.csv`
- `AdvancePaymentDetailExport.csv`

The `Export Payment Requests in 3 File Format` data export task exports a ZIP file containing the following CSV files:

- `PaymentHeaderExport.csv`
- `PaymentBankExport.csv`
- `PaymentDetailWithSplitAccountingExport.csv`

Payment Header Data in the Generic Variant

`PaymentHeaderExport.csv` is exported by both `Export Payment Requests` and `Export Payment Requests in 3 File Format`.

The following table describes the columns in `PaymentHeaderExport.csv`:

Column	Description	Format
`PaymentId`	The payment's unique reference ID.	String (50)
`InvoiceReconciliationId`	The invoice reconciliation document ID.	String (50)
`PurchasingUnit`	The unique internal identifier of the purchasing unit.	String (50)
`InvoiceNumber`	The invoice number.	String (50)
`InvoiceDate`	The invoice date.	Date
`Supplier`	The unique internal identifier of the supplier.	String (50)
`GrossAmount`	The gross invoice amount.	Numeric
`GrossAmountCurrency`	The invoice transaction currency.	String (50)
`PaymentTerms`	The unique internal identifier of the payment terms selected for the invoice. Payment terms specify the time period in which a supplier requests payment and any discounts that the supplier chooses to make available for early payment.	String (50)
`ForTaxAccrual`	A value that specifies whether the payment is an accrual payment.	Boolean
`SupplierLocation`	The unique internal identifier of the supplier location.	String (50)
`SupplierLocationContactID`	The contact ID of the contact person at the supplier location specified in the `SupplierLocation` column.	String (50)
`RemittanceLocation`	The unique internal identifier of the remittance location. When sending a payment to a given supplier location, SAP Ariba Buying and Invoicing sends to the remittance location associated with that supplier location.	String (50)
`InvoiceType`	The type, or purpose, of the invoice. For price adjustments, indicates line-item credit memo or line-item debit memo.	String
`IsPriceAdjustmentInvoice`	Indicates whether the invoice is a price adjustment. Applies to line-item credit memos and line-item debit memos only.	Boolean
`ReferencedInvoiceNumber`	Reference number of the original invoice line. Applies to line-item credit memos and line-item debit memos only.	String

Following is an example of `PaymentHeaderExport.csv`:

```
PaymentId,InvoiceReconciliationId,PurchasingUnit,InvoiceNumber,InvoiceDate,Supplier,
GrossAmount,GrossAmountCurrency,PaymentTerms,ForTaxAccrual,SupplierLocation,Supplier
LocationContactID,RemittanceLocation,InvoiceType,IsPriceAdjustmentInvoice,Referenced
InvoiceNumber

"PAYSUPPLIER_INVOICE_NUM-11","IRSUPPLIER_INVOICE_NUM-11","US005","SUPPLIER_INVOICE_N
UM","07/22/2006 00:00:00
-0700","prid515","29.9700","USD","PT1","No","10151,63,RL515",,,
```

Payment Bank Export Data in the Generic Variant

`PaymentBankExport.csv` is exported by both `Export Payment Requests` and `Export Payment Requests in 3 File Format`.

The following table describes the columns in `PaymentBankExport.csv`:

Column	Description	Format
`PaymentId`	The payment's unique reference ID. This value matches a payment defined in `PaymentHeaderExport.csv`.	String (50)
`InvoiceReconciliationId`	The invoice reconciliation document ID.	String (50)
`PurchasingUnit`	The unique internal identifier of the purchasing unit. Purchasing units keep different sets of data separate.	String (50)
`PaymentMethod`	The unique internal identifier of the payment method. A payment method defines how a buying organization makes a payment to a supplier.	String (50)
`BankAccountType`	The unique internal identifier of the bank account type, such as `checking` or `savings`.	String (50)
`BuyerBankLocation`	The unique internal identifier of the SAP Ariba Buying and Invoicing buyer bank payment location.	String (50)
`ScheduledPaymentDate`	The date of the scheduled payment.	Date
`BankId`	The bank identifier.	String (25)
`Amount`	The payment amount.	Numeric
`GrossAmount`	The payment gross amount.	Numeric
`DiscountAmount`	The discount amount.	Numeric
`AmountCurrency`	The transaction currency code for the payment.	String (50)

Following is an example of `PaymentBankExport.csv`:

```
PaymentId,InvoiceReconciliationId,PurchasingUnit,PaymentMethod,BankAccountType,Buyer
BankLocation,ScheduledPaymentDate,BankId,Amount,GrossAmount,DiscountAmount,AmountCur
rency

"PAYSUPPLIER_INVOICE_NUM-11","IRSUPPLIER_INVOICE_NUM-11","US005","check","checking",
"BuyerBankLocation-1","08/11/2006 11:18:47
-0700","TRBNK","29.9700","29.9700","0.3891","USD"
```

Payment Detail Data in the Generic Variant

`PaymentDetailExport.csv` is exported only by the `Export Payment Requests` data export task. It provides line item payment details.

The following table describes the columns in `PaymentDetailExport.csv`:

Column	Description	Format
`PaymentId`	The payment's unique reference ID. This value matches a payment defined in `PaymentHeaderExport.csv`.	String (50)
`InvoiceReconciliationId`	The invoice reconciliation document ID.	String (50)
`PurchasingUnit`	The unique internal identifier of the purchasing unit. Purchasing units keep different sets of data separate.	String (50)
`LineNumber`	The line number in the voucher.	Numeric
`ParentLineNumber`	The voucher line number of the parent for a charge line on the invoice.	Numeric
`LineTypeCategory`	The line category. Line categories separate procurement line items from the various special handling categories, such as tax and freight. The line categories are as follows: • `1` - Line Item (for all line items, including non-catalog, customer catalog and PunchOut) • `2` - Tax Charge (for all tax-related charges) • `4` - Freight Charge (for all freight-related charges) • `8` - Handling Charge (for all handling-related charges) • `32` - Discount (for discounts related to invoicing) • `64` - Withholding Tax	String
`TaxCode`	The line item tax code. SAP Ariba Buying and Invoicing initially sets the tax code based on invoice information and the tax code it obtains from the tax code lookup table. However, if a business user modifies the tax code during invoice reconciliation, the modified tax code, not the initial tax code, appears in this column. In sites using enhanced third-party tax calculation for invoices, this is the tax code from the posting call.	String (50)
`TaxType`	The line item tax type, for example, `SalesTax`.	String (50)
`IsAccrual`	A value that indicates whether the line item is a tax accrual line item. `IsAccrual` is set to **Yes** if: • an 'Accrual Tax Variance' exception is disputed: The reconciler agrees to pay the supplier no tax and accrues the entire calculated tax amount to be paid to the government authority. • an 'Under Tax Variance' exception is disputed: The reconciler agrees to pay the supplier the charged tax amount and accrue the remaining difference (calculated – charged) to be paid to the government authority. • Reconcilers check the Accrual check box for a tax detail line during reconciliation.	Boolean

Column	Description	Format
`Description`	A line item description.	String (2000)
`Amount`	For a line item, the value in this column is the amount charged for the commodity. For a tax line item, the value in this column is the tax amount to pay to the supplier. **Note** This column is also applicable for withholding tax values.	Numeric
`UnitPrice`	For goods or services lines, `UnitPrice` represents the line item unit price. For tax lines that are disputed (but reconcilers have not checked the Accrual flag), `UnitPrice` contains the expected tax amount (the total tax amount for the reconciled line item, including both the tax amount paid to the supplier and the accrual amount to be paid separately to the government authority). For tax lines that have been manually flagged as accruals, `UnitPrice` is set to 0 (zero). For all other tax lines, `UnitPrice` contains the tax amount.	Numeric
`AmountCurrency`	The currency code for the line item amount.	String (50)
`ERPCommodityCode`	The ERP commodity code for the line item. This column is blank if your site does not use ERP commodity codes.	Numeric
`TaxAdjustmentAmount`	The accrued tax amount to be paid separately to the government authority. For tax line items with Under Tax Variance exceptions that are disputed, `TaxAdjustmentAmount` is calculated as the difference between `UnitPrice` (total expected tax amount) and `Amount` (tax amount to pay to supplier).	Numeric
ERPOrderID	The ERP purchase order number.	Numeric
ERPOrderLineNumber	The unique identifier of each line in an ERP purchase order	Numeric
PRICEBASISQUANTITY	The quantity corresponding to the unit price of the item specified by the supplier. The unit price is based on the price unit quantity of the item.	Big Decimal
CONVERSIONFACTOR	The ratio used to convert the ordered unit to the price unit. The unit price is based on the price unit quantity and the price unit.	Big Decimal
PRICEBASISQUANTITYDESC	Any information for the advanced pricing details.	String (2000)
PRICEBASISQUANTITYUOM	The unit of measure for the unit price.	String (50)

Following is an example of `PaymentDetailExport.csv`:

```
PaymentId,InvoiceReconciliationId,PurchasingUnit,LineNumber,ParentLineNumber,LineTyp
eCategory,
TaxCode,TaxType,IsAccrual,Description,Amount,UnitPrice,AmountCurrency,ERPCommodityCo
de,
TaxAdjustmentAmount,ERPOrderID,ERPOrderLineNumber,
```

```
,PriceBasisQuantity,PriceBasisQuantityUOM,ConversionFactor,PriceBasisQuantityDesc,




"PAYSUPPLIER_INVOICE_NUM-11","IRSUPPLIER_INVOICE_NUM-11","US005","1","","1","TC16","
","No",
"Maps & Globes, Map Measurer, Simply run it along the route to be measured. One
pride shows miles and kilometers. The other pride shows statute miles, kilometers
and nautical miles. Attached magnifier, zippered
case.","12.9700","12.9700","USD","","3.52","2.0000","GAL","1","Gallon price"
"PAYSUPPLIER_INVOICE_NUM-11","IRSUPPLIER_INVOICE_NUM-11","US005","2","1","2","TC16",
"SalesTax",
"No","Sales tax - line level","2.0000","2.0000","USD","","2.64",,
```

Payment Split Accounting Data in the Generic Variant

`PaymentSplitAccountingExport.csv` is only exported by the `Export Payment Requests` data export tasks. It contains information about split accounting data associated with invoices.

The following table describes the columns in `PaymentSplitAccountingExport.csv`:

Column	Description	Format
`PaymentId`	The payment's unique reference ID. This value matches a payment defined in `PaymentHeaderExport.csv`.	String (50)
`InvoiceReconciliationId`	The invoice reconciliation document ID.	String (50)
`PurchasingUnit`	The unique internal identifier of the purchasing unit.	String (50)
`LineNumber`	The line number of the voucher.	Numeric
`SplitLineNumber`	The split accounting line number.	Numeric
`Company`	The unique internal identifier of the company.	String (50)
`BusinessUnit`	The unique internal identifier of the business unit.	String (50)
`CostCenter`	The unique internal identifier for the cost center.	String (50)
`Account`	The unique internal identifier of the account, for example, `Expense`,	String (50)
`SubAccount`	The unique internal identifier of the subaccount.	String (50)
`Product`	The unique internal identifier of the product.	String (50)
`Project`	The unique internal identifier of the project.	String (50)
`Quantity`	The line item quantity.	Numeric
`Amount`	The line item amount.	Numeric
`Region`	The unique internal identifier of the accounting region.	String (50)
`AccountingCombinationID`	The unique internal identifier of an accounting combination defined in SAP Ariba Buying and Invoicing.	String (50)

Following is an example of `PaymentSplitAccountingExport.csv`:

```
PaymentId,InvoiceReconciliationId,PurchasingUnit,LineNumber,SplitLineNumber,Company,
BusinessUnit,CostCenter,Account,SubAccount,Product,Project,Quantity,Amount,Region,Ac
countingCombinationID

"PAY545-21","IR545-21","US005","1","1","GD","US005","3300","7720","1210",,,"5.0000",
"359.9500",,
"PAY545-21","IR545-21","US005","2","1","GD","US005","3300","7720","1210",,,"1.0000",
"0.0000"
"PAY24324-29","IR24324-29","US100","1","1","GD","US100","3100","7720","5008",,,"10.0
000","25.7000",,
```

Advance Payment Detail Data in the Generic Variant

`AdvancePaymentDetailExport.csv` contains the details of the advance payments adjusted against the invoice for which the payment request is generated.

The following table describes the columns in the `AdvancePaymentDetailExport.csv` file.

Column	Description	Format
`PaymentID`	The unique identifier of the payment request created for the invoice	String (50)
`InvoiceReconciliationID`	The unique identifier of the invoice reconciliation document for the invoice	String (50)
`OrderID`	The unique identifier of the purchase order associated with the invoice and advance payment documents	String (50)
`AdvancePaymentID`	The unique identifier of the advance payment adjusted against the invoice	String (50)
`ERPAdvancePaymentID`	The unique identifier of the payment request created in the ERP for the advance payment	String (50)
`ConsumedAmount_Amount`	The advance payment amount adjusted against the invoice, in the user's currency	Numeric
`ConsumedAmount_Currency`	The unique name of the invoice currency	String (50)
`ConsumedAmount_InBaseCurrency`	The advance payment amount adjusted against the invoice, in the site base currency	Numeric

Following is an example of `AdvancePaymentDetailExport.csv`:

```
AdvancePaymentID,InvoiceReconciliationID,PaymentID,OrderID,ERPAdvancePaymentID,Consu
medAmount_Amount,ConsumedAmount_Currency,ConsumedAmount_InBaseCurrency
"APMT10","INV-IR101","PAY101","PO101","ERPAPMT10",10.00,"USD",10.00
```

Payment Detail with Split Accounting Data in the Generic Variant

`PaymentDetailWithSplitAccountingExport.csv` is exported only by the `Export Payment Requests in 3 File Format` data export task.

The following table describes the columns in `PaymentDetailWithSplitAccountingExport.csv`:

Column	Description	Format
`PaymentId`	The payment's unique reference ID. This value matches a payment defined in `PaymentHeaderExport.csv`.	String (50)
`InvoiceReconciliationId`	The invoice reconciliation document ID.	String (50)
`PurchasingUnit`	The unique internal identifier of the purchasing unit. Purchasing units keep different sets of data separate.	String (50)
`LineNumber`	The line number of the voucher.	Numeric
`ParentLineNumber`	The voucher line number of the parent of the charge line on the invoice.	Numeric
`LineTypeCategory`	The line category. Line categories separate procurement line items from the various special handling categories, such as tax and freight. The line categories are as follows: • `1` - Line Item (for all line items, including non-catalog, customer catalog and PunchOut) • `2` - Tax Charge (for all tax-related charges) • `4` - Freight Charge (for all freight-related charges) • `8` - Handling Charge (for all handling-related charges) • `32` - Discount (for discounts related to invoicing) • `64` - Withholding Tax	String
`TaxCode`	The line item tax code. SAP Ariba Buying and Invoicing initially sets the tax code based on invoice information and the tax code it obtains from the tax code lookup table. However, if a business user modifies the tax code during invoice reconciliation, the modified tax code, not the initial tax code, appears in this column. For more information on the tax code lookup table, see Tax Data in SAP Ariba Procurement Solutions. For more information on invoice reconciliation, see Reconciling Invoices.	String (50)
`TaxType`	The line item tax type, for example, `SalesTax`.	String (50)
`IsAccrual`	A value (always `False`) that indicates whether the line item is a tax accrual line item.	Boolean
`Description`	A line item description.	String (2000)
`UnitPrice`	The line item unit price.	Numeric
`AmountCurrency`	The currency code for the line item amount.	String (50)
`SplitLineNumber`	The split accounting line number.	Numeric
`Company`	The unique internal identifier of the company.	String (50)

Column	Description	Format
`BusinessUnit`	The unique internal identifier of the business unit.	String (50)
`CostCenter`	The unique internal identifier for the cost center.	String (50)
`Account`	The unique internal identifier of the account, for example, `Expense`,	String (50)
`SubAccount`	The unique internal identifier of the subaccount. Subaccounts are accounting entities that are subordinate to the accounts.	String (50)
`Product`	The unique internal identifier of the product.	String (50)
`Project`	The unique internal identifier of the project.	String (50)
`Region`	The unique internal identifier of the accounting region.	String (50)
`AccountingCombinationID`	The unique internal identifier of an accounting combination defined in SAP Ariba Buying and Invoicing.	String (50)
`Quantity`	The line item quantity.	Numeric
`Amount`	The line item amount.	Numeric
PRICEBASISQUANTITY	The quantity corresponding to the unit price of the item specified by the supplier. The unit price is based on the price unit quantity of the item.	Big Decimal
CONVERSIONFACTOR	Ratio used to convert the ordered unit to the price unit. The unit price is based on the price unit quantity and the price unit.	Big Decimal
PRICEBASISQUANTITYDESC	Any information for advanced pricing details.	String (2000)
PRICEBASISQUANTITYUOM	The unit of measure for the unit price.	String (50)

Following is an example of `PaymentDetailwithSplitAccountingExport.csv`:

```
PaymentId,InvoiceReconciliationId,PurchasingUnit,LineNumber,ParentLineNumber,LineTyp
eCategory,
TaxCode,TaxType,IsAccrual,Description,UnitPrice,AmountCurrency,SplitLineNumber,Compa
ny,
BusinessUnit,CostCenter,Account,SubAccount,Product,Project,Region,AccountingCombinat
ionID,
Quantity,Amount,



,PriceBasisQuantity,PriceBasisQuantityUOM,ConversionFactor,PriceBasisQuantityDesc

"PAYtestgreaerinv-91","IRtestgreaerinv-91","US005","1",,"1","TC21",,"No","Enter a
description for this
item.","1.0000","USD","1","GD","US005","3300","7752","1210",,,,"261","150.0000","150
.0000","2.0000","GAL","1","Gallon price"
```

Amount Precision in Payment Request Export Files

During export, a configuration parameter controls the precision of amount values in payment request export files.

Amounts are rounded to the precision specified in the configuration parameter. If rounding of the amount fields in the split accounting records or line level records during payment request export results in a discrepancy with

the line level total or header level total respectively, then the amount values of the last line level or split accounting level are adjusted so that the total equals the line level or header level total based on the specified precision.

Example: Split amounts are 3.33333, 3.33333, 3.33334, and the line level amount is 10.00000. During export, with precision set to 3, the split amounts are exported as: 3.333, 3.333 and 3.334. If precision is set to 2, then the split amount values are: 3.33, 3.33 and 3.34.

The amount values in the exported CSV files are always exported with 4 decimal places regardless of the precision setting. If the precision is set to 3, amount values are exported as 1.2340. If the precision is set to 2, amount values are exported as 1.2300.

By default, the decimal amount precision is set as follows:

- For SAP and Generic sites: 4 decimal places
- For PeopleSoft sites: 3 decimal places

The maximum out-of-the-box supported precision is 4 decimal places.

Columns Affected by Amount Precision

The columns affected by amount precision depend on your ERP system.

PeopleSoft:

- `MERCHANDISE_AMT` in `PaymentDetailExport.csv`
- `GROSS_AMT` in `PaymentHeaderExport.csv`
- `AMOUNT` in `PaymentSplitAccountingExport.csv`
- `PAYMENT_AMT`, `PAYMENT_GROSS_AMT`, `PAYMENT_DISC_AMT` in `PaymentBankExport.csv`

Generic:

- `Amount` in `PaymentDetailExport.csv`
- `GrossAmount` in `PaymentHeaderExport.csv`
- `Amount` in `PaymentSplitAccountingExport.csv`
- `Amount`, `GrossAmount`, `DiscountAmount` in `PaymentBankExport.csv`

SAP:

Precision adjustment is not required in SAP integrated sites. SAP provides its own rounding handling.

> **Note**
>
> Contact SAP Ariba Customer Support if you need to change the amount precision to be able to import payment request files into an ERP system that cannot handle the default amount precision.

About Importing Remittances

Importing remittances applies to existing customers using the remittance import tasks in SAP Ariba Buying and Invoicing or SAP Ariba Invoice Management, or you are an existing or new customer and do not have all of your suppliers either enabled or configured for quick enablement on SAP Business Network.

The SAP Business Network payment integration toolkit only loads remittance information for invoices that are recognized on SAP Business Network. Therefore, it does not load remittance information for invoices that were created manually by a buyer user in SAP Ariba Buying and Invoicing for a supplier who is not enabled on SAP Business Network, since those invoices are not copied to the associated SAP Business Network buyer account. Therefore, you must use the remittance import tasks in SAP Ariba Buying and Invoicing.

Asynchronous remittance import applies to customers who have integrated their SAP Ariba Procurement solutions with their SAP back-end system through the SAP Integration Suite, managed gateway for spend management and SAP Business Network. Remittance information can be imported from SAP back-end systems using the new web service remittance import task. For more information, see About Importing Remittances Asynchronously [page 195].

Introduction to Remittance Information [page 191]

Import Remittances Data Import Task [page 193]

About Importing Remittances Asynchronously [page 195]

Introduction to Remittance Information

Remittances, also called remittance advice, are summary statements that provide details about payment transactions. For example, a remittance might list the invoices associated with the payment, the payment method used, bank information, discount information, and amount paid.

Remittance advice is imported to SAP Business Network using SAP Integration Suite, managed gateway for spend management and SAP Business Network from your external system, and copied to your SAP Ariba Invoice Management or SAP Ariba Buying and Invoicing solution.

Some older SAP Ariba sites might use the **Import Remittance** data import task. This option is deprecated and might be removed in a future release.

For advance payments, members of the **Customer Administrator** group can import remittance information using the **Import Advance Payment Remittance** task.

If remittance data is not successfully pulled from SAP Business Network or an external system, members of the **Payment Administrator** group receive an email notification indicating that there's a problem.

About Pulling Remittances from SAP ERP

You can run the SAP function module `ZARIBA_SSP_REMITTANCE_PULL` to transfer remittance transaction information from the SAP database into CSV files.

You can use the DB Connector to extract remittance transaction information from the following ERP system databases:

- Oracle
- DB2
- SQL Server

About Importing Remittances in SAP Ariba Buying and Invoicing

When importing remittance information from an external ERP system into SAP Ariba Buying and Invoicing, note the following:

- SAP Ariba Buying and Invoicing does not consider accrual payments in determining if an invoice is completely paid.
- Importing accrual remittances in SAP Ariba Buying and Invoicing is not supported.
- Payment documents generated in an SAP ERP don't specify a payment method by default. For SAP Ariba Buying and Invoicing to import remittance transaction data, specify a payment method value (for example, `check`) in the `Remittance.csv` file.

Remittance Locations

When the SAP Ariba solution generates a payment request, it uses the remittance location associated with the supplier location. You define remittance locations in SAP Ariba Buying and Invoicing and SAP Ariba Invoice Management. Each supplier location must have its own remittance location. You can't associate multiple supplier locations with the same remittance location, but you can associate multiple remittance locations with the same supplier location.

Additional References

Common Data Import and Administration for SAP Ariba Procurement Solutions

Related Information

Exporting Payment Requests in SAP Ariba Buying and Invoicing [page 165]
Remit To on invoice reconciliations is from invoice

Import Remittances Data Import Task

The `Import Remittances` data import task imports transactional remittance information from an external ERP system to your SAP Ariba procurement solution.

`Import Remittances` reads data from the following CSV files:

CSV File	Description
`Remittance.csv`	Defines the remittance information.
`RemittanceDetails.csv`	Defines the details of each remittance specified in `Remittance.csv`.

Following is an example of `Remittance.csv` for PeopleSoft and Generic:

```
UTF8


LookupID,SupplierLocation,SupplierLocationContactID,PurchasingUnit,TransactionType,C
HECK_ID,
PaymentNumber,PaymentDate,CanceledDate,NetAmount,NetAmountDefaultCurrency,PaymentMet
hodType,
ExternalProcessedState,CreatedDate,VENDOR_ID,SupplierRemitToAddressName,VENDOR_SITE_
CODE,
SupplierRemitToAddressStreet,SupplierRemitToAddressCity,SupplierRemitToAddressState,
SupplierRemitToAddressCountry,RemitToAddressPostalCode,BuyerBankID,BuyerBankAccountI
D,
BuyerBankAccountType,BuyerBankName,BUYER_BANK_BRANCH_NAME,BUYER_ROUTING_NUMBER,BUYER
_BANK_NUMBER,SupplierBankID,SupplierBankAccountID,SupplierBankName,SupplierBankAccou
ntType,
SUPPLIER_BANK_BRANCH_NAME,SUPPLIER_ROUTING_NUMBER,SUPPLIER_BANK_NUMBER,PAYMENT_UPDAT
E_DATE,
GrossAmount,AdjustmentAmount,DiscountAmount,GrossAmountDefaultCurrency,AdjustmentDef
aultCurrency,DiscountAmountDefaultCurrency,BuyerRemitFromAddressName,BuyerRemitFromA
ddressStreet,
BuyerRemitFromAddressCity,BuyerRemitFromAddressState,BuyerRemitAddressPostalCode,
BuyerRemitAddressCountry,BuyerBankIDType,SupplierBankIDType,SupplierBankAccountIDTyp
e,
BuyerBankAccountIDType,BatchID,NumberofPayments,AllPaymentsAmount

29155,4815,4820,600,1,29155,100062,6/23/2008 0:00,,1500,EUR,check,0,6/23/2008
21:52,4814,
JCN Technologies,Albany,"1000815 Main St.,,",NEW YORK,NY,US,10281,13641,560032,,
ICICI,ICICI-Bangalore,100000,60001,13682,12321400,CA-123214-JCN,,CA-123214-JCN-
BAN,,123214,
6/23/2008 21:52,1500,0,0,EUR,EUR,EUR,,,,,,,,,,Quick Payment: ID=29155,1,1500
```

Following is an example of `Remittance.csv` for SAP:

```
UTF8


LOOKUPID,COMP_CODE,PAYDOC,FISCALYEAR,VENDOR,CHECKNO,STAT,VERSION,POSTINGDATE,CREATIO
NDATE,
CURRENCY,PAYMENTMETHOD,NAME1,NAME2,NAME3,NAME4,STREET,CITY,STATE,COUNTRY,POSTALCODE,
POBOX,PSLT2,
PAYEEBANKNAME,PAYEEBANKACCNO,PAYEEBANKNO,PAYEESWIFT,PAYERBANKNAME,PAYERBANKACCNO,PAY
ERBANKNO,
```

```
NETAMOUNT

         PMT_77_IRSAP-
Invoice-03-22,3000,78,2006,500004,79,0,001,20060822,20060822,USD,C,RL6,,,,
300 10th Avenue,San Francisco,CA,US,94111,,,Bank of
America,10004912,121000358,abaRoutingNumber,
Citibank,30050021,134329042,109.04
         PMT_81_IRSAP-
Invoice-04-23,3000,82,2006,500004,83,0,000,20060822,20060822,USD,C,RL6,,,,
300 10th Avenue,San Francisco,CA,US,94111,,,Bank of
America,10004912,121000358,abaRoutingNumber,
Citibank,30050021,134329042,109.04
         PMT_85_IRAK-SAP-
John-01-24,3000,86,2006,1000,87,0,001,20060822,20060822,USD,A,RL3,,,,
Kolping Str. 15,Berlin,CA,US,12001,,,Mellon
Bank,10003230,238100235,abaRoutingNumber,
Citibank,30050021,134329042,66.4
         PMT_89_IRAK-
Credit-666-25,3000,90,2006,500004,91,0,001,20060822,20060822,USD,C,RL6,,,,
300 10th Avenue,San Francisco,CA,US,94111,,,Bank of
America,10004912,121000358,abaRoutingNumber,
Citibank,30050021,134329042,-20
```

Following is an example of `RemittanceDetails.csv` for PeopleSoft and Generic:

```
UTF8

     LookupID,NumberInCollection,PayableDate,PayableReferenceNumber,PayableType,
ExternalPayableReferenceNumber,SupplierPayableReferenceNumber,SecondaryPayableRefere
nceNumber,
SecondaryPayableType,ExternalSecondaryPayableReferenceNumber,GrossAmountDefaultCurre
ncy,
GrossAmount,DiscountAmountDefaultCurrency,DiscountAmount,AdjustmentAmountDefaultCurr
ency,
AdjustmentAmount,NetAmountDefaultCurrency,NetAmount

R6,1,9/12/2006,IR66666-12,1,MyRef1,INVNPO2-68,,2,,USD,42.75,USD,1.2825,USD,0,USD,41.
4675
```

Following is an example of `RemittanceDetails.csv` for SAP:

```
UTF8


LOOKUPID,COMP_CODE,PAYDOC,FISCALYEAR,DOC_NO,ITEM_NO,DOCDATE,REFDOC,PONUMBER,DEBITCRE
DIT,
GROSSAMOUNT,CASHDISCOUNT,NETAMOUNT,ITEMTEXT,CURRENCY

3000:1500000023:2008,3000,1500000023,2008,5100000840,1,20080401,INV_123,,H,100,0,100
,
IRINV_123-10,USD
```

Related Information

Introduction to Remittance Information [page 191]
About Importing Remittances Asynchronously [page 195]

About Importing Remittances Asynchronously

You can import remittance information asynchronously if you have integrated your SAP Ariba Procurement solutions with your SAP back-end system through the SAP Integration Suite, managed gateway for spend management and SAP Business Network.

You can import remittance information asynchronously using the new **Import Remittances** web service task in the **Integration Manager** workspace. This task can be configured for use with the SAP Integration Suite, managed gateway for spend management and SAP Business Network. The end point URL for the new web service task must be configured in the buying solution from SAP Ariba. Customer administrators must enter the URL in the **Integration Manager** page.

Prerequisites

Before using this feature, ensure the following:

- Support for the SAP Integration Suite, managed gateway for spend management and SAP Business Network is enabled in your solution.
- Support for asynchronous integration events for the web service channel in SAP Integration Suite, managed gateway for spend management and SAP Business Network is enabled in your solution.

About Importing and Exporting Advance Payment Data

Members of the **Payment Agent** and **Payment Manager** groups can create advance payments for purchase orders in SAP Ariba Buying and SAP Ariba Buying and Invoicing.

Though the advance payments are created in SAP Ariba Buying solutions, the actual payment for these happens in the external system using the External System payment model.

Advance payments can be exported to external systems using the file channel as well as the web services channel. Before using the web services channel, ensure that your site is configured to use web services for data integration.

The table below lists the data import and export tasks that enable you to import and export advance payment data to the external ERP system. The following table lists the tasks in the sequence in which they need to be run.

Task Name	CSV Files Required	Task Description	Supported Integration Channels
Import Document Types for Advance Payment	`DocumentType.csv`	Imports ERP document types that are used for advance payments. This task is required and available only for sites integrated with SAP ERP systems.	File channel
Import GL Indicators for Advance Payment	`GLIndicator.csv`	Imports ERP GL indicators that are used for advance payments. This task is required and available only for sites integrated with SAP ERP systems.	File channel
Import Translations for Advance Payment Document Types	`DocumentType.csv`	Imports translations for ERP document types that are used for advance payments. This task is required and available only for sites integrated with SAP ERP systems.	File channel
Import Translations for Advance Payment GL Indicators	`GLIndicator.csv`	Imports translations for ERP GL indicators that are used for advance payments. This task is required and available only for sites integrated with SAP ERP systems.	File channel
Export Advance Payments	`AdvancePaymentExport.csv` `AdvancePaymentLineItemExport.csv`	Exports approved advance payments for integration with external systems	File channel Web Services channel

Task Name	CSV Files Required	Task Description	Supported Integration Channels
Import Advance Payment ID	`AdvancePaymentIDImport.csv`	Imports IDs returned by the ERP after successful upload of advance payments from SAP Ariba Buying solutions	File channel Web Services channel: Internally invoked in response to the **Export Advance Payments** task
Import Advance Payment Errors	`AdvancePaymentErrorImport.csv`	Imports errors generated by the external system while uploading advance payment documents sent by the SAP Ariba Buying solutions	File channel Web Services channel: Internally invoked in response to the **Export Advance Payments** task
Import Advance Payment Remittance	`AdvancePaymentRemittance.csv` `AdvancePaymentRemittanceDetail.csv`	Imports remittance information for advance payment documents exported to external systems	File channel Web Services channel: Real-time import triggered by payments made in the external ERP system
Export Cancel Advance Payments	`CancelAdvancePaymentExport.csv`	Exports advance payments with status **Canceling** for integration with external systems	File channel Web Services channel
Import Advance Payment Cancellation Reasons	`AdvancePaymentExport.csv`	Imports a list of reasons for cancellation of advance payments.	File channel
Import Cancel Advance Payment ID	`CancelAdvancePaymentIDImport.csv`	Imports IDs of canceled advance payments returned by the external system after successful upload of canceled advance payments from SAP Ariba Buying solutions	File channel Web Services channel: Internally invoked in response to the **Export Cancel Advance Payments** task
Export ERP Document Types for Advance Payment	`DocumentTypeExport.csv`	Exports the ERP document types that are used for advance payments. This task is available only for sites integrated with SAP ERP systems.	File channel
Import Cancel Advance Payment Errors	`CancelAdvancePaymentErrorImport.csv`	Imports the errors generated by the ERP system while uploading advance payment documents marked for cancellation in the SAP Ariba Buying solutions.	File channel

Task Name	CSV Files Required	Task Description	Supported Integration Channels
Export GL Indicators for Advance Payments	`GLIndicatorExport.csv`	Exports the ERP GL indicators that are used for advance payments. This task is available only for sites integrated with SAP ERP systems.	File channel
Export Advance Payment Cancellation Reasons	`AdvancePaymentCancelReasonCodesExport_sap.csv` (For SAP ERP) `AdvancePaymentCancelReasonCodesExport_psoft.csv` (For PeopleSoft ERP) `AdvancePaymentCancelReasonCodesExport_generic.csv` (For Generic ERP)	Exports IDs and descriptions associated with the reasons for cancellation of advance payments to the ERP system.	File channel

About Importing Document Types for Advance Payments [page 199]

About Importing Translations for Advance Payment Document Types [page 199]

About Importing GL Indicators for Advance Payment [page 200]

About Importing Translations for Advance Payment GL Indicators [page 201]

About Exporting Advance Payments [page 201]

About Exporting Payment Requests with Advance Payments [page 209]

About Importing Advance Payment ID [page 230]

About Importing Advance Payment Errors [page 231]

About importing advance payment remittance [page 234]

About Exporting Cancel Advance Payments [page 237]

About Importing Cancel Advance Payment ID [page 239]

About Importing Cancel Advance Payment Errors [page 240]

About Importing Advance Payment Cancellation Reasons [page 242]

About Exporting Advance Payment Cancellation Reasons [page 243]

About Importing Document Types for Advance Payments

This task is required only for sites integrated with SAP ERP systems. It enables you to import the ERP document types that can be used for advance payments. The imported document types are used while creating advance payments.

This task is available using only the file channel. You can import document types for advance payments using the **Import Document Types for Advance Payment** task on the **Data Import/Export** page in Ariba Administrator.

The **Export ERP Document Types for Advance Payment** task enables you to export the ERP document types being used in the site for advance payments. It uses the same fields as that used for the **Import Document Types for Advance Payment** task.

Import Document Types for Advance Payment: Sample File

The table below provides descriptions and sample values for fields in the `DocumentType.csv` file required to import ERP document types for advance payments. This file is required only for sites integrated with SAP ERP systems.

Field Name	Required	Description	Sample Value
BLART	Yes	String. The unique identifier of the ERP document type	CP
LTEXT	Yes	MultiLingualString. The name of the ERP document type	Customer Payment

About Importing Translations for Advance Payment Document Types

This task is required only for sites integrated with SAP ERP systems. It is available using only the file channel. It enables you to import translations for the ERP document types used for advance payments. The translated document type names are displayed on the user interface while creating and viewing advance payments.

You can import translations for document types using the **Import Translations for Advance Payment Document Types** task on the **Data Import/Export** page in Ariba Administrator.

Import Translations for Advance Payment Document Types: Sample File

The table below provides descriptions and sample values for fields in the `DocumentType.csv` file required to import translations for ERP document types used for advance payments. This file is required only for sites integrated with SAP ERP systems.

Field Name	Required	Description	Sample Value
BLART	Yes	String. The unique identifier of the ERP document type	CP
LTEXT	No	MultiLingualString. The name of the ERP document type	Customer Payment
Language	Yes	String. The language to which the general ledger name is translated	French

About Importing GL Indicators for Advance Payment

This task is required only for sites integrated with SAP ERP systems. It enables you to import the ERP general ledger indicators that can be used for advance payments. The imported general ledger indicators are used while creating advance payments.

This task is available using only the file channel. You can import GL indicators for advance payments using the **Import GL Indicators for Advance Payment** task on the **Data Import/Export** page in Ariba Administrator.

The **Export GL Indicators for Advance Payments** task enables you to export the ERP GL indicators being used in the site for advance payments. It uses the same fields as that used for the **Import GL Indicators for Advance Payment** task.

Import GL Indicators for Advance Payment: Sample File

The table below provides descriptions and sample values for fields in the `GLIndicator.csv` file required to import and export ERP general ledger indicators for advance payments. This file is required only for sites integrated with SAP ERP systems.

Field Name	Required	Description	Sample Value
UMSKZ	Yes	String. The unique identifier of the ERP GL indicator	A

Field Name	Required	Description	Sample Value
LTEXT	Yes	MultiLingualString. The name of the ERP document type	Down Payment

About Importing Translations for Advance Payment GL Indicators

This task is required only for sites integrated with SAP ERP systems. It is available using only the file channel. This task enables you to import translations for the general ledgers used for advance payments. The translated general ledger names are displayed on the user interface while creating and viewing advance payments.

You can import translations for general ledger names using the **Import Translations for Advance Payment GL Indicators** task on the **Data Import/Export** page in Ariba Administrator.

Import Translations for Advance Payment GL Indicators: Sample File

The table below provides descriptions and sample values for fields in the `GLIndicator.csv` file required to import translations for ERP general ledger indicators for advance payments. This file is required only for sites integrated with SAP ERP systems.

Field Name	Required	Description	Sample Value
BLART	Yes	The unique identifier of the ERP GL indicator	A
LTEXT	No	The translated name of the ERP GL indicator	Paiement au vendeur
Language	Yes	The language to which the GL indicator name is translated	French

About Exporting Advance Payments

The **Export Advance Payments** task enables you to export the advance payment documents in your site for integration with external systems. You can use the file channel or Web Services channel to export advance payments.

Exporting Advance Payments Using the File Channel [page 202]

Exporting Advance Payments Using the Web Services Channel [page 205]

Exporting Advance Payments Using the File Channel

Before you export advance payments using the file channel, SAP Ariba Customer Support must complete the following configuration activities:

- Ensure that the parameter, `Application.Settlement.TransactionPush.AdvancePayment.Enabled` is enabled. Advance payments are not exported to a CSV file if this is not enabled.
- Configure a list of status strings using the parameter `Application.Settlement.TransactionPush.AdvancePayment.Statuses`. When the status of an advance payment document changes to any of the statuses specified in this parameter, the advance payment is automatically queued for export. By default, only approved advance payments are exported to the CSV file.

This task uses the following CSV files:

- `AdvancePaymentExport.csv` (for header-level advance payments)
- `AdvancePaymentLineItemExport.csv` (for line-level advance payments)

After you export advance payments, you need to import the advance payment IDs generated by the external system using the **Import Advance Payment ID** task. An advance payment document proceeds to the **Paying** status only when an advance payment ID is received for it from the external system.

If the export fails, you need to import the errors generated by the external system using the **Import Advance Payment Errors** task. In this case, the status of the advance payment document in SAP Ariba Buying solutions changes back to **Composing** and an email notification is sent to the requester, with details of the error. For more information on this email notification, see Payments Messages.

Export Advance Payments: Sample File - Generic and PeopleSoft ERP Systems [page 202]

Export Advance Payments: Sample File - SAP ERP System [page 204]

Export Advance Payments: Sample File - Generic and PeopleSoft ERP Systems

The following table provides descriptions and sample values for the fields in the `AdvancePaymentExport.csv` file generated by exporting header-level advance payments for integration with Generic and PeopleSoft ERP systems:

Field Name	Required	Description	Sample Value
`Advance Payment ID`	Yes	String. The system-generated unique identifier of the advance payment document	APMT10
`Title`	No	String. The user-specified title or name of the advance payment document	Advance payment for PO12

Field Name	Required	Description	Sample Value
`Amount`	Yes	BigDecimal. The amount to be paid as advance payment, in the user's currency	50
`Currency`	Yes	String. The unique name of the user's currency	EUR
`Amount_InBaseCurrency`	No	BigDecimal. The advance payment amount converted to the site base currency	64
`Order ID`	Yes	String. The unique identifier of the purchase order for which the advance payment is made	PO12
`Supplier`	Yes	String. The unique identifier of the supplier to which the advance payment is made	11
`Payment Due Date`	Yes	Date. The due date for making the advance payment.	12/31/2015 12:00:00 AM
`Payment Method`	Yes	String. The unique identifier of the payment method by which the advance payment needs to be made. The value specified here must match a value defined in the `PaymentMethodType.csv.` file.	CHK

The following table provides descriptions and sample values for fields in the `AdvancePaymentLineItemExport.csv` file generated by exporting line-level advance payments for integration with Generic and PeopleSoft ERP systems:

Field Name	Required	Description	Sample Value
`Advance Payment ID`	Yes	String. The system-generated unique identifier of the advance payment document.	APMT10
`Line Item`	No	String. The name or description of the line item.	Item name
`Advance Payment Amount`	Yes	BigDecimal. The amount to be paid as advance payment for a line item in the user's currency.	50

Export Advance Payments: Sample File - SAP ERP System

The following table provides descriptions and sample values for fields in the `AdvancePaymentExport.csv` file generated by exporting header-level advance payments for integration with SAP ERP system:

Field Name	Required	Description	Sample Value
`Advance Payment ID`	Yes	String. The system-generated unique identifier of the advance payment document.	APMT10
`Title`	No	String. The user-specified title or name of the advance payment document.	Advance payment for PO12
`Creation Date`	Yes	Date. The date when the advance payment document is created.	20160518
`Amount`	Yes	BigDecimal. The amount to be paid as advance payment, in the user's currency.	50
`Currency`	Yes	String. The unique name of the user's currency.	EUR
`Amount_InBaseCurrency`	No	BigDecimal. The advance payment amount converted to the site base currency.	64
`Order ID`	Yes	String. The unique identifier of the purchase order for which the advance payment is made.	PO12
`Supplier`	Yes	String. The unique identifier of the supplier to which the advance payment is made.	11
`Payment Due Date`	Yes	Date. The due date for making the advance payment.	12/31/2015 12:00:00 AM
`Payment Method`	Yes	String. The unique identifier of the payment method by which the advance payment needs to be made. The value specified here must match a value defined in the `PaymentMethodType.csv.` file.	CHK

Field Name	Required	Description	Sample Value
`CompanyCode`	Yes	String. The unique identifier of the company code of the purchase order associated with the advance payment.	001
`DocumentType`	Yes	String. The unique identifier of the ERP document type of the advance payment.	CP
`GeneralLedger`	Yes	String. The unique identifier of the ERP GL indicator of the advance payment.	A

The following table provides descriptions and sample values for fields in the `AdvancePaymentLineItemExport.csv` file generated by exporting line-level advance payments for integration with SAP ERP system:

Field Name	Required	Description	Sample Value
`Advance Payment ID`	Yes	String. The system-generated unique identifier of the advance payment document.	APMT10
`Line Item`	Yes	String. The name or description of the line item.	Item
`Advance Payment Amount`	Yes	BigDecimal. The amount to be paid as advance payment for the line item in the user's currency.	50
`PO_ITEM`	Yes	String. The line number of the item on the associated order.	12

Exporting Advance Payments Using the Web Services Channel

The **Export Advance Payments** task on the **Web Service Status** tab uses web services to export advance payment documents to external systems. On receiving the advance payment document, the external system sends the advance payment ID back to SAP Ariba Buying solutions. The status of the advance payment document in SAP Ariba Buying solutions changes to **Paying** when this ID is received. Payments are then made for these advance payments in the external system.

If the export fails, the external system sends an error message to SAP Ariba Buying solutions. In this case, the status of the advance payment document in SAP Ariba Buying solutions changes back to **Composing** and an email

notification is sent to the requester, with details of the error. For more information on this email notification, see Payments Messages.

AdvancePaymentExportRequest Elements [page 206]

AdvancePaymentExportReply Elements [page 207]

`AdvancePaymentExportRequest` Elements

`AdvancePayment_AdvancePaymentExportWS_Item`

The table below describes the fields in an `AdvancePayment_AdvancePaymentExportWS_Item` in the `AdvancePaymentExportRequest` sent to external systems.

Element	Required	Type and Description	Sample Value
`UniqueName`	Yes	String. The system-generated unique identifier of the advance payment document.	APMT10
`Name`	No	String. The user-specified title or name of the advance payment document	Advance payment for PO12
`Supplier`	Yes	String. The `Name` element within the `Supplier` element holds the name of the supplier to whom the advance payment is made	Supplier Corporation
`PurchaseOrder`	Yes	String. The `OrderID` element within the `PurchaseOrder` element holds the unique identifier of the purchase order for which the advance payment is made	PO12
`PaymentMethodType`	Yes	String. The `UniqueName` element within the `PaymentMethodType` element holds the unique identifier of the payment method by which the advance payment is made.	Cheque
`NetDueDate`	Yes	Date. The due date for making the advance payment.	12/31/2015 12:00:00 AM

Element	Required	Type and Description	Sample Value
`Amount`	Yes	BigDecimal. The `Amount` element within the `Amount` element holds the amount to be paid as advance payment, in user's currency	50
`Currency`	Yes	String. The `UniqueName` element within the `Currency` element, which is in turn within the `Amount` element, holds the unique name of the user's currency	EUR
`ApproxAmountInBaseCurrency`	No	BigDecimal. The `ApproxAmountInBaseCurrency` element within the `Amount` element holds the advance payment amount converted to the base currency	75
`CompanyCode`	Yes	String. The unique identifier of the company code of the purchase order associated with the advance payment. This field is applicable only for sites integrated with SAP ERP systems.	001
`DocumentType`	Yes	String. The unique identifier of the ERP document type of the advance payment. This field is applicable only for sites integrated with SAP ERP systems.	CP
`GeneralLedger`	Yes	String. The unique identifier of the ERP GL indicator of the advance payment. This field is applicable only for sites integrated with SAP ERP systems.	A

`AdvancePaymentExportReply` Elements

When advance payment documents are exported from SAP Ariba Buying solutions to external systems, the external system sends either an acknowledgement on receiving the advance payment document or the details of

the errors that occured while receiving the advance payment document. The acknowledgement or error details are sent in an `AdvancePaymentExportReply` in the `AdvancePayment_AdvancePaymentIDImport_Item` or `AdvancePaymentError_AdvancePaymentErrorImport_Item` item.

`AdvancePayment_AdvancePaymentIDImport_Item`

The table below describes the fields in an `AdvancePayment_AdvancePaymentIDImport_Item` in the `AdvancePaymentExportReply` sent by external systems.

Element	Required	Type and Description	Sample Value
`ERPRequestID`	Yes	String. The unique identifier generated for the advance payment document in the external system	Pay001245
`UniqueName`	Yes	String. The system-generated unique identifier of the advance payment document.	APMT102

`AdvancePaymentError_AdvancePaymentErrorImport_Item`

The table below describes the fields in an `AdvancePaymentError_AdvancePaymentErrorImport_Item` in the `AdvancePaymentExportReply` sent by external systems.

Element	Required	Type and Description	Sample Value
`Id`	Yes	String. The unique identifier of the advance payment document	APMT102
`Date`	No	Date. The date of creation of the error record	9/12/2006
`ERP`	No	String. The ERP type, such as PeopleSoft or SAP	Oracle
`ErrorTable`	No	String. The name of the table holding the error records	AP_INVOICE_LINES_INTERFACE
`ErrorValue`	No	String. The description of the error generated	INVALID DISTRIBUTION ACCT
`NumInSet`	No	Integer. The number of error records in a batch	1

Element	Required	Type and Description	Sample Value
`Type`	No	String. The type of the transaction for which the error was generated	Invoice

About Exporting Payment Requests with Advance Payments

The `Export Payment Requests with Advance Payments` task exports payment request details including details of advance payments and Funds Management accounting information (if enabled for SAP ERP) to the external ERP system. Use this task only if your site is configured to export approved invoices for payment directly from the SAP Ariba Buying solutions to the ERP system in CSV file format, and not from SAP Business Network.

Export Payment Requests with Advance Payments: Sample Files - SAP ERP [page 209]

Export Payment Requests with Advance Payments: Sample Files - PeopleSoft ERP [page 219]

Export Payment Requests with Advance Payments: Sample Files - Generic ERP [page 224]

Export Payment Requests with Advance Payments: Sample Files - SAP ERP

The `Export Payment Requests with Advance Payments` task for SAP ERP uses the following CSV files:

- `PaymentHeaderExport.csv`
- `PaymentLineItemDet.csv`
- `EnhancedPaymentAccountDet.csv`
- `PaymentDetailExport.csv`
- `PaymentTaxExport.csv`
- `AdvancePaymentDetailExport.csv`

The following table lists the fields in the `PaymentHeaderExport.csv` file:

Field name	Required	Type and description	Sample value
`DIFF_INV`	Yes	String. The unique identifier for the supplier.	125
`BLINE_DATE`	Yes	Date. Defines the baseline date of the payment.	20160101

Field name	Required	Type and description	Sample value
CURRENCY	Yes	String. The type of currency of the payment.	USD
BLINE_DATE_USERTIMEZONE	Yes	Date. Defines the baseline date of the payment in the user's timezone. The format of the date is YYYYMMDD.	20160815
PMNTTRMS	Yes	String. The unique code for the payment terms.	10
DOC_DATE	Yes	Date. The date on the invoice.	USD
PYMT_METH	Yes	String. The payment method used.	Card
BLINE_DATE_ISO_USERTIMEZONE	Yes	Date. Defines the baseline date of the payment in the requester's time zone.	08/28/2008
ACCRUAL_PMNT_AMT	Yes	BigDecimal. The accrued amount for payment.	250
GROSS_AMOUNT_ROUNDED	Yes	BigDecimal. The gross amount on the invoice reconciliation in ERP precision.	275
REF_DOC_NO	Yes	Defines the invoice number as reference document number.	IN23
URL	Yes	String. The URL for the payment.	url
BLINE_DATE_ISO	Yes	Date. Defines the baseline date of the payment.	08/28/2008
COMP_CODE	Yes	String. Specifies the company code applicable for the payment.	1000
GROSS_AMOUNT	Yes	BigDecimal. The gross amount of the payment.	230
INVOICE_IND	Yes	String. Indicates whether the document is an invoice or a credit memo.	INV
ITEM_TEXT	Yes	String. The description for the line item.	Line

Field name	Required	Type and description	Sample value
PAYMENTID	Yes	String. The unique ID of every payment for which a status is displayed.	PAYCK1-12

The following table lists the fields in the `PaymentLineItemDet.csv` file:

Field name	Required	Type and description	Sample value
ITEM_TEXT	Yes	String. The description for the line item.	Line
IS_COMPOSITE	Yes	Boolean. A flag to indicate whether a line is composite.	True
PO_UNIT	Yes	String. A unit of measure.	EA
QUANTITY	Yes	BigDecimal. The quantity for a line item.	3
PO_PR_UOM	No	String. The quantity for which the price is quoted.	5
ITEM_AMOUNT_ROUNDED	No	BigDecimal. The amount of the line item in ERP precision.	310
LINE_TYPE	Yes	String. The unique internal identifier of the line type for the line item. This field defines the tax type for indirect and withholding tax lines. For charges, this field specifies the values freight charge, handling charge, or charge. For discount lines, this field specifies the value discount. For other line items, this field specifies the values customer catalog item or non catalog item.	WHT
PO_ITEM	Yes	String. The line number of the item on the order.	12
PBQ_DESC	No	String. The description for the price basis quantity fields for a line item on the requisition.	200

Field name	Required	Type and description	Sample value
INVOICE_DOC_ITEM	Yes	Integer. The line number of an item on the invoice.	1
CONV_UNIT	No	BigDecimal. The quantity for which the price is quoted for.	25
PAYMENTID	Yes	String. The unique ID for every payment that has its status displayed.	PAYsap01-20
PO_NUMBER	Yes	String. The unique ID of the purchase order for the ERP system it belongs to.	23645
TAX_CODE	Yes	String. The unique code for the tax type.	12
POLINERECEIVINGTYPE	Yes	String. The purchase order receiving type.	3
ITEM_AMOUNT	Yes	BigDecimal. The amount for a line item.	50
CATEGORY	Yes	Integer. The category of a line item.	1
ORDER	Yes	String. The purchase order number in the ERP system.	EPCK24
PRICE_UNIT	No	BigDecimal. The price for the quoted quantity.	25

The following table lists the fields in the `EnhancedPaymentAccountDet.csv` file:

Field name	Required	Type and description	Sample value
GL_ACCOUNT	Yes	String. The unique value of the general ledger accounting type for each line item.	50506
LINE_COMP_CODE	No	String. The unique value of the company code for the line item. This field will be populated for non-purchase order invoices if your site is configured to enable accounting splits for line items on invoices.	3000

Field name	Required	Type and description	Sample value
SUB_NUMBER	Yes	String. The asset subnumber for the asset account assignment category for a line item.	0005
PO_UNIT	Yes	String. The unique unit of measure value.	EA
PO_PR_UOM	No	String. The unit of measure for the quantity defined by price basis quantity.	EA
SERIAL_NO	Yes	String. The serial number of a line item.	2
COSTCENTER	Yes	String. The unique value of the cost center account assignment category for a line item.	2100
ASSET_NO	Yes	String. the unique value of the asset account assignment category for each line item.	1005
ITEM_AMOUNT	Yes	BigDecimal. The amount for a line item.	24
LINE_TYPE	Yes	String. The unique internal identifier of the line type for an item. This field holds the tax type for indirect and withholding tax lines. For charges, this field specifies the values freight charges, handling charges, or other charges. For discount lines, this field specifies the value for discounts. For other line items, this field specifies the values for customer catalog or non-catalog items.	WHTax
ITEM_AMOUNT_ROUNDED	No	BigDecimal. The amount for a line item in ERP precision.	33
WBS_ELEM	Yes	String. The unique code of the WBS element account assignment category for that line item.	3-1200/31

Field name	Required	Type and description	Sample value
INVOICE_DOC_ITEM	Yes	Integer. The line number of an item.	2
ORDERID	Yes	String. The unique value of the internal order account assignment category for a line item.	40046
PAYMENTID	Yes	String. The unique ID of a payment that has a status displayed.	PAYsap01-20
PO_NUMBER	Yes	String. The unique ID of a purchase order for the ERP system it belongs to.	4536000
POLINERECEIVINGTYPE	Yes	String. The purchase order receiving type.	2
QUANTITY	Yes	BigDecimal. The quantity for a line item.	21
TAX_CODE	Yes	String. The unique tax type code.	29
CATEGORY	Yes	Integer. The category of a line item.	20
ORDER	Yes	String. The purchase order number in the ERP system.	EPCK24
ACTN	No	String. The unique value of the network account assignment category for each line item.	Network1
ACTY	No	String. The unique value of the activity number account assignment category for each line item.	4001

The following table lists the fields in the `PaymentDetailExport.csv` file:

Field name	Required	Type and description	Sample value
ITEM_TEXT	Yes	String. The description for the line item.	Line
GL_ACCOUNT	Yes	String. The unique value of the general ledger accounting type for a line item.	56500

Field name	Required	Type and description	Sample value
LINE_COMP_CODE	Yes	String. The unique value of the Company Code for the line item.	3000
SUB_NUMBER	No	String. The subnumber of the asset account assignment category for a line item.	0005
PO_PR_UOM	No	String. The quantity for which the price is quoted.	5
CHARGE_FLAG	Yes	Integer. The line number of an item.	2
ITEM_AMOUNT	Yes	BigDecimal. The amount for the line item in ERP precision.	50
LINE_TYPE		String. The unique internal identifier of the line type for the line item. This field defines the tax type for indirect and withholding tax lines. For charges, this field specifies the values FreightCharge, HandlingCharge, or Charge. For discount lines, this field specifies the value Discount. For other line items, this field specifies the values CatalogItem or NonCatalogItem.	WHT
WBS_ELEM	Yes	String. The unique code of the WBS element account assignment category for a line item.	3-1200/31
ORDERID	Yes	String. The unique value of the internal order account assignment category for a line item.	4560
DB_CR_IND	Yes	String. A debit or credit indicator.	D
PO_NUMBER	Yes	String. The unique ID of the purchase order for the ERP system it belongs to.	23654
IS_COMPOSITE	Yes	Boolean. A flag to indicate whether a line is composite.	True

Field name	Required	Type and description	Sample value
HDR_LEVEL	No	Boolean. Specifies whether a line item is at the header level.	No
COSTCENTER	Yes	String. The unique value of the cost center account assignment category for a line item.	2500
ASSET_NO	Yes	String. The unique value of the asset account assignment category for a line item.	1005
PBQ_DESC	No	String. The description for the price basis quantity fields for a line item on the requisition.	200
ITEM_AMOUNT_ROUNDED	No	BigDecimal. The amount for the line item in ERP precision.	250
COMP_CODE	Yes	String. The company code of the payment.	3000
PLANT	Yes	String. The unique code of the ship-to location for a line item.	3200
INVOICE_DOC_ITEM	Yes	Integer. The line number of an item on the invoice.	1
CONV_UNIT	No	BigDecimal. The quantity for which the price is quoted for.	25
PAYMENTID	Yes	String. The unique ID for every payment that has its status displayed.	PAYCK30
PO_NUMBER	Yes	String. The unique ID of the purchase order for the ERP system it belongs to.	23645
TAX_CODE	Yes	String. The unique code for the tax type.	12
LINE_TYPE_LOOKUP_CODE	Yes	String. The category of line item.	3
PRICE_UNIT	No	BigDecimal. The price for the quoted quantity.	25

Field name	Required	Type and description	Sample value
ACTN	No	String. The unique value of the network account assignment category for each line item.	Network1
ACTY	No	String. The unique value of the activity number account assignment category for each line item.	4001

The following table lists the fields in the `PaymentTaxExport.csv` file:

Field name	Required	Type and description	Sample value
TAX_AMOUNT	Yes	BigDecimal. Defines the amount of tax.	50
INVOICE_DOC_ITEM	Yes	Integer. Defines the line number of the invoice line item associated with the tax distribution.	12
TAX_AMOUNT_ROUNDED	No	BigDecimal. Defines the amount of tax in the precision defined the ERP system.	50
TAX_CODE	Yes	String. Defines the tax code.	10
DB_CR_IND	Yes	String. Indicates a debit or a credit.	D
PAYMENTID	Yes	String. The unique ID of a payment that has a status displayed.	PAYsap01-20
ITEM_TEXT	Yes	String. The description for a line.	21
REF_LINE_NUM	Yes	Integer. The reference line number in the tax distribution vector.	6

The following table lists the fields in the `AdvancePaymentDetailExport.csv` file:

Field name	Required	Type and description	Sample value
PAYMENTID	Yes	String. The unique ID of a payment that has a status displayed.	PAYsap01-20

Field name	Required	Type and description	Sample value
CONSUMEDAMOUNT_AMOUNT	Yes	BigDecimal. The advance payment amount adjusted against the invoice in the user's currency.	12
FISCALYEAR	Yes	Integer. The financial year for the advance payment transaction.	2016
CONSUMEDAMOUNT_CURRENCY	Yes	String. The currency applicable for the user.	USD
INVOICERECONCILIATIONID	Yes	String. The unique identifier of the invoice reconciliation document for the invoice.	35
ADVANCEPAYMENTID	Yes	String. The unique identifier of the advance payment that is adjusted against the invoice.	APMT10
ERPADVANCEPAYMENTID	Yes	String. The unique identifier of the payment request created in the ERP for the advance payment.	210
CONSUMEDAMOUNT_INBASECURRENCY	Yes	BigDecimal. The advance payment amount adjusted against the invoice in the site's base currency.	250
ORDERID	Yes	String. The unique identifier of the purchase order associated with the invoice and the advance payment documents.	21
ORDERITEM	Yes	String. The line number assigned to a line item on the advance payment.	6
ERPADVANCEPAYMENTITEM	Yes	Integer. The line item for advance payment remittance.	22

Export Payment Requests with Advance Payments: Sample Files - PeopleSoft ERP

The `Export Payment Requests with Advance Payments` task for PeopleSoft ERP uses the following CSV files:

- `PaymentHeaderExport.csv`
- `PaymentDetailExport.csv`
- `PaymentSplitAccountingExport.csv`
- `PaymentBankExport.csv`
- `AdvancePaymentDetailExport.csv`

The following table lists the fields in the `PaymentHeaderExport.csv` file:

Field name	Required	Type and description	Sample value
VENDOR_LOC	Yes	String. The identifier for the location of the vendor.	10
TXN_CURRENCY_CD	Yes	String. The currency code for the tax.	21
PAYMENT_TERMS	Yes	String. The unique code for the payment terms.	25
INVOICE_DT	Yes	String. The date on the invoice or the voucher.	20160815
GROSS_AMT	Yes	BigDecimal. The total amount on the invoice.	1000
INVOICE_ID	Yes	String. The unique ID on the invoice or the voucher.	PYT234
REMIT_ADDR_SEQ_NUM	Yes	String. The sequence number for the remittance address.	23
VENDOR_ADDR_SEQ_NUM	Yes	String. The sequence number of the supplier's location.	345
VENDOR_SETID	Yes	String. The set ID of the supplier.	100
ECQUEUEINSTANCE	Yes	String. Unique identifier for the invoice.	INV10
BUSINESS_UNIT	Yes	String. The businness unit of the user.	DL25

Field name	Required	Type and description	Sample value
PMNT_ID	Yes	String. The unique payment ID generated by the buyer for the invoice or the voucher.	PMNT10
VENDOR_ID	Yes	String. The unique ID of the supplier.	235
TAX_ACCRUAL_PMNT	Yes	Boolean. Specifies whether tax is accrued.	Y

The following table lists the fields in the `PaymentDetailExport.csv` file:

Field name	Required	Type and description	Sample value
DESCR	Yes	String. The description for a line on the invoice.	Text
CURRENCY_CD	Yes	String. The currency code for a line item on an invoice.	USD
SHIPTO	Yes	String. The ship to location for the invoice line item.	NGLS
UNIT_PRICE	Yes	BigDecimal. The unit price for a line item.	3
IsComposite	No	Boolean. The flag indicating whether a line is a composite.	True
BUSINESS_UNIT_PO	Yes	String. The business unit of the associated order for a purchase order-based invoice	265
UNIT_OF_MEASURE	Yes	String. The unit of measure for a line item.	EA
PriceBasisQuantityUOM	Yes	String. The quantity for which the price is quoted for.	12
TAX_ACCRUAL	Yes	Boolean. Specifies whether tax is accrued.	Yes
MERCHANDISE_AMT	Yes	BigDecimal. The total amount on the order.	28
PARENT_LINE_NUM	Yes	Integer. The parent line number for a line on the invoice.	25
TAX_TYPE	Yes	String. The type of tax.	PAYsap01-20

Field name	Required	Type and description	Sample value
LINE_NBR	Yes	String. The purchase order line number associated with the invoice reconciliation line number.	21
PriceBasisQuantityDesc	No	String. The description that describes the price basis quantity fields for an item on the invoice.	A description.
VOUCHER_LINE_NUM	Yes	Integer. The voucher line number.	3
ECQUEUEINSTANCE	Yes	String. The queue instance which is unique for a buyer instance.	50
BUSINESS_UNIT	Yes	String. The business unit of the user.	DL24
ConversionFactor	No	BigDecimal. The quantity for which the price is quoted for.	12
PMNT_ID	No	String. The payment ID generated by the buyer.	PMNT10
TAX_CODE	Yes	String. The unique tax code.	10
PO_ID	Yes	String. The ID of the order associated with a purchase order-based invoice.	PO23
LINE_TYPE_LOOKUP_CODE	Yes	Integer. The unique identifier for the line type.	12
RECEIVING_TYPE	Yes	Integer. The purchase order receiving type. This value specifies whether the receipts are amount based.	2
PriceBasisQuantity	Yes	BigDecimal. The quantity for which the price is quoted for.	25

The following table lists the fields in the `PaymentSplitAccountingExport.csv` file:

Field name	Required	Type and description	Sample value
SPLIT_TYPE	Yes	String. Defines the split type.	2

Field name	Required	Type and description	Sample value
DEPTID	Yes	String. The department to which the user belongs.	5000
BUSINESS_UNIT_GL	Yes	String. The business unit of the user.	DL2003
ACCOUNT	Yes	String. The account number.	24521
MERCHANDISE_AMT	Yes	BigDecimal. The total amount of the purchase order.	2350
VOUCHER_LINE_NUM	Yes	Integer. The voucher line number for a line item.	2
DISTRIB_LINE_NUM	Yes	Integer. The distribution line number for a line item	21
ECQUEUEINSTANCE	Yes	String. The queue instance which is unique for a buyer instance.	50
BUSINESS_UNIT	Yes	String. The businness unit of the user.	24
PMNT_ID	No	String. The payment ID generated by the buyer.	PMNT10
PO_ID	Yes	String. The ID of the order associated with a purchase order-based invoice.	PO23
QTY_VCHR	Yes	BigDecimal. The quantity of a line item.	3
RECEIVING_TYPE	Yes	Integer. The purchase order receiving type. This value specifies whether the receipts are amount based.	1

The following table lists the fields in the `PaymentBankExport.csv` file:

Field name	Required	Type and description	Sample value
PAYMENT_AMT	Yes	BigDecimal. The payment amount.	250

Field name	Required	Type and description	Sample value
PAYMENT_GROSS_AMT	Yes	Integer. Defines the line number of the invoice line item associated with the tax distribution.	12
BANK_ACCT_KEY	No	BigDecimal. Defines the amount of tax in the precision defined the ERP system.	50
PAYMENT_CURRENCY_CD	Yes	String. Defines the tax code.	10
SCHEDULED_PAY_DT	Yes	Date. The scheduled payment date.	20160101
PYMNT_METHOD	Yes	String. The payment method.	25
BANK_CD	Yes	String. The code which uniquely indentifies a bank.	21
ECQUEUEINSTANCE	Yes	String. The queue instance which is unique for a buyer instance.	50
BUSINESS_UNIT	Yes	String. The business unit of the user.	200
COMPOSITE_ACCT_KEY	Yes	String. The unique identifier of the composite account.	32
PMNT_ID	Yes	String. The payment ID generated by the buyer.	PMNT10
PAYMENT_DISC_AMT	Yes	BigDecimal. The discount amount for the payment.	2

The following table lists the fields in the `AdvancePaymentDetailExport.csv` file:

Field name	Required	Type and description	Sample value
PMNT_ID	Yes	String. The unique identifier of the payment request created for the invoice.	PMNT10
CONSUMEDAMOUNT_AMOUNT	Yes	BigDecimal. The advance payment amount adjusted against the invoice, in the user's currency.	12

Field name	Required	Type and description	Sample value
`FiscalYear`	Yes	Integer. The financial year for the advance payment transaction.	2016
`CONSUMEDAMOUNT_CURRENCY`	Yes	String. The currency applicable for the user.	50
`ECQUEUEINSTANCE`	Yes	String. The queue instance which is unique for a buyer instance.	50
`ADVANCEPAYMENT_ID`	Yes	String. The unique identifier of the advance payment that is adjusted against the invoice.	APMT10
`ERPAdvancePaymentItem`	Yes	Integer. The line item for the advance payment remittance.	21
`ERPADVANCEPAYMENT_ID`	Yes	String. The unique identifier of the payment request created in the ERP for the advance payment.	210
`CONSUMEDAMOUNT_INBASECURRENCY`	Yes	BigDecimal. The advance payment amount adjusted against the invoice in the site's base currency.	250
`ORDER_ID`	Yes	String. The unique identifier of the purchase order associated with the invoice and the advance payment documents.	21

Export Payment Requests with Advance Payments: Sample Files - Generic ERP

The `Export Payment Requests with Advance Payments` task for Generic ERP uses the following CSV files:

- `PaymentHeaderExport.csv`
- `PaymentDetailExport.csv`
- `PaymentSplitAccountingExport.csv`
- `PaymentBankExport.csv`
- `AdvancePaymentDetailExport.csv`

The following table lists the fields in the `PaymentHeaderExport.csv` file:

Field name	Required	Type and description	Sample value
`PurchasingUnit`	Yes	String. The unique ID for the purchasing unit.	200
`GrossAmountCurrency`	Yes	String. The currency code for the gross amount on the invoice.	USD
`PaymentTerms`	Yes	String. The payment terms on the invoice.	10001
`SupplierLocation`	Yes	String. The unique internal identifier of the supplier location for which the payment is made.	2550
`InvoiceDate`	Yes	String. The date on which the invoice is created.	02/19/2008 16:13:29 +0530
`GrossAmount`	Yes	BigDecimal. The gross payment amount.	500
`InvoiceNumber`	Yes	String. The invoice number associated with invoice.	INV25
`RemittanceLocation`	Yes	String. The remittance location for the invoice.	LOC15
`Supplier`	Yes	String. The unique identifier for the supplier to whom the invoice is sent.	4814
`InvoiceReconciliationId`	Yes	String. The unique identifier for the invoice reconciliation.	IR56
`PaymentId`	Yes	String. The unique identifier for the payment.	PAYINVGF12-10
`ForTaxAccrual`	Yes	Boolean. Indicates whether tax accrual is applicable for the invoice.	No
`SupplierLocationContactID`	Yes	String. The ID assigned to the contact person at the supplier location.	450

The following table lists the fields in the `PaymentDetailExport.csv` file:

Field name	Required	Type and description	Sample value
`PurchasingUnit`	Yes	String. The unique ID for the purchasing unit.	500
`Description`	Yes	String. The description of the line item.	Description
`ERPOrderID`	Yes	String. The purchase order ID associated with the invoice reconciliation line item.	32
`PriceBasisQuantityUOM`	Yes	Boolean. The quantity for which the price is quoted for.	EA
`IsAccrual`	Yes	Boolean. Specifies whether the line item has an accrual amount.	No
`ERPOrderLineNumber`	Yes	Integer. The purchase order line number associated with the invoice reconciliation line number.	21
`Amount`	Yes	BigDecimal. The amount for an item.	12
`ParentLineNumber`	Yes	Integer. The unique line number of the parent line item.	Yes
`TaxAdjustmentAmount`	Yes	BigDecimal. The amount adjusted as tax.	25
`AmountCurrency`	Yes	String. The currency type for the payment amount.	USD
`UnitPrice`	No	BigDecimal. The net price of a line item on the order.	22
`IsComposite`	Yes	Boolean. The Voucher Line number.	3
`Quantity`	Yes	BigDecimal. The net quantity of a line item on the order.	10
`TaxType`	No	String. The unique internal identifier of a tax type.	50

Field name	Required	Type and description	Sample value
`PriceBasisQuantityDesc`	Yes	String. The description that describes the price basis quantity fields for an item on the invoice.	A description.
`LineNumber`	Yes	Integer. The unique identifier indicating the line number.	2
`InvoiceReconciliationId`	Yes	String. The unique identifier of the invoice reconciliation.	IR12
`ERPCommodityCode`	Yes	String. The unique internal identifier of the ERP commodity code.	BENEFIT:AUTOLEASE
`ConversionFactor`	Yes	BigDecimal. The quantity for which the price is quoted for.	25
`PaymentId`	Yes	String. The unique identifier for the payment.	PAYINVGF12-10
`LineTypeCategory`	Yes	Integer. The unique identifier for tax, shipping, and miscellaneous for a line.	1
`TaxCode`	Yes	String. The tax code applied for a line item.	2
`PriceBasisQuantity`	Yes	BigDecimal. The quantity for which the price is quoted for.	15

The following table lists the fields in the `PaymentSplitAccountingExport.csv` file:

Field name	Required	Type and description	Sample value
`PurchasingUnit`	Yes	String. The unique ID for the purchasing unit.	500
`BusinessUnit`	Yes	String. The unique ID of the business unit for each split line item.	1001
`Product`	Yes	String. The unique ID of the product for each split line item.	752

Field name	Required	Type and description	Sample value
`SubAccount`	Yes	String. The unique ID of the sub account for each split line item.	650
`CostCenter`	Yes	String. The unique ID of the cost center for a split line item.	2150
`Company`	Yes	String. The unique ID of the company for each split line item.	2000
`Project`	Yes	String. The unique ID of the project for each split line item.	2100
`Region`	Yes	String. The unique ID of the region for each split line item.	750
`Amount`	Yes	BigDecimal. The amount for a split line item.	25
`LineNumber`	Yes	Integer. The unique identifier that specifies the line number.	2
`SplitLineNumber`	Yes	Integer. The unique identifier that specifies the line number of the split account.	3
`InvoiceReconciliationId`	Yes	String. The unique identifier of the invoice reconciliation.	12
`PaymentId`	Yes	String. The unique identifier of the payment.	PMNT12
`Account`	Yes	Account. The unique internal identifier of an account.	125
`Quantity`	Yes	BigDecimal. The number of split accounts.	5
`AccountingCombinationID`	Yes	String. The ID for the split accounting combination.	24

The following table lists the fields in the `PaymentBankExport.csv` file:

Field name	Required	Type and description	Sample value
`PurchasingUnit`	Yes	String. The unique ID for the purchasing unit.	500
`Amount`	Yes	BigDecimal. The amount for a line item.	300
`GrossAmount`	Yes	BigDecimal. The gross payment amount.	1500
`BankAccountType`	Yes	String. Defines the tax code.	10
`AmountCurrency`	Yes	Date. The currency used for the payment.	USD
`ScheduledPaymentDate`	Yes	Date. The scheduled payment date.	20160801
`PaymentMethod`	Yes	String. The code which uniquely indentifies a bank.	21
`BankId`	Yes	String. The identifier of the paying bank.	BNK50
`InvoiceReconciliationId`	Yes	String. The unique identifier for invoice reconciliation.	IRINVGF12-10
`BuyerBankLocation`	Yes	String. The unique identifier of the bank location of the paying organization.	PYORG12
`PaymentId`	Yes	String. The unique identifier of the payment.	PMNT12
`DiscountAmount`	Yes	BigDecimal. The value for the amount specified as discount.	10

The following table lists the fields in the `AdvancePaymentDetailExport.csv` file:

Field name	Required	Type and description	Sample value
`PaymentId`	Yes	String. The unique identifier of the payment request created for the invoice.	PMNT12
`ConsumedAmount_Amount`	Yes	BigDecimal. The advance payment amount adjusted against the invoice, in the user's currency.	12

Field name	Required	Type and description	Sample value
`FiscalYear`	Yes	Integer. The financial year for the advance payment transaction.	2016
`ConsumedAmount_Currency`	Yes	String. The currency applicable for the user.	50
`InvoiceReconciliationId`	Yes	String. The unique identifier of the invoice reconciliation document for the invoice.	IR50
`ERPAdvancePaymentId`	Yes	String. The unique identifier of the payment request created for the advance payment in the ERP system.	AP10
`AdvancePaymentId`	Yes	Integer. The line item for the advance payment remittance.	APMT21
`ERPAdvancePaymentItem`	Yes	String. The unique identifier of the payment request created in the ERP for the advance payment.	210
`ConsumedAmount_InBaseCurrency`	Yes	BigDecimal. The advance payment amount adjusted against the invoice in the site's base currency.	250
`OrderId`	Yes	String. The unique identifier of the purchase order associated with the invoice and the advance payment documents.	21

About Importing Advance Payment ID

This task enables you to import the unique identifiers generated by the external system for the advance payment documents received from SAP Ariba Buying solutions. The status of the advance payment document in SAP Ariba Buying solutions changes from **Approved** to **Paying** when an ERP ID is received for it.

To import advance payment IDs using the file channel, run the **Import Advance Payment ID** task on the **Data Import/Export** page in Ariba Administrator.

In case of the Web Services channel, advance payments IDs are imported in response to the exported advance payments and you do not need to manually invoke any task.

Import Advance Payment ID: Sample File

The table below provides descriptions and sample values for fields in the `AdvancePaymentIDImport.csv` file required to import advance payment IDs generated by the external systems. This file can be used for all types of ERP systems.

Field Name	Required	Type and Description	Sample Value
`AdvancePayment_ID`	Yes	String. The system-generated unique identifier of the advance payment document	APMT10
`ERPPostingDate`	Yes	Date. The date of posting a document in the ERP system.	20160518
`ERP_Payment_ID`	Yes	String. The unique identifier generated for the advance payment document in the external system	00005429
`FISCALYEAR`	Yes	String. The financial year of the advance payment transaction.	2016

About Importing Advance Payment Errors

This task enables you to import the errors generated by the external system while uploading the advance payment documents sent by SAP Ariba Buying solutions. The imported error data is included in the email notification sent to the requesters of the advance payment documents.

To import advance payment errors using the file channel, run the **Import Advance Payment Errors** task on the **Data Import/Export** page in Ariba Administrator.

In case of the Web Services channel, advance payments errors are imported in response to the exported advance payments and you do not need to manually invoke any task.

Import Advance Payment Errors: Sample File - Generic ERP Systems [page 232]

Import Advance Payment Errors: Sample File - SAP ERP Systems [page 232]

Import Advance Payment Errors: Sample File - PeopleSoft ERP Systems [page 233]

Import Advance Payment Errors: **Sample File - Generic ERP Systems**

The table below provides descriptions and sample values for fields in the `AdvancePaymentErrorImport.csv` file required to import the errors generated by the external system while uploading advance payment documents sent by SAP Ariba Buying solutions.

Field Name	Required	Description	Sample Value
`AdvancePaymentID`	Yes	String. The unique identifier of the advance payment document	APMT10
`Type`	Yes	String. The type of the transaction for which the error was generated	Invoice
`Erp`	Yes	String. The ERP type, such as PeopleSoft or SAP	Oracle
`CreationDate`	Yes	Date. The date of creation of the error record	9/12/2006
`ParentTable`	Yes	String. The name of the table holding the error records	AP_INVOICE_LINES_INTERFACE
`RejectLookupCode`	Yes	String. The description of the error generated	INVALID DISTRIBUTION ACCT
`NumInSet`	No	Integer. The number of error records in a batch	0

Import Advance Payment Errors: **Sample File - SAP ERP Systems**

The table below provides descriptions and sample values for fields in the `AdvancePaymentErrorImport.csv` file required to import the errors generated by the external system while uploading advance payment documents sent by SAP Ariba Buying solutions. These fields are used for SAP variants.

Field Name	Required	Description	Sample Value
`ADVANCEPAYMENTID`	Yes	String. The unique identifier of the advance payment document	APMT10

Field Name	Required	Description	Sample Value
FLDNAME	Yes	String. The name of the field for which the error is generated	PO_ITEMo
MSGID	Yes	String. The unique identifier of the error message	M8
TYPE	Yes	String. The type of the transaction for which the error was generated	E
MSGNR	Yes	String. The sequence number of the error	375
MESSAGE	Yes	MultiLingualString. The description of the error	Fill in mandatory field PO_ITEM (table parameter ITEMDATA, row 000003)
DATETIME2	Yes	Date. The date of creation of the error record	3/6/2015 6:30:00 AM
NUMINSET	No	Integer. The number of error records in a batch	0

Import Advance Payment Errors: **Sample File - PeopleSoft ERP Systems**

The table below provides descriptions and sample values for fields in the `AdvancePaymentErrorImport.csv` file required to import the errors generated by external systems while uploading advance payment documents sent by SAP Ariba Buying solutions. These fields are used for PeopleSoft ERP systems.

Field Name	Required	Description	Sample Value
AdvancePaymentID	Yes	String. The unique identifier of the advance payment document	APMT10
Type	Yes	String. The type of the transaction for which the error was generated	Invoice
Erp	Yes	String. The ERP type, such as PeopleSoft or SAP	Oracle

Field Name	Required	Description	Sample Value
`CreationDate`	Yes	Date. The date of creation of the error record	2015-8-24
`RecordName`	Yes	String. The name of the table holding the error records	PS_VOUCHER
`ErrorMessage`	Yes	MultiLingualString. The description of the error generated	AP_VCHRBLD results in error
`ErrorCode`	Yes	String. The error code of the error generated	9999
`NumInSet`	No	Integer. The number of error records in a batch	0

About importing advance payment remittance

This task imports remittance information for advance payments from external systems. Multiple remittance details can be received for one advance payment if the external system makes the payment in installments. The status of an advance payment changes to **Paid** when remittance information is received for its entire amount at the header level.

To import advance payment remittance using the file channel, run the **Import Advance Payment Remittance** task on the **Data Import/Export** page in Ariba Administrator.

> **Note**
>
> For a successful import of advance payment remittance data, first run the **Import Advance Payment ID** task. That task changes the status of advance payments to **Paying**, which is required before the remittance import can change the status to **Paid**.

Line-level and header-level advance payment remittances

For each partial remittance, a line-level advance payment is to be created. When you import line-level advance payment remittances, the import includes both a header-level file and a detail (line-level) file. For line-level advance payments:

- The remittance amount must be specified in the line-level file, `AdvancePaymentRemittanceDetail.csv`.
- The `Amount` field in the header file, `AdvancePaymentRemittance.csv`, must be 0.
- Except for `Amount`, fields that exist in both `AdvancePaymentRemittance.csv` and `AdvancePaymentRemittanceDetail.csv` are expected to contain the same value in both files.

If the header-level file includes an amount other than 0 for a line-level advance payment, the status of the advance payment doesn't change to **Paid**.

For header-level advance payments, specify the entire amount in `AdvancePaymentRemittance.csv`. The `AdvancePaymentRemittanceDetail.csv` file can be empty.

Web services

In case of the web services channel, advance payments remittance information is imported as and when the payments are made by the external system. You do not need to manually invoke any task to import this information.

Related Information

About Importing Advance Payment ID [page 230]
Import Advance Payment Remittance: sample file [page 235]

Import Advance Payment Remittance: sample file

Sample values for header-level advance payment remittance

The following table provides descriptions and sample values for fields in the `AdvancePaymentRemittance.csv` file required to import remittance information for header-level advance payments from the external systems. This file can be used for all types of ERP systems.

For header-level advance payments, the file `AdvancePaymentRemittanceDetail.csv` can be blank.

Field name	Required	Description	Sample value
`ERPRequestID`	Yes	String. The unique identifier generated for the advance payment document in the external system.	00005429
`DocID`	Yes	String. The unique identifier for the advance payment remittance document in the external ERP system.	APMT10

Field name	Required	Description	Sample value
`LookupID`	No	String. The unique identifier generated for the remittance document in the external system.	00346
`Amount`	Yes	BigDecimal. The remittance amount	40
`Currency`	No	String. The currency in which the remittance was made	USD

Sample values for line-level advance payment remittance

The following table provides descriptions and sample values for fields in the `AdvancePaymentRemittance.csv` and `AdvancePaymentRemittanceDetail.csv` files required to import remittance information for line-level advance payments from the external systems. This file can be used for all types of ERP systems.

Field name	Required	Description	Sample value
`AdvancePaymentRemittance.csv`			
`LookupID`	No	String. The unique identifier generated for the remittance document in the external system.	00346
`DocID`	Yes	String. The unique identifier for the advance payment remittance document in the external ERP system.	APMT10
`ERPRequestID`	Yes	String. The unique identifier generated for the advance payment document in the external system.	00005429
`Amount`	Yes	BigDecimal. The remittance amount.	0 For line-level advance payments, the `Amount` value in the header-level file must be 0.
`Currency`	No	String. The currency in which the remittance was made.	USD
`AdvancePaymentRemittanceDetail.csv`			

Field name	Required	Description	Sample value
`DocID`	Yes	String. The unique identifier for the advance payment remittance document in the external ERP system.	APMT10
`DocumentLineNumber`	Yes	Integer. The line number of the item in the advance payment remittance document in the ERP system.	23
`ERPRequestID`	Yes	String. The unique identifier generated for the advance payment document in the external system.	00005429
`ERPRequestLineNumber`	Yes	String. The unique identifier for the line item in the advance payment document.	512
`Amount`	Yes	BigDecimal. The remittance amount.	40
`Currency`	No	String. The currency in which the remittance was made.	USD

About Exporting Cancel Advance Payments

The **Export Cancel Advance Payments** task enables you to export advance payments with **Canceling** status for integration with external systems. Based on the exported canceled advance payments, the advance payments are canceled in the external system. You can use the file channel or Web Services channel to export canceled advance payments.

Exporting Cancel Advance Payments Using the File Channel [page 237]

Exporting Cancel Advance Payments Using the Web Services Channel [page 238]

Exporting Cancel Advance Payments Using the File Channel

You can export cancel advance payments to the external system using the **Export Cancel Advance Payments** task on the **Data Import/Export** page in Ariba Administrator.

After you export cancel advance payments, you need to import the IDs of the advance payments canceled in the external system, using the **Import Cancel Advance Payment ID** task. The status of an advance payment document

changes from **Canceling** to **Canceled** only when a canceled advance payment ID is received for it from the external system.

Export Cancel Advance Payments: Sample File

The table below provides descriptions and sample values for fields in the `CancelAdvancePaymentExport.csv` file generated by exporting canceled advance payments for integration with external systems. This file is used for all types of ERP systems.

Field Name	Required	Description	Sample Value
`Advance Payment ID`	Yes	String. The system-generated unique identifier of the advance payment document.	APMT10
`ERP_Payment_ID`	No	String. The unique identifier of the payment request created in the ERP for the advance payment	AP10093

Exporting Cancel Advance Payments Using the Web Services Channel

The **Export Cancel Advance Payments** task on the **Web Service Status** tab uses web services to export the canceled advance payment documents to external systems. On receiving these details, the external system cancels the advance payments and sends the IDs of the canceled advance payments to SAP Ariba Buying solutions. The status of the advance payment document in SAP Ariba Buying solutions changes from **Canceling** to **Canceled** when these IDs are received. If an error occurs in the external system while canceling an advance payment, the advance payment document in SAP Ariba Buying solutions stays in the **Canceling** state until it is manually marked as canceled.

`CancelAdvancePaymentExportRequest` Elements

`CancelAdvancePaymentExportWS_Item`

The table below describes the fields in the `AdvancePayment_CancelAdvancePaymentExportWS_Item` in the `CanceledAdvancePaymentExportRequest` sent to external systems.

Element	Required	Type and Description	Sample Value
`UniqueName`	Yes	String. The system-generated unique identifier of the canceled advance payment document.	APMT10
`ERPRequestID`	No	String. The unique identifier generated for the advance payment document in the external system	Pay001245

`CancelAdvancePaymentExportReply` Elements

`AdvancePayment_CancelAdvancePaymentIDImport_Item`

When canceled advance payment documents are exported from SAP Ariba Buying solutions to external systems, the external system sends back the IDs of the advance payment documents canceled in the external system. The IDs are sent in `AdvancePayment_CancelAdvancePaymentIDImport_Item` in the `CancelAdvancePaymentExportReply` item.

Element	Required	Type and Description	Sample Value
`ERPRequestID`	Yes	String. The unique identifier of the canceled advance payment document in the external system	Pay001245
`UniqueName`	Yes	String. The system-generated unique identifier of the advance payment document.	APMT10

About Importing Cancel Advance Payment ID

This task enables you to import the IDs of the advance payment documents that have been successfully canceled in the external system. This task follows the **Export Cancel Advance Payments** task, which exports the canceled advance payments to external systems. The status of the advance payment document in SAP Ariba Buying solutions changes from **Canceling** to **Canceled** when a canceled advance payment ID is received for it.

To import canceled advance payment IDs using the file channel, run the **Import Cancel Advance Payment ID** task on the **Data Import/Export** page in Ariba Administrator.

In case of the Web Services channel, cancel advance payments IDs are imported in response to the exported cancel advance payments and you do not need to manually invoke any task.

Import Cancel Advance Payment ID: Sample File

The table below provides descriptions and sample values for fields in the `CancelAdvancePaymentIDImport.csv` file required to import IDs of advance payments canceled in the external systems. This file can be used for sites integrated with all ERP systems.

Field Name	Required	Type and Description	Sample Value
`ADVANCEPAYMENTID`	Yes	String. The unique identifier of the canceled advance payment document in SAP Ariba Buying solutions.	APMT10
`ERPPAYMENTID`	Yes	String. The unique identifier of the canceled advance payment document in the external system.	00005429
`FISC_YEAR`	Yes	Integer. The financial year of the canceled advance payment transaction.	2016

About Importing Cancel Advance Payment Errors

This task imports the errors generated by the external ERP system while uploading advance payment documents marked for cancellation in the SAP Ariba Buying solutions.

Import Cancel Advance Payment Errors: Sample File

The **Import Cancel Advance Payment Errors** task uses the `CancelAdvancePaymentErrorImport.csv` file to import the errors generated while uploading the cancelled advance payments.

The following table provides the descriptions and sample values for the fields in the `CancelAdvancePaymentErrorImport.csv` file for integration with SAP ERP:

Field name	Required	Description	Sample value
`DATETIME2`	Yes	Date. The date of creation of the error record.	2016-08-15
`MESSAGE`	Yes	String. The description of the error.	Fill in and try field.

Field name	Required	Description	Sample value
MSGNR	Yes	String. Defines the error number.	23
FLDNAME	Yes	String. The field that has an error.	PO Item
MSGID	Yes	String. The error message ID.	M10
ADVANCEPAYMENTID	Yes	String. The system-generated unique identifier of the advance payment document.	APMT-10
NUMINSET	No	Integer. The number of error records in a batch.	3
TYPE	Yes	String. The type of the transaction for which the error was generated.	Advance Payment

The following table provides the descriptions and sample values for the fields in the `CancelAdvancePaymentErrorImport.csv` file for integration with the PeopleSoft ERP:

Field name	Required	Description	Sample value
CreationDate	Yes	Date. The date of creation of the error record.	2016-08-15
Erp	No	String. The type of the ERP system.	PeopleSoft
ErrorCode	No	String. Error code assigned to the failed advance payment document.	1026
ErrorMessage	No	String. Description of the error.	AP_VCHRBLD results in error
AdvancePaymentID	Yes	String. The unique identifier of the advance payment document.	APMT-10
NumInSet	No	String. The number of error records in a batch.	0
RecordName	No	String. Name of the error table.	AP_VOUCHER

Field name	Required	Description	Sample value
`Type`	No	String. The type of the transaction for which the error was generated.	Advance Payment

The following table provides the descriptions and sample values for the fields in the `CancelAdvancePaymentErrorImport.csv` file for integration with the Generic ERP:

Field name	Required	Description	Sample value
`CreationDate`	Yes	Date. The date of creation of the error record.	2016-08-15
`Erp`	Yes	String. The type of the ERP system.	Oracle
`ParentTable`	Yes	String. The name of the table containing the error records.	AP_INVOICE_LINES_INTERFACE
`RejectLookupCode`	Yes	String. The description of the error.	Invalid accounting distribution.
`AdvancePaymentID`	Yes	String. The unique identifier of the advance payment document.	APMT-10
`NumInSet`	No	String. The number of error records in a batch.	0
`Type`	Yes	String. The type of the transaction for which the error was generated.	Invoice

About Importing Advance Payment Cancellation Reasons

The **Import Advance Payment Cancellation Reasons** task enables you to import the IDs and descriptions associated with the reasons for cancellation of advance payments. The reasons for cancellation imported by this task are listed on the **Cancel Advance Payment** popup on the advance payment page to enable users choose a reason for cancellation of an advance payment.

Import Advance Payment Cancellation Reasons: Sample File

The **Import Advance Payment Cancellation Reasons** task uses the `AdvancePaymentCancelReasonCodesImport.csv` file to list the reasons for cancellation of advance payments. The following table provides the descriptions and sample values for the fields in the `AdvancePaymentCancelReasonCodesImport.csv` file.

Field name	Required	Description	Sample value
REASON	Yes	String. A unique identifier associated with a description specifying the reason for cancellation of an advance payment.	3
ALTERNATE_POSTING	Yes	Boolean. Specifies whether the reason code requires a posting date.	False
TEXT	Yes	String. Describes the reason for cancellation of an advance payment.	Reversal in closed period.

About Exporting Advance Payment Cancellation Reasons

This task exports IDs and descriptions associated with the reasons for cancellation of advance payments to the external ERP system.

Export Advance Payment Cancellation Reasons: Sample File

The **Export Advance Payment Cancellation Reasons** task uses the following CSV files:

- `AdvancePaymentCancelReasonCodesExport_sap.csv` for SAP ERP
- `AdvancePaymentCancelReasonCodesExport_psoft.csv` for PeopleSoft ERP
- `AdvancePaymentCancelReasonCodesExport_generic.csv` for Generic ERP

The following table provides the descriptions and sample values for the fields used in the CSV files for the **Export Advance Payment Cancellation Reasons** task:

Field name	Required	Description	Sample value
REASON	Yes	String. A unique identifier associated with a description specifying the reason for cancellation of an advance payment.	3
ALTERNATE_POSTING	Yes	Boolean. Specifies whether the reason code requires a posting date.	No
TEXT	Yes	String. Describes the reason for cancellation of an advance payment.	Reversal in closed period.

Configuring Support for One-Time Vendor Invoicing

Non-PO Invoices for One-Time Vendors [page 245]

One-Time Vendors in SAP Ariba Solutions [page 246]

Changes to Invoicing Functionality When One-Time Vendor Invoicing Is Enabled [page 249]

Configuration Overview for One-Time Vendor Invoicing [page 251]

Configuring a One-Time Vendor in SAP Ariba Buying and Invoicing [page 256]

Configuring SAP Business Network to Support One-Time Vendor ICS Invoices [page 258]

Supplier and Payment Information on One-Time Vendor Invoices [page 258]

Multiple One-Time Vendor Invoices for the Same Supplier [page 260]

Preparing ICS Providers for One-Time Vendor Invoices [page 262]

ICS Exceptions for One-Time Vendor Invoices [page 268]

Configuring Approval Rules for One-Time Vendor Invoices [page 268]

Integration with External Systems [page 270]

Creating a Non-PO Invoice for a One-Time Vendor [page 273]

Non-PO Invoices for One-Time Vendors

SAP Ariba Buying and Invoicing sites integrated with SAP ERP support one-time vendor invoices.

Buyers sometimes make one-time purchases from suppliers that aren't in their established list of suppliers. For example:

- A department manager might purchase a cake from a local baker or rent bicycles for a team outing.
- A buyer might use a temporary supplier if the usual supplier can't provide the required goods.

With one-time vendor support, when a supplier sends the invoice for such a one-time purchase, Accounts Payable (AP) can enter and process the non-PO invoice without adding a new supplier record to their master data. All the payment information is stored directly on the invoice.

This feature is for non-PO invoices entered by the buyer's AP department and for paper invoices converted via the buyer's invoice conversion service (also known as SAP Ariba Open ICS). The feature is supported on sites integrated with SAP ERP.

For details about the workflow of one-time vendor invoices, see One-Time Vendors in SAP Ariba Solutions [page 246].

Prerequisites

One-time vendor invoices are supported on SAP Ariba Buying and Invoicing sites integrated with SAP ERP.

To enable this feature, have your Designated Support Contact (DSC) submit a case. SAP Support will follow up to complete the case. They'll configure the following to enable this feature on your site:

- Site configuration parameter `Application.Invoicing.AllowOneTimeVendorInvoices`

Restrictions

- You create one-time vendor records (also referred to as one-time vendor types) manually in Ariba Administrator rather than importing them.
- **Ship From** information can be entered and edited at the header level but not at the line level.
- One-time vendor credit memos and debit memos are not supported.
- This feature supports ICS invoices from the buyer's invoice conversion service (open ICS) but not from SAP Ariba invoice conversion services.

Related Information

Invoicing and payment process guide

One-Time Vendors in SAP Ariba Solutions

In SAP Ariba solutions, a one-time vendor is a special type of supplier record. It's for a group of suppliers with whom you conduct business only once or rarely.

The typical process for one-time vendor invoices is to have a type of vendor record specifically for one-time vendor invoices (for example, a vendor record called "One-time Expenses," or several vendors for different types of expenses, such as "Catering" and "Charitable Donations"). A one-time vendor record represents an account from which the ERP system pays one-time vendor invoices. When the invoice entry user creates an invoice and checks the one-time vendor option, additional fields let the user choose the type of one-time vendor and enter the actual supplier name and payment information. The supplier and payment information is used for payments and remittances as well as reporting, and the invoice is registered against the one-time vendor record. The invoice can then be routed for approval if required.

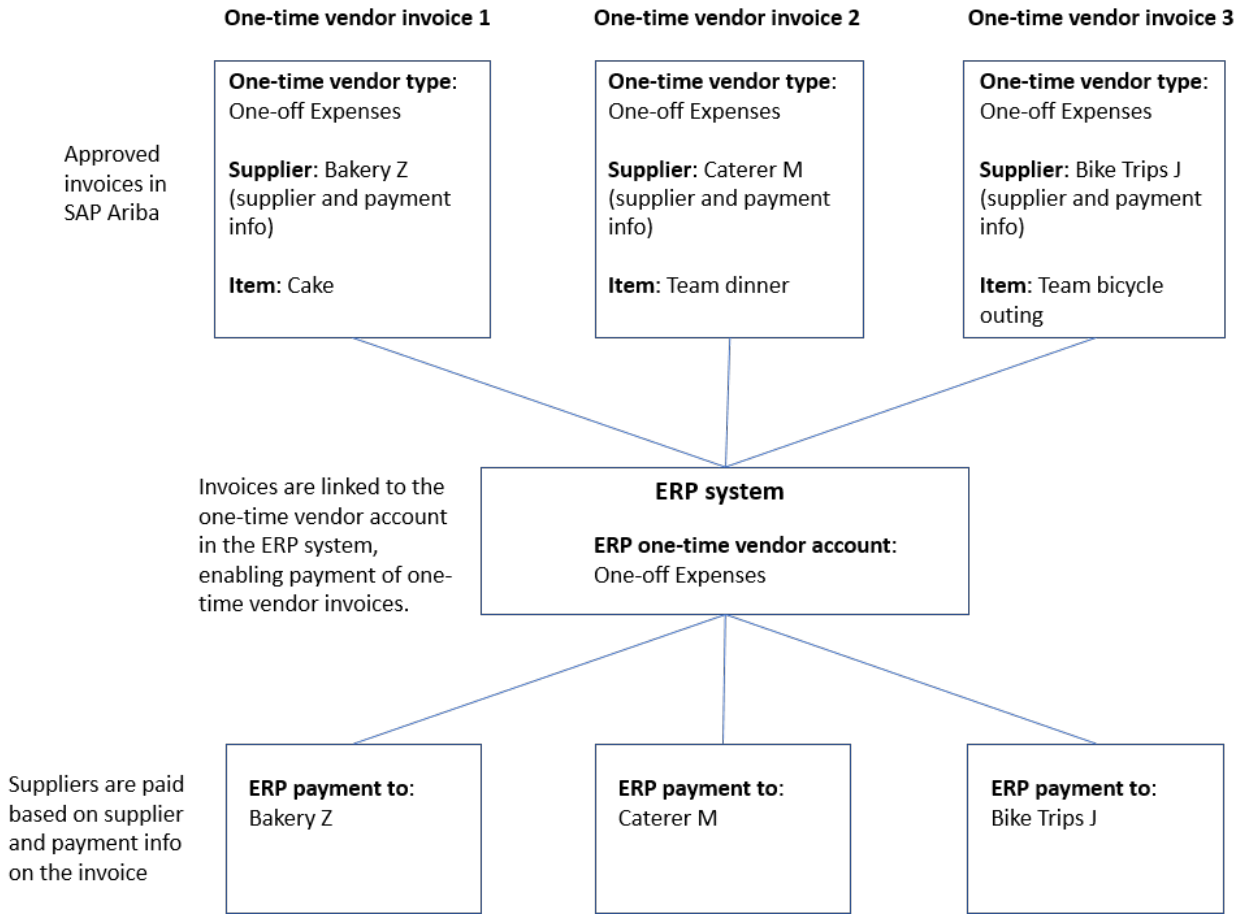
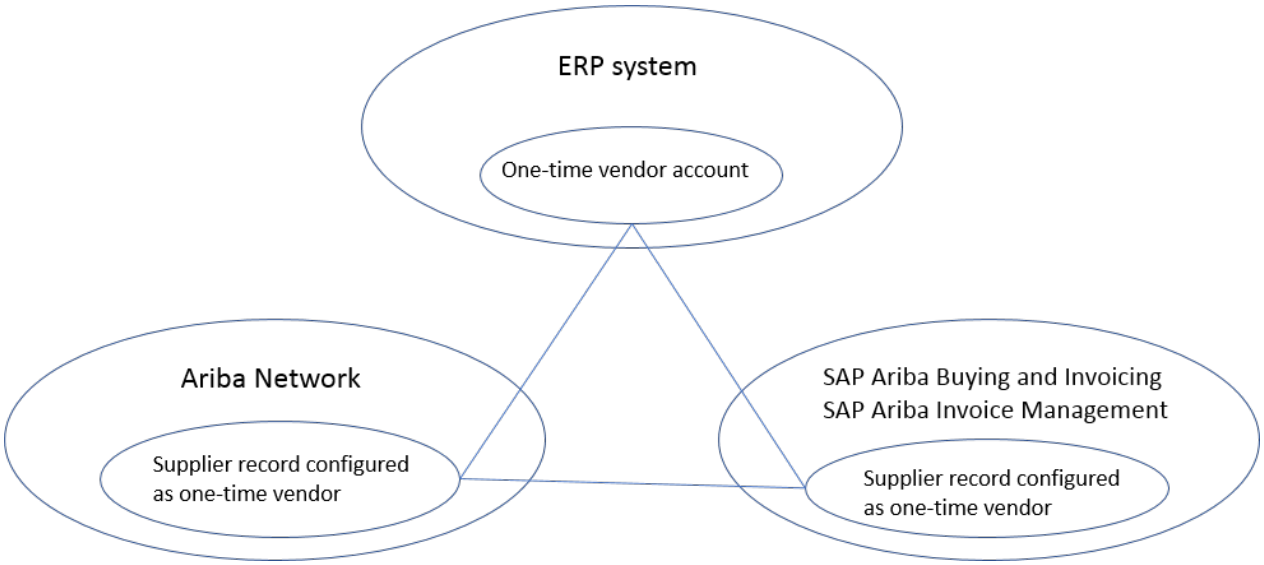


Figure 3: One-Time Vendor Invoices and Payments

In SAP Ariba, one-time vendors don't reference supplier-specific master data such as supplier name and payment information. Instead, users enter the supplier details directly on the invoice.

The one-time vendor record contains an ID that enables invoices to be routed to the appropriate one-time vendor account in your ERP system. One-time vendors are not actual suppliers. The suppliers aren't aware that the buyer is processing their invoices using SAP Ariba. They're just sending invoices and getting paid.

The ERP system, SAP Business Network, and SAP Ariba Buying and Invoicing must all have corresponding one-time vendor information for one-time vendor invoices and payments to flow correctly.



Supplier export from SAP Ariba Buying and Invoicing includes one-time vendor records.

Table 4: Regular Suppliers Versus the Individual Suppliers Listed on One-Time Vendor Invoices

	Regular supplier	Supplier on one-time vendor invoice
A record for the supplier exists in the buyer's ERP system master data and SAP Ariba solution as part of master data	Yes	No
Buyer's SAP Business Network account has a record for the supplier	Yes	No
Buyer has an SAP Business Network trading relationship with the supplier	Yes	No
Suppliers can be notified automatically when invoices are rejected	Yes	No
Payment information (for example, Remit To addresses, bank information) is master data associated with the supplier	Yes	No Payment information for the supplier exists only on the invoice.
Payment information is included in payment exports from the SAP Ariba solution so the ERP system can pay the supplier	Yes Payment information comes from the supplier record.	Yes Payment information comes from the invoice fields.
Invoice information can be included in reports	Yes	Yes

Related Information

Configuration Overview for One-Time Vendor Invoicing [page 251]

Configuring a One-Time Vendor in SAP Ariba Buying and Invoicing [page 256]
Configuring SAP Business Network to Support One-Time Vendor ICS Invoices [page 258]

Changes to Invoicing Functionality When One-Time Vendor Invoicing Is Enabled

Turning on support for one-time vendor invoicing results in several changes in the user interface, reporting, and other functionality related to invoicing.

User Interface Changes

The **Invoice Entry** page includes the following changes:

- A **One-time Vendor** checkbox is displayed when the **Non-PO** invoice type is selected.
  When the invoice is not in edit mode, the checkbox is replaced with **One-time Vendor: Yes**, and the field is displayed only for one-time vendor invoices.
- When **One-time Vendor** is checked:
  - A **One-time Vendor Type** chooser is added. It contains a list of one-time vendor records. (As with all choosers, before the chooser opens, a list of the most recently used choices appears. Even though the chooser contains only one-time vendor records, the list of most recent choices might also include regular suppliers.)
  - The **Supplier**, **Supplier Contact**, and **Supplier Location** fields are removed.
  - A **Supplier Name** text field is added.
  - A **Supplier Information** section is added, with **Remit To** and **Ship From** sections for entering supplier address information.
  - A **Payment Information** section is added. Only users in the **Edit One Time Vendor Invoice Payment Information** group can edit the **Payment Information** section. Users who can view invoices can see the section on saved and submitted invoices for users who can view invoices. The **Bank Account ID** and **IBAN ID** fields, if used, are masked when the payment information is not editable (when the invoice isn't in edit mode or the user isn't authorized to edit payment information).
    In the **Payment Information** section, the **Payment Method** field is required before the invoice can be fully approved. You can have SAP Ariba customize your site so that the payment method determines which payment fields are displayed. For example, if the selected payment method is Wire, bank account information is required. A payment method of Check might require only a remit-to address.

The one-time vendor information provided on the **Invoice Entry** tab or through the buyer's invoice conversion service (SAP Ariba Open ICS) is available on the related IR document and payment request.

In searches for invoices, IR documents, and payment requests:

- You can use the **Supplier** search criteria to find invoices based on the value in the **One-time Vendor Type** field. The **Supplier** chooser in these searches includes one-time vendor records as well as regular suppliers.
- In search results, you can add the column **One-time Vendor Name**. This column displays the value that's in the **Supplier Name** field of one-time vendor invoices so you can see which invoices are for each supplier associated with a particular one-time vendor type.

To add the column, click the **Table Options** menu . Click **More** if you don't see **One-time Vendor Name** on the menu.

Changes to Printouts

Printouts for one-time vendor invoices and IR documents include one-time vendor information: an indicator that the invoice is for a one-time vendor (**One-time Vendor**: **Yes**), a **One-time Vendor Type** field, and a **Supplier** field that corresponds to the **Supplier Name** field of the invoice. The printouts don't include a **Supplier Location** field.

Administrator Interface Changes

The page for creating a supplier (the **Create Partitioned Supplier** page) includes the following new fields at the bottom:

- **One-time vendor**: Checkbox to indicate whether the supplier is to be used for one-time vendor invoices.
- **Invoices allowed per supplier**: A limit for the number of invoices that can be submitted for a supplier using this one-time vendor. This field is visible if **One-time vendor** is checked.
- **When maximum number of invoices is exceeded**: Determines whether users can submit invoices for a supplier even though the maximum number of invoices has been reached for that supplier. This field is visible when **One-time vendor** is checked.
  - **Show warning**: Informs the user that their invoice exceeds the maximum number. Allows the user to submit the invoice anyway.
  - **Show error**: Informs the user that their invoice exceeds the maximum number. Prevents the user from submitting the invoice.

Changes to Groups

This feature adds a new group, **Edit One Time Vendor Invoice Payment Information**. Users who belong to this group can enter and edit payment information on one-time vendor invoices. Users who don't belong to this group can't edit payment information on one-time vendor invoices. The group **Edit One Time Vendor Invoice Payment Information** requires the permission to create invoices.

By adding users to the **Edit One Time Vendor Invoice Payment Information** group, you can implement a process where one person enters basic invoice information (supplier, line items) and submits the invoice, and an authorized approver then adds the payment information. In this way, buyers can separate the duties of the person keying in the transaction and the person entering the supplier bank details.

> **Note**
>
> Regardless of whether the approval flow includes approvers from the **Edit One Time Vendor Invoice Payment Information** group, users can submit invoices without adding payment information. However, one-time vendor invoices require payment information before they can be fully approved. The invoice must have a payment method and any other payment-related fields your site requires for one-time vendor invoices. At some point, a member of the **Edit One Time Vendor Invoice Payment Information** group must enter the information.

Changes to Reporting

This feature adds a new fact, **One Time Vendor**.

The **Invoice (Procurement and Invoicing)** fact:

- Includes one-time vendor invoices as well as regular invoices. Prepackaged reports built from the **Invoice (Procurement and Invoicing)** fact include one-time vendor invoices.
- Can be joined with the **One Time Vendor** fact to include one-time vendor invoice amounts and counts separately from the combined invoice amount and counts already included in the **Invoice (Procurement and Invoicing)** fact. For example, if the report shows **[INV-P&I] Invoice Count** is 100 and **[OTV] Number of Invoices** is 20, you know that 20 of the 100 invoices are one-time vendor invoices.

The **Supplier** dimension includes the following new fields:

- `IsOneTimeVendorShell`: Corresponds to the supplier checkbox **One-time vendor**, which indicates whether the supplier is to be used for one-time vendor invoices.
- `MaxInvoiceCount`: Corresponds to the supplier field **Invoices allowed per supplier**, which sets a limit for the number of invoices that can be submitted for a supplier using the one-time vendor.
- `IsHardLimit`: Indicates whether the supplier field **When maximum number of invoices is exceeded** is set to **Show error** (a hard limit preventing users from submitting invoices) or **Show warning** (doesn't prevent users from submitting invoices).
  This field is visible when **One-time vendor** is checked.

Changes to Data Import and Export

Payment request export includes one-time vendor information if present. For details, see Integration with External Systems [page 270].

cXML Changes

The following extrinsics under the `InvoiceDetailRequestHeader` element are for one-time vendor invoices:

- `IsOneTimeVendorInvoice`: Holds a Boolean value to distinguish one-time vendor invoices (a Yes value) from regular invoices.
- `OneTimeVendorSupplierName`: Holds the name of the supplier the invoice is from.

Configuration Overview for One-Time Vendor Invoicing

This table lists the high-level steps for configuring one-time vendor invoicing.

An SAP Ariba representative performs some of these steps for you, as indicated in the **Step** column of the table.

Table 5: Configuration Steps for One-Time Vendor Invoicing

Step	Description and considerations	Topics with more information
Implementation planning		
Plan your configuration	Address these questions: • Which supplier or payment fields on the invoice do you want to use for supplier unique ID validation? • Which fields do you want to be displayed for each payment method? Which fields are mandatory for each payment method? • Do you need to have any custom fields added? • For each one-time vendor, how many invoices do you want to allow per individual supplier? When the limit is reached, do you want users to see a warning and be allowed to submit the invoice, or an error and be prevented from submitting the invoice? • Which users do you want to authorize to enter and edit payment information on one-time vendor invoices? • What modifications to the invoice approval process are necessary? • Which fields do you want to include in the export of payment information? For CSV exports, do you need changes to the column order? • What steps do you need to take to roll out the one-time vendor process to users and your invoice conversion service?	Multiple One-Time Vendor Invoices for the Same Supplier [page 260] Preparing ICS Providers for One-Time Vendor Invoices [page 262] Configuring Approval Rules for One-Time Vendor Invoices [page 268]
If you already have one-time vendor accounts in your ERP system, gather information about them so you can add the required one-time vendors to your SAP Ariba solution and SAP Business Network.	For each one-time vendor in your ERP system, collect the following information: • ERP vendor ID (unique ID to use for routing) • Name of one-time vendor	
Feature enablement		

Step	Description and considerations	Topics with more information
(SAP Ariba step) Turn on the parameter for one-time vendor invoicing.	An SAP Ariba representative sets the parameter `Application.Invoicing.AllowOneTimeVendorInvoices` to **Yes**.	
(Optional but recommended SAP Ariba step) Configure the fields to use for supplier validation, any additional custom fields in invoices and IR documents, and which fields to include (or exclude) in payment export tasks.	Have your designated support contact log a service request. An SAP Ariba representative will follow up to complete the request.	Multiple One-Time Vendor Invoices for the Same Supplier [page 260]
One-time vendor data synchronization between ERP, SAP Ariba solution, SAP Business Network		
If not already done, configure one-time vendors in your ERP system.		
For each ERP one-time vendor, create a one-time vendor in the SAP Ariba solution.	In Ariba Administrator, create a supplier organization, supplier (partitioned supplier), and supplier location. Mark the supplier as a one-time vendor, and configure an ID that corresponds to the IDs in your ERP system. Map the supplier to a purchasing unit or purchasing organization if your site uses them, so that the one-time vendors users can work with are filtered appropriately.	Configuring a One-Time Vendor in SAP Ariba Buying and Invoicing [page 256]
Add the one-time vendors to your SAP Business Network buyer account.	Add a supplier record and mark it as a one-time vendor record. Configure a unique supplier ID (private ID) for each one-time vendor.	Configuring SAP Business Network to Support One-Time Vendor ICS Invoices [page 258] Supplier IDs
Assign SAP Business Network IDs in the SAP Ariba solution.	Add the one-time vendor SAP Business Network IDs to the one-time vendor records in the SAP Ariba solution.	Adding, Updating, and Deleting Suppliers
ERP integration		
Configure integration mappings.	Map one-time vendor invoice fields to fields in your ERP system for payment export.	Integration with External Systems [page 270]
Enable export tasks to include any new fields added to support one-time invoicing.	A member of the **Integration Admin** group accepts changes to the data definition for affected integration tasks.	Reviewing and Accepting Data Definition Changes to Export and Import Tasks

Step	Description and considerations	Topics with more information
User permissions		
Give users permission to enter and edit payment information on one-time vendor invoices.	Members of the **Customer Administrator** or **Customer User Admin** group add users to the group **Edit One Time Vendor Invoice Payment Information**. Consider whether to add members of the **Invoice Conversion Specialist** group to this group.	Managing User Groups in Ariba Administrator
ICS support		
Give the open ICS provider the information they need to process one-time vendor invoices.	Open ICS providers need to know following: • IDs of your one-time vendors • The cXML fields specific to one-time vendor invoices	Preparing ICS Providers for One-Time Vendor Invoices [page 262]
Approval process configuration		
Update the invoice approval process	If some users can create one-time vendor invoices but can't enter payment information, create an approval rule that adds the group **Edit One Time Vendor Invoice Payment Information** when necessary. Modify the invoice approval process for ICS invoices as needed. Add other approval rules as required by your one-time vendor process.	Configuring Approval Rules for One-Time Vendor Invoices [page 268]
Communicate the one-time vendor process to users		

Step	Description and considerations	Topics with more information
Inform users who enter non-PO invoices	Let invoice entry users know: • About the process and UI changes for one-time vendor invoices • What happens if they can't enter payment information • What to do if they get a warning or error that the maximum number of invoices has been reached for the supplier • What do to if they get a warning that the specified supplier name differs from the name on the most recently submitted invoice for that supplier (based on the unique identifier configured for one-time vendor invoices, such as supplier VAT ID)	Changes to Invoicing Functionality When One-Time Vendor Invoicing Is Enabled [page 249] Creating a Non-PO Invoice for a One-Time Vendor [page 273] Multiple One-Time Vendor Invoices for the Same Supplier [page 260]
Inform invoice conversion specialists	Let invoice conversion specialists know: • The process for one-time vendor invoices • The names and IDs of one-time vendors • That invoices are routed to them if the maximum number of invoices has been reached, but they won't see the warning or error until they save or submit the invoice • What do to if they get a warning that the specified supplier name differs from the name on the most recently submitted invoice for that supplier (based on the unique identifier configured for one-time vendor invoices, such as supplier VAT ID) • What to do if they can't edit payment information but errors say that changes are needed (for example, a required IBAN is missing, but the invoice conversion specialist can't enter it because they don't belong to the group **Edit One Time Vendor Invoice Payment Information**)	ICS Exceptions for One-Time Vendor Invoices [page 268] Preparing ICS Providers for One-Time Vendor Invoices [page 262]

Step	Description and considerations	Topics with more information
Inform SAP Business Network buyer administrators	Let SAP Business Network buyer administrators know: • The names and IDs of one-time vendors • The process for adding one-time vendors in the future (requires an SAP Ariba representative)	One-Time Vendors in SAP Ariba Solutions [page 246] Configuring SAP Business Network to Support One-Time Vendor ICS Invoices [page 258]

Configuring a One-Time Vendor in SAP Ariba Buying and Invoicing

Use this procedure to create a one-time vendor record for a group of suppliers with whom you conduct business only once or rarely. You can manually add a supplier in Ariba Administrator and mark it as a one-time vendor.

Prerequisites

- One-time vendor invoicing is supported on sites integrated with SAP ERP.
- Your site must be configured to allow one-time vendor invoices. (The parameter `Application.Invoicing.AllowOneTimeVendorInvoices` must be set to **Yes**.)
- Members of the **Customer Administrator** group can perform this task.
- Obtain the one-time vendor ID used in your ERP system and the Ariba Network ID of the one-time vendor record. One-time vendor invoices sent to the ERP system include the ERP vendor ID so the ERP system can assign the invoice to the correct one-time vendor.

Procedure

1. From the dashboard, choose **Manage** > **Core Administration**.
2. Choose **Supplier Manager** > **Suppliers**.
3. Perform one of the following procedures:
   - If you don't already have a record for the one-time vendor, create one:
     1. Click **Create New**.
        The **Create Common Supplier** page opens.
     2. Enter a name.
     3. On the **Organization IDs** tab, create an organization ID of type `networkid` for the Ariba Network ID of the one-time vendor record.

    4. Go to the **Partitioned Suppliers** tab, and click **Create New**.
    5. Edit the name if you want.
  - If a record for the one-time vendor already exists in your system (from a supplier import, for example):
    1. Find the record for the one-time vendor, and click **Actions** > **Edit**.
    2. On the **Organization IDs** tab, create an organization ID of type `networkid` for the Ariba Network ID of the one-time vendor record.
    3. Go to the **Partitioned Suppliers** tab.
4. Scroll to the bottom of the **Profile** tab for the `partitioned` supplier, and check **One-time vendor**.

   Additional fields appear.
5. In the **Invoices allowed per supplier** field, enter the number invoices allowed per supplier using this one-time vendor.

   If you don't enter a value in the field, the limit of invoices is 0. If you don't want to limit the number of invoices, enter -1.
6. In the field **When maximum number of invoices is exceeded**, determine what happens when a user tries to submit an invoice after the maximum number of invoices allowed has been reached:
   - **Show warning**: Informs the user that the maximum number of invoices is exceeded, but lets the user submit the invoice anyway.
   - **Show error**: Informs the user that the maximum number of invoices is exceeded, and prevents the user from submitting the invoice.
7. Go to the **Organization IDs** tab for the partitioned supplier.
8. Go to the **Supplier Locations** tab for the partitioned supplier.
9. Click **Create New**.

   The **Create Supplier Location** page opens.
10. Enter a name.
11. Enter the vendor ID from your SAP ERP in the **Unique Name** field, click **OK**, and save any unsaved supplier-related information you entered.

Results

Users who enter one-time vendor invoices can specify the one-time vendor in the **One-time Vendor Type** field of the invoice.

> **Note**
>
> One-time vendor records are included in the integration task for exporting suppliers. Because the **One-time vendor** field of the supplier record is not exported, it's not possible to distinguish one-time vendor records from regular supplier records in the export file.

Related Information

One-Time Vendors in SAP Ariba Solutions [page 246]

Configuring SAP Business Network to Support One-Time Vendor ICS Invoices

For invoices from your invoice conversion service (ICS invoices), one-time vendor invoices are routed through SAP Business Network. To make this possible, your SAP Business Network buyer account must include records for your one-time vendors.

One-time vendor records on SAP Business Network are not actual suppliers, just as they are not actual suppliers in SAP Ariba Procurement solutions. There is no trading relationship. One-time vendor records on SAP Business Network simply enable the transaction flow of one-time vendor invoices.

For each one-time vendor record on SAP Business Network, assign a unique supplier ID (also known as a private ID).

> **Note**
>
> On SAP Business Network, there's nothing on a supplier record that indicates whether the record is for a one-time vendor. One-time vendor invoices count toward the invoice threshold used in ICS quick enablement.

The information in the one-time vendor record, along with one-time vendor transactional details, can be deleted when the buyer terminates its contract with SAP Ariba.

Supplier and Payment Information on One-Time Vendor Invoices

Supplier and payment information for one-time vendor invoices is stored directly on the invoice rather than as part of a supplier record.

Supplier Information

Administrators work with SAP Ariba to determine:

- Which fields to display on the **Invoice Entry** page
- Whether to include **Ship From** and custom fields in exports of payment information

> **Note**
>
> When you're ready to include supplier information from one-time vendor invoices in payment exports, a member of the **Integration Admin** group must first accept the changes to the data definition.

The following **Supplier Information** fields are included on one-time vendor invoices by default.

- Supplier name (always required)
- Ship From address

- Remit To address

You can have SAP Ariba add custom fields to the **Supplier Information** section of one-time vendor invoices. The fields are added to an internal class called `OneTimeVendorDetail` and are configured to allow editing on one-time vendor invoices.

Payment Information

Only users who belong to the group **Edit One Time Vendor Invoice Payment Information** can enter and edit payment information on one-time vendor invoices.

Depending on your configuration, the payment method on the invoice can determine which payment-related fields are required. Administrators work with SAP Ariba to determine:

- Which fields to display on the **Invoice Entry** page for each payment method
- Which fields are mandatory for each payment method
- Whether to include all the fields or a subset of the fields in exports of payment information

> **Note**
>
> When you're ready to include payment information from one-time vendor invoices in payment exports, a member of the **Integration Admin** group must first accept the changes to the data definition.

The following payment-related fields are included on one-time vendor invoices by default. SAP Ariba can hide fields you don't want to use.

- Remit To address
- Payment method
- Bank name
- Bank branch
- Bank account type
- Bank account ID type
- Bank account ID
- Bank ID type
- Bank ID
- Bank address
- Supplier VAT ID (displayed if customized to allow editing on one-time vendor invoices)

You can have SAP Ariba add custom fields in the Payment Information section of one-time vendor invoices. Here are some examples:

- Bank ABA routing number
- Name on bank account
- International Bank Account Number (IBAN)
- Bank Identifier Code (BIC)
- SwiftID
- Supplier Tax ID

Make sure to provide guidance to invoice entry users and your invoice conversion service provider regarding the information required on one-time vendor invoices.

Multiple One-Time Vendor Invoices for the Same Supplier

SAP Ariba solutions check one-time invoices to see if any invoices were previously submitted for the same supplier. Then the system checks for supplier name consistency, duplicate invoice numbers, and whether the maximum number of invoices has been reached for that supplier.

The following sections explain the process of checking supplier information.

Unique Identifiers for Suppliers on One-Time Vendor Invoices

To check the names of suppliers on one-time vendor invoices, the SAP Ariba solution checks the invoice fields your organization decided to use as unique identifiers for the suppliers. An SAP Ariba representative configures these fields based on requirements your organization provides during implementation of the one-time vendor invoice feature. Some buyers might choose to use the supplier VAT ID, bank account ID, or IBAN on the invoice. Others might choose to use supplier address information or custom fields. The fields you choose are used for all one-time vendor invoices in your site.

> **Note**
>
> If you don't have a unique ID field configured for suppliers on one-time vendor invoices, the value in the **Supplier Name** field of the invoice is used for the check. However, because supplier names are not necessarily unique and are subject to change, we strongly recommend configuring unique IDs.

You can have SAP Ariba configure multiple unique identifiers to cover different situations. For example, you might want the VAT ID to be the unique identifier in some cases and the Tax ID to be the unique identifier in other cases. Users entering an invoice for a supplier in Europe can enter the VAT ID, and users entering an invoice for a supplier in the United States can enter the Tax ID. The SAP Ariba solution uses an "or" condition when checking for unique identifiers on the invoice. The check for unique identifier passes if the invoice has either a VAT ID **or** a Tax ID.

SAP Ariba configures an invoice field as a unique ID by setting an internal property called `Use as unique identifier for one-time vendor suppliers`.

An invoice can't be approved unless it has a unique identifier for the supplier (or a supplier name, if no unique identifier is configured).

Maximum Number of Invoices Allowed per Supplier

When you configure a one-time vendor (which involves creating a supplier record that corresponds to a one-time vendor account in your ERP system), you determine how many invoices can be created on behalf of each supplier.

For example, suppose you have a one-time vendor type called Charitable Donations, and the maximum number of invoices for each supplier (that is, for each charitable organization) is 2. If two Charitable Donation invoices have been submitted for supplier ABC NGO, if someone tries to submit a third invoice for ABC NGO, an error or warning is displayed, depending on how the Charitable Donation one-time vendor is configured.

> **Note**
>
> Let users know what to do when they see an error or warning about the maximum number of invoices. For example, you might have a process for users to request that a supplier be added to your vendor master list.

Setting a maximum limit lets you keep track of the transactions involving suppliers who aren't part of your organization's vendor master list. The limit you set depends on level of risk you're willing to take with these suppliers.

Validation for the number of invoices created is based on the one-time vendor used (specified in the **One-time Vendor Type** field of the invoice) combined with the value in the field used as unique ID for suppliers on one-time vendor invoices.

> **Note**
>
> An invoice isn't counted toward the limit until the field configured as the unique ID for suppliers is filled in. If the unique ID is a payment field, and your organization has a 2-step one-time vendor invoice process that calls for an approver (not a preparer) to fill in payment information, and the unique ID is a payment field, the invoice is not counted toward the maximum limit until the approver fills in the unique ID. Consider whether it's best to configure a non-payment field as the unique ID to avoid issues with the limit.

Here's a detailed example illustrating how the maximum limit works:

1. Your site has a one-time vendor called One-time Catering. For One-time Catering, the maximum invoices allowed per supplier is set to 3. The unique identifier for suppliers on one-time vendor invoices is Tax ID.
2. A department purchases a catered lunch from Nellie's Deli and enters a one-time vendor invoice. The invoice identifies Nellie's Deli (Tax ID 123456789) as the supplier and references the one-time vendor called One-time Catering.
3. As soon as the user enters the Tax ID, the SAP Ariba solution checks the invoice against the maximum count setting.
4. Two other departments also enter One-time Catering invoices for Nellie's Deli using the same Tax ID. They both pass the check against the maximum count setting. At this point, the maximum of 3 invoices is reached for Nellie's Deli.
5. Someone tries to enter a fourth invoice for Nellie's Deli using the same Tax ID.
6. Validation reveals that the fourth invoice exceeds the maximum number of invoices allowed for a single supplier under One-time Catering.
7. Depending on how One-time Catering is configured, the SAP Ariba solution displays a warning or an error saying that the maximum has been exceeded. In the latter case, the invoice can't be submitted.

Invoices with the following statuses don't count toward the maximum limit: **Denied**, **Canceled**, **Rejected**.

Duplicate Invoice Numbers

When a one-time vendor invoice is entered, the SAP Ariba solution checks for duplicate invoice numbers. If previously entered invoices reference the same unique ID for the supplier, the SAP Ariba solution checks the invoice

number on the previous invoices to make sure the invoice number on the new invoice is unique. If a duplicate invoice number is found, an error message is displayed. The invoice can't be submitted.

The check for duplicate invoice numbers considers **all** one-time vendor invoices, not just those that have the same value in the **One-time Vendor Type** field.

The check for for duplicate one-time vendor invoice numbers doesn't take the year into account. It checks all invoices ever created with the same one-time vendor and same unique ID for the supplier.

Supplier Name Inconsistencies

When a one-time vendor invoice is entered, if an invoice already existed referencing the same one-time vendor and the same unique ID for the supplier, the SAP Ariba solution checks whether the supplier name is consistent between the latest existing invoice and the new invoice.

This check is to prevent inadvertent differences in supplier names. However, different supplier names with the same unique ID **are** allowed. Such a situation can happen if, for example, a business changes its name or is acquired by a another company, but the value your site uses as the unique ID doesn't change.

If the supplier names are different, a message lets the user know. The user can choose to either keep the supplier name they entered or use the one that was previously used.

Here's an example illustrating the check for supplier name consistency:

1. There's a one-time vendor called One-time Catering. The unique identifier for suppliers is Tax ID.
2. A user submits a one-time vendor invoice for Nellie's Deli, Tax ID 123456789, referencing the one-time vendor called One-time Catering.
3. Validation reveals a previous one-time vendor invoice referencing One-time Catering with Tax ID 123456789, but the supplier name on that invoice is Nellie's Cafe.
4. A message is displayed, informing the user of the name difference.
5. The user chooses to change the supplier name to Nellie's Deli or keeps the name Nellie's Cafe.

Preparing ICS Providers for One-Time Vendor Invoices

One-time vendor invoices sent from your invoice conversion service (ICS invoices) require additional information for proper processing.

The following information is required in ICS cXML invoices:

- One-time vendor ID
- Name of the supplier who sent the invoice
- An indicator that the invoice is a one-time vendor invoice
- Payment information

> **Note**
>
> One-time vendor invoices count toward the invoice threshold used in ICS quick enablement.

Supplier Credentials in One-Time Vendor ICS Invoices

Identify the one-time vendor record in the `From` element of the header using the Ariba Network ID of the one-time vendor record.

Identify the name of the supplier who sent the invoice using the `OneTimeSupplierName` extrinsic.

Extrinsics

For ICS invoices, include the following extrinsics in the `InvoiceDetailRequestHeader` element.

`<Extrinsic name="OneTimeSupplierName">Bakery Z</Extrinsic>`	The name of the supplier who sent the invoice.
`<Extrinsic name="IsOneTimeVendorInvoice">true</Extrinsic>`	Indicates that the invoice is a one-time vendor invoice.

Payment Information

Depending on your configuration, the payment method on the invoice can determine which payment-related fields are required. It's a good idea to let the ICS provider know what payment information you expect one-time vendor invoices to include.

ICS Exception Handling for One-Time Vendor Invoices

Members of the **Invoice Conversion Specialist** group can edit ICS one-time vendor invoices. However, they can't enter or edit payment information unless they also belong to the group **Edit One Time Vendor Invoice Payment Information**.

If any users belong to the **Invoice Conversion Specialist** group, consider adding them to the **Edit One Time Vendor Invoice Payment Information**, too, so they can correct payment information when necessary. The group **Edit One Time Vendor Invoice Payment Information** doesn't authorize creating invoices.

Inform the invoice conversion specialist of your organization's process for resolving errors involving one-time vendor invoice limits. If the maximum number of invoices has already been reached for a given supplier, and the limit is handled is with an error, additional invoices for the supplier have a status of **Submitted with Errors** in the SAP Ariba solution and are routed to the invoice conversion specialist.

Example: ICS Invoice cXML Excerpts

Sample Code

```
<cXML payloadID="invoiceOTV97" timestamp="2018-06-26T15:50:14-07:00">
    <Header>
        <From>
            <Credential domain="NetworkId">
                <Identity>AN02004473907</
Identity>
            </Credential>
            <Credential domain="buyersystemid">
                <Identity>Buyer_373813</
Identity>
            </Credential>
        </From>
        <To>
            <Credential domain="NetworkId">
                <Identity>AN71000002392</
Identity>
            </Credential>
        </To>
        <Sender>
            <Credential domain="NetworkID">
                <Identity>AN01000000001</Identity>
                <SharedSecret>welcome1a</SharedSecret>
                </Credential>
            <UserAgent>ICS 123</UserAgent>
        </Sender>
</Header>
    <Request deploymentMode="production">
        <InvoiceDetailRequest>
            <InvoiceDetailRequestHeader invoiceDate="2018-05-23T11:02:48+05:30"
invoiceID="ICSTestOTV97" invoiceOrigin="supplier" operation="new"
purpose="standard">
            <InvoiceDetailHeaderIndicator />
            <InvoiceDetailLineIndicator isAccountingInLine="yes" />
            <InvoicePartner>
               <Contact addressID="3000" role="billTo">
                  <Name xml:lang="en">New York</Name>
                  <PostalAddress name="default">
                     <Street>691 Broadway</Street>
                     <City>NEW YORK</City>
                     <State>NY</State>
                     <PostalCode>10001</PostalCode>
                     <Country isoCountryCode="US">United States</Country>
                  </PostalAddress>
                  <Phone name="work">cid:image001.png@01D3F2C3.812A3D90
                     <TelephoneNumber>
                        <CountryCode isoCountryCode="US">1</CountryCode>
                        <AreaOrCityCode>040</AreaOrCityCode>
                        <Number>227070</Number>
                     </TelephoneNumber>
                  </Phone>
                  <Fax name="work">
                     <TelephoneNumber>
                        <CountryCode isoCountryCode="US" />
                        <AreaOrCityCode />
                        <Number />
                     </TelephoneNumber>
                  </Fax>
               </Contact>
            </InvoicePartner>
            <InvoicePartner>
               <Contact addressID="AD1001" role="remitTo">
                  <Name xml:lang="en" />
```

```
                  <PostalAddress name="default">
                     <Street>123 West Ave</Street>
                     <City>San Francisco</City>
                     <State>CA</State>
                     <PostalCode>90001</PostalCode>
                     <Country isoCountryCode="US">United States</Country>
                  </PostalAddress>
               </Contact>
            </InvoicePartner>
            <InvoicePartner>
               <Contact addressID="" role="from">
                  <Name xml:lang="en" />
                  <PostalAddress name="default">
                     <Street>123 West Ave</Street>
                     <City>San Francisco</City>
                     <State>CA</State>
                     <PostalCode>90001</PostalCode>
                     <Country isoCountryCode="US">United States</Country>
                  </PostalAddress>
               </Contact>
            </InvoicePartner>
            <InvoicePartner>
                <Contact addressID = "US005" role = "soldTo">
                    <Name xml:lang = "en">CA Hqtrs.</Name>
                    <PostalAddress>
                        <Street>3000 Airport Boulevard</Street>
                        <City>San Francisco</City>
                        <State>CA</State>
                        <PostalCode>94128</PostalCode>
                        <Country isoCountryCode = "US">United States</Country>
                    </PostalAddress>
                    <Email>nobody@ansmtp.ariba.com</Email>
                </Contact>
            </InvoicePartner>
            <InvoicePartner>
                <Contact addressID="" role="wireReceivingBank">
                    <Name xml:lang="en">Deutsche Bank</Name>
                    <PostalAddress name="default">
                        <Street>567 East Ave</Street>
                        <City>San Francisco</City>
                        <State>CA</State>
                        <PostalCode>90006</PostalCode>
                        <Country isoCountryCode="US">United States</Country>
                    </PostalAddress>
               </Contact>
               <IdReference domain="swiftID" identifier="SWIFT123"/>
               <IdReference domain="ibanID" identifier="IBAN234"/>
               <IdReference domain="accountName" identifier="Bill's Bakery
Checkings"/>
               <IdReference domain="accountID" identifier="123456789"/>
               <IdReference domain="accountType" identifier="Checking"/>
               <IdReference domain="branchName" identifier="West Ave Branch"/>
            </InvoicePartner>
            <InvoiceDetailShipping>
                <Contact addressID="" role="shipFrom">
                    <Name xml:lang="en" />
                    <PostalAddress name="default">
                        <Street>123 West Ave</Street>
                        <City>San Francisco</City>
                        <State>CA</State>
                        <PostalCode>90001</PostalCode>
                        <Country isoCountryCode="US">United States</Country>
                    </PostalAddress>
               </Contact>
               <Contact addressID="3000" role="shipTo">
                  <Name xml:lang="en">New York</Name>
                  <PostalAddress name="default">
                     <Street>691 Broadway</Street>
```

```
                      <City>NEW YORK</City>
                      <State>NY</State>
                      <PostalCode>10001</PostalCode>
                      <Country isoCountryCode="US">United States</Country>
                   </PostalAddress>
                   <Phone name="work">
                      <TelephoneNumber>
                         <CountryCode isoCountryCode="US">1</CountryCode>
                         <AreaOrCityCode>040</AreaOrCityCode>
                         <Number>227070</Number>
                      </TelephoneNumber>
                   </Phone>
                   <Fax name="work">
                      <TelephoneNumber>
                         <CountryCode isoCountryCode="US" />
                         <AreaOrCityCode />
                         <Number />
                      </TelephoneNumber>
                   </Fax>
                </Contact>
             </InvoiceDetailShipping>
             <Extrinsic name = "paymentMethod">wire</Extrinsic>
             <Extrinsic name = "supplierVatID">cathyVATID1</Extrinsic>
             <Extrinsic name = "OneTimeSupplierName">Bill's Bakery</Extrinsic>
                <Extrinsic name = "IsOneTimeVendorInvoice">true</Extrinsic>
             <Extrinsic name="invoiceSourceDocument"/>
             <Extrinsic name="invoiceSubmissionMethod"/>
             <Extrinsic name="CompanyCode">3000</Extrinsic>
          </InvoiceDetailRequestHeader>
          <InvoiceDetailOrder>
             <InvoiceDetailOrderInfo>
                <SupplierOrderInfo orderID="Unknown" />
             </InvoiceDetailOrderInfo>
             <InvoiceDetailItem quantity="100" invoiceLineNumber="1">
                <UnitOfMeasure>EA</UnitOfMeasure>
                <UnitPrice>
                   <Money alternateCurrency="" alternateAmount=""
currency="USD">5</Money>
                </UnitPrice>
                <InvoiceDetailItemReference lineNumber="0">
                   <ItemID>
                      <SupplierPartID/>
                   </ItemID>
                   <Description xml:lang="en-US">Layer cake</Description>
                   <Classification domain="unspsc">7313</Classification>
                </InvoiceDetailItemReference>
                <SubtotalAmount>
                   <Money alternateCurrency="" alternateAmount=""
currency="USD">100.00</Money>
                </SubtotalAmount>
                <GrossAmount>
                   <Money alternateCurrency="" alternateAmount=""
currency="USD">100.00</Money>
                </GrossAmount>
                <NetAmount>
                   <Money alternateCurrency="" alternateAmount=""
currency="USD">100.00</Money>
                </NetAmount>
                <Distribution>
                   <Accounting name="DistributionCharge">
                      <AccountingSegment id="100">
                         <Name xml:lang="en">Percentage</Name>
                         <Description xml:lang="en">Percentage</Description>
                      </AccountingSegment>
                      <AccountingSegment id="0000410000">
                         <Name xml:lang="en">GeneralLedger</Name>
                         <Description xml:lang="en">ID</Description>
                      </AccountingSegment>
```

```
                      <AccountingSegment id="0000002100">
                         <Name xml:lang="en">CostCenter</Name>
                         <Description xml:lang="en">ID</Description>
                      </AccountingSegment>
                   </Accounting>
                   <Charge>
                      <Money alternateCurrency=""alternateAmount=""
currency="USD">100.00</Money>
                   </Charge>
                </Distribution>
                <Comments />
            </InvoiceDetailItem>
         </InvoiceDetailOrder>
         <InvoiceDetailSummary>
            <SubtotalAmount>
               <Money alternateCurrency=""alternateAmount=""
currency="USD">100.00</Money>
            </SubtotalAmount>
            <Tax>
               <Money alternateCurrency=""alternateAmount=""
currency="USD">0.00</Money>
               <Description xml:lang="en">TotalTax</Description>
            </Tax>
            <SpecialHandlingAmount>
               <Money alternateCurrency=""alternateAmount=""
currency="USD">0.00</Money>
            </SpecialHandlingAmount>
            <ShippingAmount>
               <Money alternateCurrency=""alternateAmount=""
currency="USD">0.00</Money>
            </ShippingAmount>
            <GrossAmount>
               <Money alternateCurrency=""alternateAmount=""
currency="USD">100.00</Money>
            </GrossAmount>
            <InvoiceDetailDiscount>
               <Money alternateCurrency=""alternateAmount=""
currency="USD">0.00</Money>
            </InvoiceDetailDiscount>
            <TotalCharges>
               <Money alternateCurrency=""alternateAmount=""
currency="USD">0.00</Money>
            </TotalCharges>
            <NetAmount>
               <Money alternateCurrency=""alternateAmount=""
currency="USD">100.00</Money>
            </NetAmount>
            <DueAmount>
               <Money alternateCurrency=""alternateAmount=""
currency="USD">100.00</Money>
            </DueAmount>
         </InvoiceDetailSummary>
      </InvoiceDetailRequest>
   </Request>
</cXML>
```


Related Information

ICS Exceptions for One-Time Vendor Invoices [page 268]

Supplier and Payment Information on One-Time Vendor Invoices [page 258]

Configuring SAP Business Network to Support One-Time Vendor ICS Invoices [page 258]

ICS Exceptions for One-Time Vendor Invoices

One-time vendor invoices sent from buyer's invoice conversion service (SAP Ariba Open ICS) follow the same flow as standard ICS invoices. Invoices with errors or missing information have a status of **Submitted With Errors** and are routed to an invoice conversion specialist, who resolves the issues and resubmits the invoice.

> **Note**
>
> This flow requires that your SAP Business Network account and SAP Ariba solution be configured for ICS exception handling. For more information, see ICS Exception Handling in SAP Ariba Procurement solutions.

Here are some issues specific to validation of one-time vendor invoices:

- **Duplicate supplier invoice number**: An invoice with the same invoice number has already been registered in the SAP Ariba solution for that supplier.
- **Too many invoices for one supplier**: The number of invoices referencing the same one-time vendor and supplier exceeds the maximum allowed for the one-time vendor.
  Invoice conversion specialists can't see this issue by simply opening the invoice. The error or warning shows up when they try to submit the invoice.
- **Missing information**: Required supplier or payment fields don't have a value.

Related Information

Multiple One-Time Vendor Invoices for the Same Supplier [page 260]

Configuring Approval Rules for One-Time Vendor Invoices

You can create approval rules specific to one-time vendor invoices and invoice reconciliation (IR) documents.

For example, if your process for one-time vendor invoices calls for one person to create the invoice and a different person to add the payment information, you can create an approval rule that adds the **Edit One Time Vendor Invoice Payment Information** group to the approval flow. An approver from that group can add the payment information.

Use the invoice field `IsOneTimeVendorInvoice` to create a field match condition that distinguishes one-time vendor invoices from regular invoices.

- To add an approval rule specifically for one-time vendor invoices in the invoice approval process, add a **Field Match** condition and select the field `IsOneTimeVendorInvoice`.
- To add an approval rule for one-time vendor invoices in the IR approval process, add a **Field Match** condition, expand the `Invoice` field in the field selector, and select `IsOneTimeVendorInvoice`.

You can also create field match conditions using the `OneTimeVendorInvoiceDetail` fields.

Following are example use cases for one-time vendor invoice approval rules.

- If the users who create one-time vendor invoices aren't authorized to enter the payment information, create an approval rule that adds the group **Edit One Time Vendor Invoice Payment Information** to the approval flow. Such a rule says, "If the invoice preparer doesn't belong to the group **Edit One Time Vendor Invoice Payment Information**, add that group to the approval flow."
  1. In the approval process editor, insert a **None Is True** subcondition.
  2. Add the predefined condition **Preparer In Group**.
  3. Select the group **Edit One Time Vendor Invoice Payment Information**.
- To add an approval rule for one-time vendor invoices based on amount, combine an `Invoice.IsOneTimeVendorInvoice` field match with the predefined condition **Total Cost Exceeds Limit**.
- To add an approval rule specific to one-time vendor invoices from your invoice conversion service (ICS invoices), combine an `Invoice.IsOneTimeVendorInvoice` field match with the a field match for the `Invoice.InvoiceSubmissionMethod` value `PaperViaICS`.

Tips

- You might have approval rules based on supplier information on the invoice. Keep in mind that one-time vendor invoices don't reference regular supplier records.
  For example, a regular invoice has a supplier address that's automatically populated based on the specified supplier. One-time vendor invoices, on the other hand, have ad hoc address information, if any. Therefore, field matches for regular supplier-related fields don't work the same way with one-time vendor invoices.
- When creating field match conditions for one-time vendor invoices, pay attention to the **Field Label** column in the field selector. Look for fields whose labels match what you see on the page for one-time vendor invoices.
- The following table lists a few invoice fields buyers might choose to base approval rules on.

Field you want to base a condition on	Field in the field selector for invoices
All one-time vendor invoices	`IsOneTimeVendorInvoice`
All ICS one-time vendor invoices	combine an `Invoice.IsOneTimeVendorInvoice` field match with the a field match for the `Invoice.InvoiceSubmissionMethod` value `PaperViaICS`
One-time Vendor Type field on the invoice	`OnetimeVendorInvoiceDetail.Supplier`
The **Supplier Name** field on the invoice	`OnetimeVendorInvoiceDetail.SupplierName`

Field you want to base a condition on	Field in the field selector for invoices
Payment information on the invoice	The fields under `OneTimeVendorInvoiceDetail.SupplierPaymentLocation` **Note** In the field selector, the `SupplierBankInfo` fields are not applicable to one-time vendor invoices. They hold bank information associated with regular supplier records in your site.

- To add an approval condition based on payment method, create a field match condition using the field `OneTimeVendorInvoiceDetail.PaymentMethodType`.
- To add an approval condition based on the purchasing unit of the one-time vendor specified in the **One-time Vendor Type** field, create a field match condition using the field `OneTimeVendorInvoiceDetail.Supplier.ProcurementUnits`.
- To use one-time vendor invoice fields in field match conditions for invoice reconciliation (IR) documents, in the field selector, first expand the `Invoice` fields.

Integration with External Systems

Documents related to one-time vendor invoices include supplier and payment information so the ERP system can pay the suppliers and export the appropriate remittance information.

Payment information for one-time vendor invoices is included in the export of payment requests so your ERP system can pay the suppliers. Exported information includes:

- The value in the **One-time Vendor Type** field of the invoice
- The value in the **Supplier Name** field of the invoice
- The payment method
- Columns for the **Ship From** address (only if the **Ship From** fields are customized to be included in payment export)
- Fields for payment details such as bank information and **Remit To** addresses

SAP Ariba can configure the order of fields in the CSV export for you using the internal tasks `Export CSV Column Orders` and `Import CSV Column Orders`.

Remittance Import

Remittance imports can include remittance information for one-time vendor invoices. Make sure the supplier records for your one-time vendors are configured correctly so the SAP Ariba solution can map the remittance information to the correct one-time vendor record.

Payment Exports from SAP Ariba Buying and Invoicing

The following table lists fields added to the **Export Payment Requests** task for one-time vendor invoices. The table covers integration via web services and via CSV file channel. Using the file channel requires you to implement a custom integration.

You can have additional fields added to payment exports through customization.

Table 6: One-Time Vendor Info in Payment Exports via File Channel or Web Services

Field on one-time vendor invoice	Web services export element (under `OnetimeVendorInvoiceDetail` unless otherwise noted	CSV file export column (in `PaymentHeaderExport.csv`)
One-time Vendor Type	`Supplier`	`DIFF_INV`
Supplier Name	`SupplierName`	`VENDOR_NAME`
Street in the **Remit To** section	`Lines` **Note** All **Remit To** fields are under `SupplierRemitToAddress.PostalAddress`.	`REMIT_TO_STREET`
City in the **Remit To** section	`City`	`REMIT_TO_CITY`
State/Province/Region in the **Remit To** section	`State`	`REMIT_TO_STATE`
Country/Region in the **Remit To** section	`Country.UniqueName`	`REMIT_TO_COUNTRY`
Postal Code in the **Remit To** section	`PostalCode`	`REMIT_TO_POSTALCODE`
Payment Method	`PaymentMethodType` () **Note** This element is under `Payment_PaymentHeaderExportWS`, not `OnetimeVendorInvoiceDetail`.	`PYMT_METH`
Bank Name	`BankName` **Note** All bank info fields are under `SupplierPaymentLocation`.	`BANK_NAME`

Field on one-time vendor invoice	Web services export element (under `OnetimeVendorInvoiceDetail` unless otherwise noted	CSV file export column (in `PaymentHeaderExport.csv`)
Branch Name	`BranchName`	`BRANCH_NAME`
Bank Account Type	`BankAccountType.UniqueName`	`BANK_ACCOUNT_TYPE`
Bank Account ID Type	`BankAccountIDType.UniqueName`	`BANK_ACCOUNT_ID_TYPE`
Bank Account ID	`BankAccountID`	`BANK_ACCOUNT_ID`
Bank ID Type	`BankIDType.UniqueName`	`BANK_ID_TYPE`
Bank ID	`BankID`	`BANK_ID`
IBAN	`IBanID`	`IBAN_ID`
Street in the **Bank Address** section	`BankAddress.PostalAddress.Lines`	`BANK_STREET`
City in the **Bank Address** section	`BankAddress.PostalAddress.City`	`BANK_CITY`
State/Province/Region in the **Bank Address** section	`BankAddress.PostalAddress.State`	`BANK_STATE`
Postal Code in the **Bank Address** section	`BankAddress.PostalAddress.PostalCode`	`BANK_POSTALCODE`
Country/Region in the **Bank Address** section (Corresponds to the bank country key in SAP ERP.)	`BankAddress.PostalAddress.Country.UniqueName`	`BANK_COUNTRY`
The following fields in the **Ship From** section of one-time vendor invoices are included in `PaymentHeaderExport.csv` (file channel) and the `PaymentHeaderExport` element (web service) only if the fields are customized for inclusion in payment exports.		
Street in the **Ship From** section	`ShipFrom` element	`InvoiceReconciliation_OneTimeVendorInvoiceDetail_ShipFrom_STREET`
City in the **Ship From** section	`ShipFrom` element	`InvoiceReconciliation_OneTimeVendorInvoiceDetail_ShipFrom_CITY`

Field on one-time vendor invoice	Web services export element (under `OnetimeVendorInvoiceDetail` unless otherwise noted	CSV file export column (in `PaymentHeaderExport.csv`)
State in the **Ship From** section	`ShipFrom` element	`InvoiceReconciliation_OneTimeVendorInvoiceDetail_ShipFrom_STATE`
Country/Region in the **Ship From** section	`ShipFrom` element	`InvoiceReconciliation_OneTimeVendorInvoiceDetail_ShipFrom_COUNTRY`
Postal Code in the **Ship From** section	`ShipFrom` element	`InvoiceReconciliation_OneTimeVendorInvoiceDetail_ShipFrom_POSTALCODE`

Related Information

Supplier and Payment Information on One-Time Vendor Invoices [page 258]
Configuring a One-Time Vendor in SAP Ariba Buying and Invoicing [page 256]

Creating a Non-PO Invoice for a One-Time Vendor

When a supplier that you plan to use just once (or rarely) sends an invoice, and the supplier isn't in your organization's master vendor list, you can still enter and process the invoice if your site supports one-time vendor invoices.

Prerequisites

Buyer users who can create non-PO invoices can perform this task.

Members of the group **Edit One Time Vendor Invoice Payment Information** can enter and edit supplier payment information on the invoice. If you create a one-time vendor invoice and can't enter payment information, submit the invoice without the payment information. Depending on your site's invoice approval process, when you submit the invoice, it's routed to an approver who can enter payment information.

Context

Your organization determines the circumstances in which one-time vendor invoices are allowed.

Unlike regular invoices, with one-time vendor invoices, you enter supplier details directly on the invoice. An authorized user adds the payment information to the invoice.

Procedure

1. From the dashboard, click **Create** > **Invoice**.
2. In the **Type** field, choose **Non-PO**.

   A **One-time Vendor** checkbox is displayed.
3. Check **One-time Vendor**.

   A **One-time Vendor Type** chooser is displayed.
4. In the **One-time Vendor Type** chooser, search for and select the one-time vendor to use for this invoice according to your organization's process for one-time vendor invoices.
5. Fill in the fields in the **Supplier Information** section.

   Tips:
   - Make sure to enter the supplier name correctly.
   - The **Ship From** address is the same as the **Remit To** address by default. To enter a different **Ship From** address, under **Ship From**, uncheck **Same as Remit To address** and fill in the address fields.
6. If you're authorized to enter payment information (through the group **Edit One Time Vendor Invoice Payment Information**), fill in the information in the **Payment Information** section.

   If you're not authorized to enter payment information, the approval flow for the invoice will include someone who can do that for you.
7. Add the line items and any taxes, charges, and discounts.
8. Submit the invoice.

Results

The approval flow is generated, and the invoice is routed to the first approver. If no approvals are required, the invoice is automatically approved.

Invoicing and Payment Data CSV Files Reference

The invoicing and payment data CSV file reference table lists all CSV files, describes them, and indicates if they are required. A CSV file is required if the data it contains is required for SAP Ariba Procurement solutions to function properly. Unless otherwise noted, required files should not be empty.

File	Description	Required
`Accounting.csv`	Defines system-wide accounting information.	See Notes
`BankAccountIDType.csv`	Defines bank account number types.	See Notes
`BankAccountType.csv`	Defines all supported types of bank account types, such as checking accounts and savings accounts.	See Notes
`BankIDType.csv`	Defines bank identifier types.	See Notes
`BuyerPaymentBankLocation.csv`	Defines general bank payment information, such as bank names and addresses, for buyer bank accounts. It also associates each bank payment location with a bank identifier type, bank account type, and bank account number type.	See Notes
`ChargeType.csv`	Defines charge types that can be used to apply charges on invoices. It also defined the mapping between charge types used in SAP Ariba Procurement solutions and charge types used in SAP Business Network.	No
`ContractHeader.csv`	Defines header-level information for contracts.	See Notes
`ContractItemInformation.csv`	Defines line-level details for contracts.	See Notes
`DocumentType.csv`	This file is required only for sites integrated with SAP ERP systems. It defines the ERP document types that can be used for advance payments.	Yes, only for SAP variants
`GLIndicator.csv`	This file is required only for sites integrated with SAP ERP systems. It defines the ERP general ledgers that can be used for advance payments.	Yes, only for SAP variants
`InvoiceExceptionType.csv`	Defines invoice exception types. An invoice exception represents a single discrepancy between the data on the supplier invoice and the data in SAP Ariba Procurement solutions. Exceptions can represent a variety of issues, such as missing receipts, mismatched quantities or prices, duplicate invoices, or tax variances.	No
`InvoiceExceptionTypeValidation.csv`	Defines validation data for the invoice exception types defined in `InvoiceExceptionType.csv`. Invoice exception type validation data defines exactly which fields to compare and what tolerances to allow for each type of invoice exception.	No
`InvoiceExceptionTypeCCCOverride.csv`	Defines an invoice exception override driver for commodity codes.	Yes

File	Description	Required
`InvoiceExceptionTypeSupplierOverride.csv`	Defines an invoice exception override driver for suppliers.	Yes
`InvoiceExceptionTypeOverrideValidation.csv`	Defines which override driver to assign to an invoice exception, and specifies the override tolerance values, auto-rejection/auto-acceptance settings and the groups responsible to resolve invoice exception triggered by the override.	Yes
`InvoiceLimit.csv`	An approver lookup table that associates groups and invoice limits with either purchasing units or two ERP-specific "rule units." This file is read by the default invoice reconciliation approval rule called PO Invoice Cap Rule,	No
`InvoiceNonPOLimit.csv`	An approver lookup table that associates groups and non-PO invoice limits with either purchasing units or two ERP-specific 'rule units." This file is read by the default invoice reconciliation approval rule called Non-PO Invoice Cap Rule.	No
`InvoiceRejectionReason.csv`	Defines codes for the reasons users reject invoices. Applicable if the parameter **Require reason codes for invoice rejection** (`Application.Invoicing.UseInvoiceRejectionReasonCodes`) is set to **Yes**. If you enable and configure reason rejection codes, users who reject an invoice must select one or more rejection reasons from a list of rejection reasons you define.	No
`PaymentMethodType.csv`	Defines payment methods. A payment method defines how a buying organization makes a payment to a supplier. Typical payment methods include check, Automatic Clearing House (ACH) electronic transfer, wire transfer, and credit card (purchasing card).	Yes
`PaymentTerms.csv`	Contains payment term names and descriptions. Payment terms specify the time period in which a supplier requests payment and any discounts that the supplier chooses to make available for early payment. For example, payment terms of Net30 might mean that a payment is due within 30 days, but with a 10 percent discount available if payment is made within 15 days.	Yes
`PaymentTermStepDetails.csv`	Contains details about the specific discounts you define in `PaymentTermSteps.csv`. You can define several possible discount schedules.	Yes
`PaymentTermSteps.csv`	Contains installment levels for each payment term you define in `PaymentTerms.csv`.	Yes
`RemittanceLocation.csv`	Contains a list of remittance locations. When generating a payment request, the SAP Ariba invoicing solution uses the remittance location associated with that supplier location. Each supplier location must have its own remittance location.	Yes
`RemittanceLocationDetails.csv`	Defines the details of each remittance location specified in `RemittanceLocation.csv`.	Yes

File	Description	Required
`SLRemittanceInformation.csv`	Defines SAP Business Network payment information for supplier locations.	Yes
`SLRemittanceInformationDetails.csv`	Associates supplier locations with remittance locations. Each supplier location must have its own remittance location. You cannot associate multiple supplier locations with the same remittance location.	Yes
`SupplierPaymentBankLocation.csv`	Defines general bank payment information, such as bank names and addresses, for supplier bank accounts. It also associates each bank payment location with a bank identifier type, bank account type, and bank account number type.	See Notes
`TaxCodeHeader.csv` and `TaxCodeDetails.csv`	Defines tax codes that SAP Ariba Procurement solutions use to calculate tax rates.	Yes
`TaxCodeLookup.csv`	Defines the tax code lookup table.	Yes
`TaxCodeLookupFieldMapping.csv`	Defines the mapping between tax code lookup fields and line item fields.	Yes
`TaxComponent.csv`	Defines tax components that can assigned to tax codes, to apply taxes on invoices.	Yes
`TaxRateLookupByTaxCode.csv`	Defines the tax rate lookup by tax code table,	Yes
`TaxType.csv`	Defines tax types that can be used to apply taxes on invoices in SAP Ariba Procurement solutions.	Yes
`TaxTypeMap.csv`	Maps the common tax names found on invoices to tax-related procurement line types.	Yes
`TermPricing.csv`	Defines pricing terms for contracts.	See Notes.
`TieredPricing.csv`	Defines pricing tiers for contract items.	See Notes.
`WHTLookupFieldMapping.csv`	Defines the mapping between tax code lookup fields and line item fields for withholding taxes.	Yes

Note

- `BankAccountIDType.csv`, `BankAccountType.csv`, `BankIDType.csv`, `BuyerPaymentBankLocation.csv`, and `SupplierPaymentBankLocation.csv` are required only if you use the SAP Business Network payment model. If you use the External System payment model, bank information is optional.
- `ContractHeader.csv` and `ContractItemInformation.csv` are required for services invoicing features such as service invoice accumulation against contracts, supplier punch-in from SAP Business Network during service sheet creation, and service sheet validation. `Accounting.csv`, `TermPricing.csv`, and `TieredPricing.csv` are optional contract information that can be uploaded alongside contract header and item data to further define contract item pricing and accounting. See the Common Data Import and Administration for SAP Ariba Procurement Solutions for more information.

Cancellation of Approved Invoices in the ERP System

If an approved invoice is canceled or reversed in the ERP system (for example, if it was posted to the wrong account or with the wrong amount), the status of the invoice reconciliation (IR) document can be changed to **Rejected**.

- In SAP Ariba Invoice Management, IRs with a status of **Paying** or **Paying Failed** automatically switch to **Rejected** status when SAP Business Network sends a status update request saying the invoice has been rejected.
- In SAP Ariba Buying and Invoicing, an integration administrator for your organization sets up a web service to receive invoice cancellations from the ERP system.
- Alternatively, invoice administrators can manually force the IR to go from **Paying** or **Paying Failed** to **Rejected** using a **Force Reject** button.

  > **Note**
  >
  > When you click **Force Reject**, a warning that explains what the force reject action does and doesn't do is displayed. For more information, see Enhanced warnings for force actions on invoices and payments.

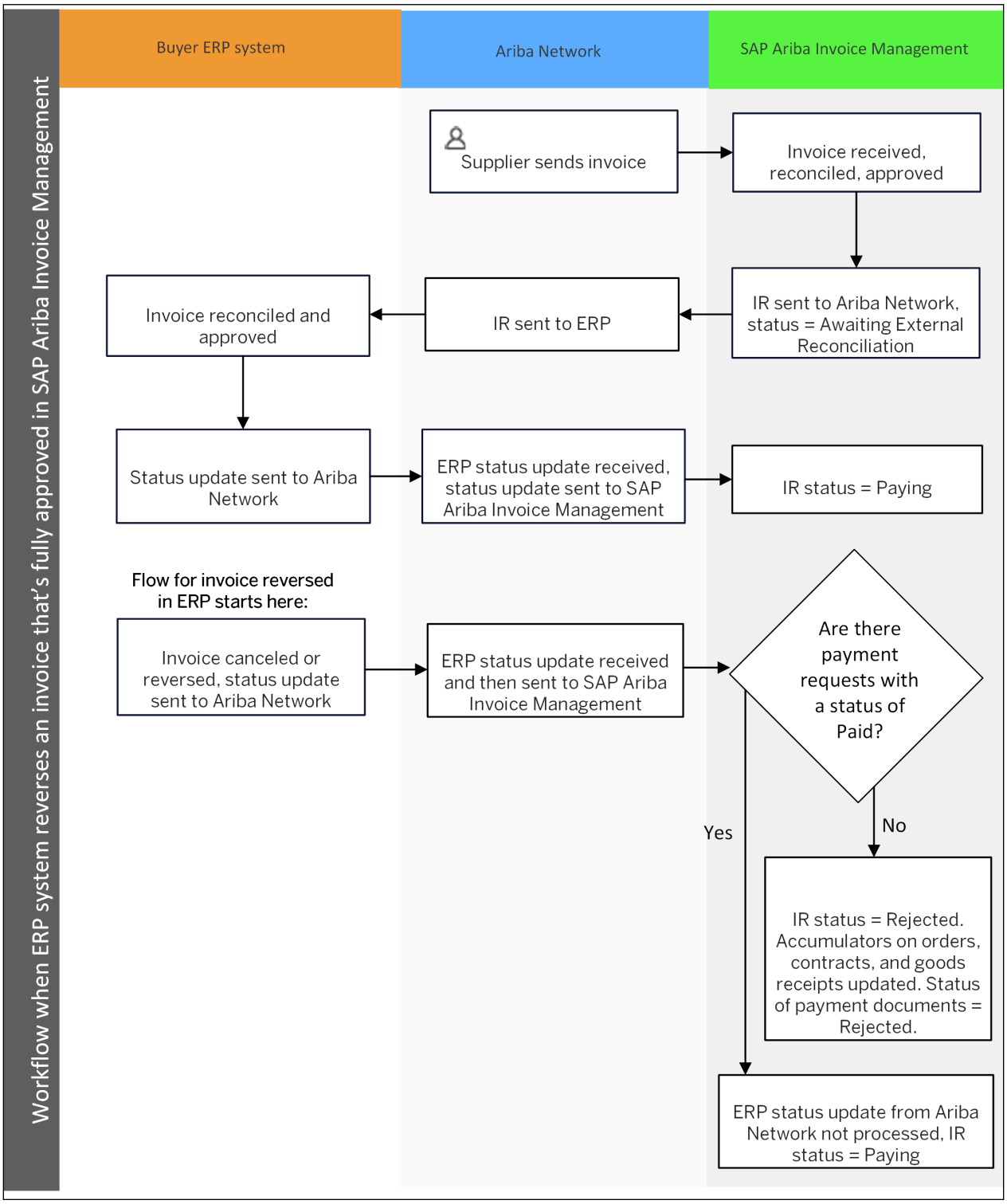
- If the ERP system cancels an invoice that is already paid, the IR document status changes from **Paid** status to **Paying** status.

When the IR is rejected or a payment is cancelled, accumulators on the purchase orders, contracts, and goods receipts that were updated by the invoice are decremented for the amount on the rejected invoice.

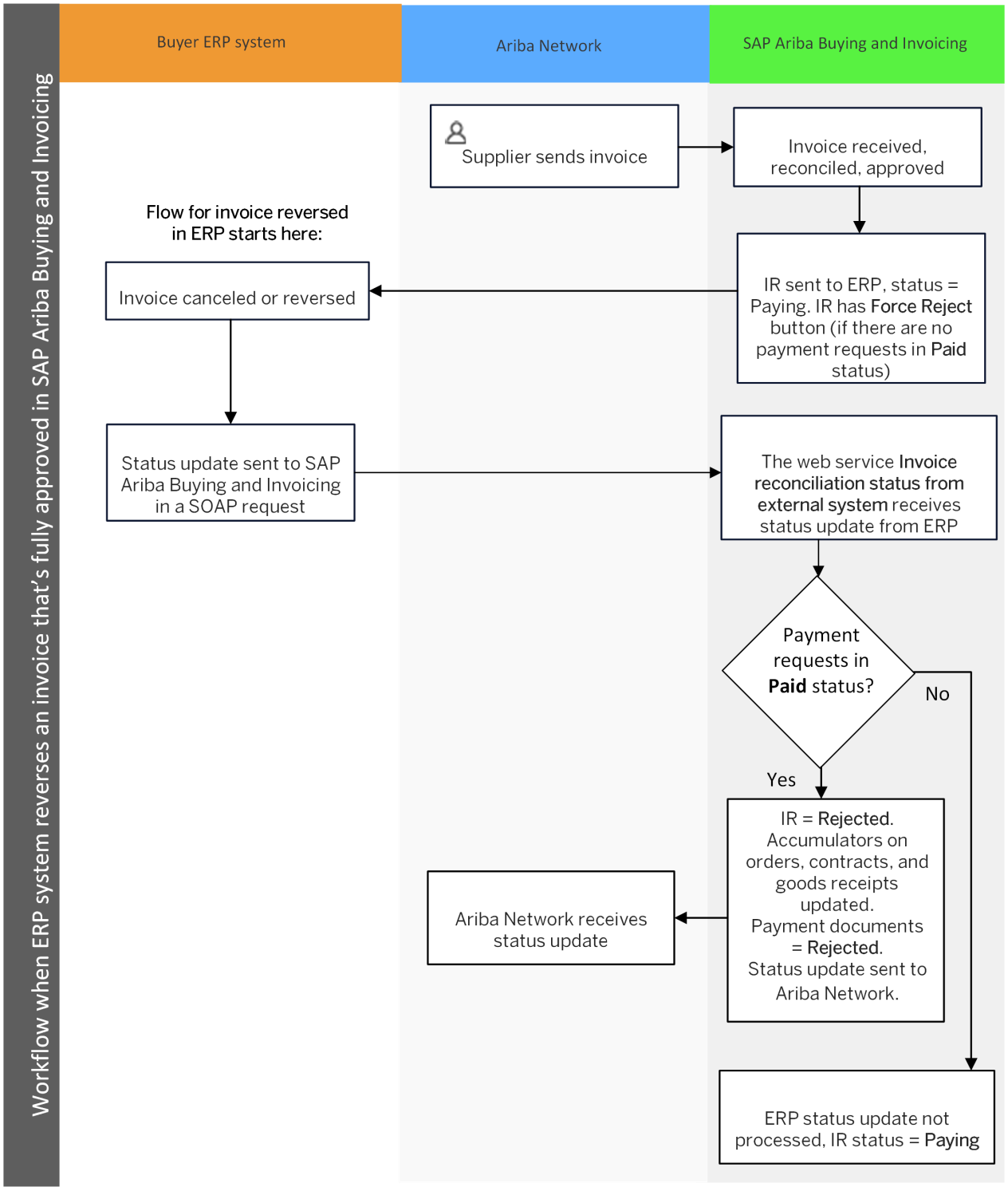
Restrictions

- Fully or partially paid invoices can't be rejected.
- You can reject an invoice against a closed PO, but the closed PO isn't automatically reopened when you do so. (On the other hand, if you reject an invoice against a closed contract, the contract **is** reopened.)

Workflow for Automatic Rejection in SAP Ariba Invoice Management

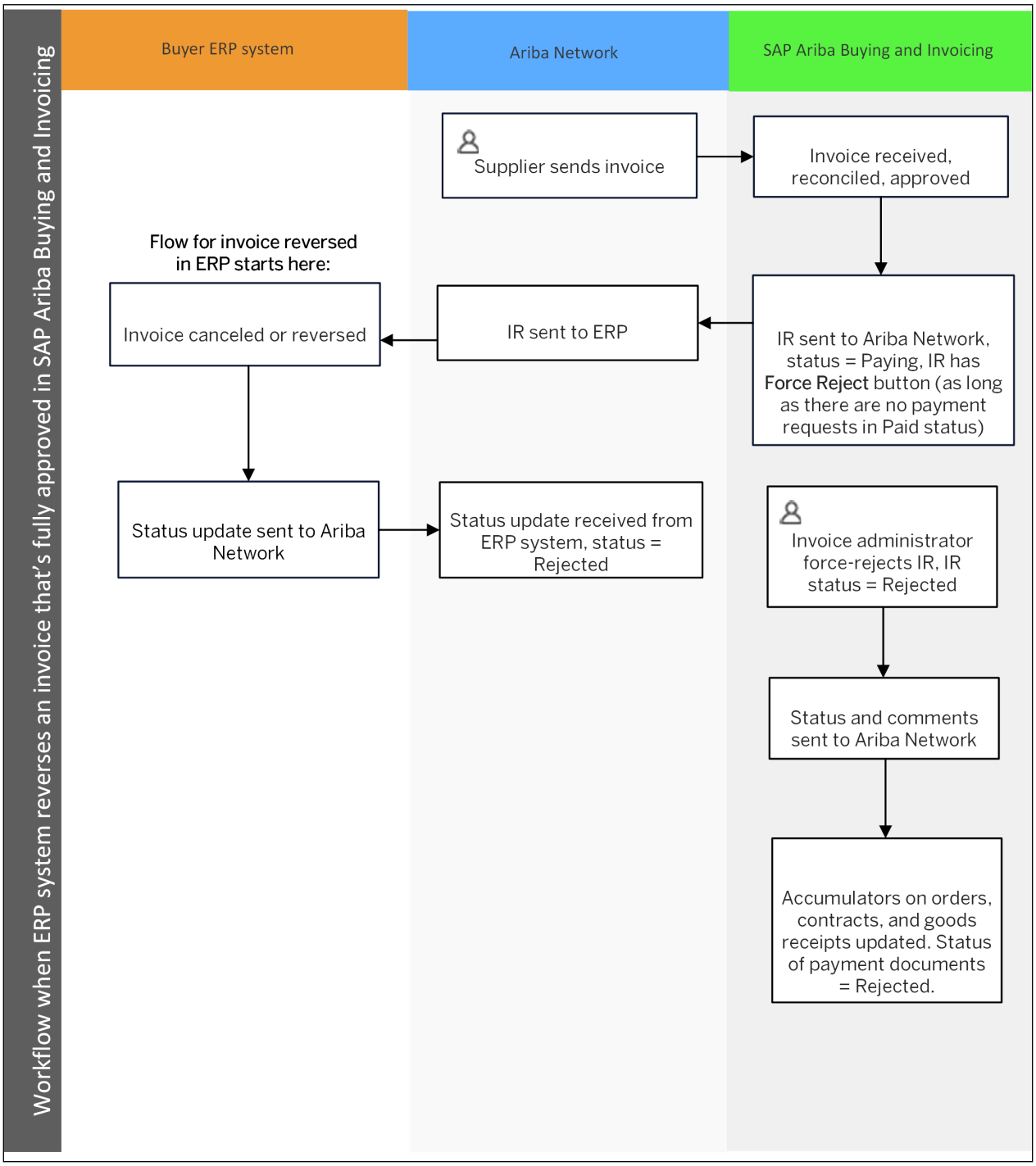


Workflow for Automatic Rejection in SAP Ariba Buying and Invoicing



> **Note**
>
> Buyers who choose not to invoke the web service when an invoice is canceled in the ERP system can use the **Force Reject** button to manually reject the invoice in SAP Ariba Buying and Invoicing.

Manual Rejection in SAP Ariba Buying and Invoicing



> **Note**
>
> If you force-reject an IR that was not already reversed in the ERP system, you can manually reverse the invoice in the ERP system either before or after you force-reject the IR in the SAP Ariba solution.

Configuring a Web Service for Auto-Rejecting Approved Invoices Canceled in the ERP System

Use this procedure to configure a web service for auto-rejecting approved invoices canceled in the ERP system.

Prerequisites

You must be a member of the **Customer Administrator** or **Integration Admin** group to configure integration events.

Context

For SAP Ariba Buying and Invoicing to receive invoice cancellations from the ERP system and change the invoice reconciliation status to **Rejected**, you configure an inbound end point for the SOAP request sent by the ERP system, and then enable a web service. The web service **Invoice reconciliation status from external system** is an inbound task that provides status updates from an external ERP system. If a fully approved invoice is canceled in the ERP system, the web service receives the status updates in the form of SOAP requests from the ERP system, and then changes the status of the invoice reconciliation to **Rejected** in SAP Ariba Buying and Invoicing.

This feature is supported in SAP Integration Suite, managed gateway for spend management and SAP Business Network 2003 (Q1 2020) release and later.

Procedure

1. From the dashboard, choose **Manage** > **Core Administration**.
2. Choose **Integration Manager** > **End Point Configuration**.
3. Click **Create New**.
4. Enter a name for the end point.
5. Choose the type of end point (**Inbound**).
6. Perform one of the following actions:
   - If you are using WS-Security, paste the contents of the certificate file from the EAI system in the **Certificate File** field under **Web Services Security**. Provide the SAP Ariba certificate to the ERP administrator.
   - To use HTTP Basic Authentication, navigate to the **HTTP Authentication** area. Enter the user ID in the **Sign In** field and the password in the **Password** fields. Provide this information to the ERP administrator.
7. Click **Save**.
8. Choose **Integration Manager** > **Integration Configuration**.

9. Find the web services task called **Invoice reconciliation status from external system**, and choose **Actions** > **Edit**.

   The **Edit data import/export task** page opens.
10. In the **Status** field, select **Enabled**.
11. In the **End point** field, select the end point from the dropdown.
12. Click the **View WSDL** link to generate the WSDL file for the service.
13. Click **Save**.
14. Transfer the WSDL file to the ERP system and build the appropriate SOAP requests and process responses.

> **Note**
>
> In the SAP Ariba test environment, the XML namespace (`xmlns`) in the WSDL differs from the XML namespace used when connecting to the SAP Ariba production environment. When migrating from the test environment to the production environment, generate a new WSDL from the production environment.

Related Information

Configuring an ERP SOAP Request to Send Invoice Status Updates to SAP Ariba Buying and Invoicing [page 283]
Cancellation of Approved Invoices in the ERP System [page 278]

Configuring an ERP SOAP Request to Send Invoice Status Updates to SAP Ariba Buying and Invoicing

In SAP Ariba Buying and Invoicing, the web service **Invoice reconciliation status from external system** receives a SOAP request from the ERP system whenever a reconciled invoice is canceled or reversed in the ERP system.

The SOAP request from the ERP system (`InvoiceReconciliationStatusPullRequest`) must include information to identify the IR document in SAP Ariba Buying and Invoicing. This identifier can either be the following:

`UniqueName`: The unique name, or ID, of the IR document in SAP Ariba Buying and Invoicing, for example, `<UniqueName>IRINV_ERP-19299-56</UniqueName>`

or it can be the combination of the following elements:

- `InvoiceNumber`: The supplier invoice number in SAP Ariba Buying and Invoicing
- `InvoiceDate`: The date on the invoice (the date the invoice was created)

> **Note**
>
> `InvoiceDate` is required only if your site allows the reuse of invoice numbers within a calendar year, determined by the site configuration parameter `Application.Invoicing.AllowReusingInvoiceNumberAcrossCalendarYear`.

Valid date formats:

- yyyy-MM-dd'T'hh:mm:ssZ (for example, 2018-03-29T13:23:29-8:00)
- yyyy-MM-dd
- yyyyMMdd

- `SupplierLocation.UniqueName`: The unique name of the supplier location

The ERP system sends the request to the following URL: `https://<hostname>/Buyer/soap/<SAP Ariba site ID>/InvoiceReconciliationStatusPull`

Following is an example of the request from the ERP system:

Sample Code

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:urn="urn:Ariba:Buyer:">
 <soapenv:Header>
      <urn:Headers>
      <!--Variant and partition values are not used, do not specify values for
them-->
         <urn:variant></urn:variant>
         <urn:partition></urn:partition>
      </urn:Headers>
   </soapenv:Header>
   <soapenv:Body>
      <urn:InvoiceReconciliationStatusPullRequest partition="" variant="">
         <urn:IRStatusChangeInput_Item>
            <urn:item>
               <!--You may enter the following 7 items in any order-->
               <urn:CommentsToSupplier>Rejected due to wrong cost center</
urn:CommentsToSupplier>
               <urn:InvoiceDate>2018-03-29T13:23:29-8:00</urn:InvoiceDate>
               <urn:InvoiceNumber>IR1234</urn:InvoiceNumber>
               <urn:InvoiceReconciliationId></urn:InvoiceReconciliationId>
               <urn:RejectionReasonCodes>
                  <!--0 or more repetitions:-->
                  <urn:item></urn:item>
               </urn:RejectionReasonCodes>
               <urn:RequestedAction>reject</urn:RequestedAction>
               <urn:SupplierLocation.UniqueName>11</
urn:SupplierLocation.UniqueName>
            </urn:item>
         </urn:IRStatusChangeInput_Item>
      </urn:InvoiceReconciliationStatusPullRequest>
   </soapenv:Body>
</soapenv:Envelope>
```

Related Information

Configuring a Web Service for Auto-Rejecting Approved Invoices Canceled in the ERP System [page 282]
Cancellation of Approved Invoices in the ERP System [page 278]

Resending Invoice Documents to SAP Business Network

When an invoice or invoice reconciliation (IR) document fails to reach SAP Business Network because of a network error (ANERR), and the system has retried the transmission the maximum number of times, you can manually resend the document.

Prerequisites

Members of the **Invoice Administrator** or **Customer Administrator** group can perform this task.

Context

Typically, when an invoice or IR document fails to reach SAP Business Network, the invoice administrator is notified, makes any corrections needed (for example, updating supplier data), and waits for the SAP Ariba solution to retry the transmission.

Sometimes, however, the situation is still unresolved when the maximum number of automatic retries is reached. In such cases, the invoice administrator can manually resend the document.

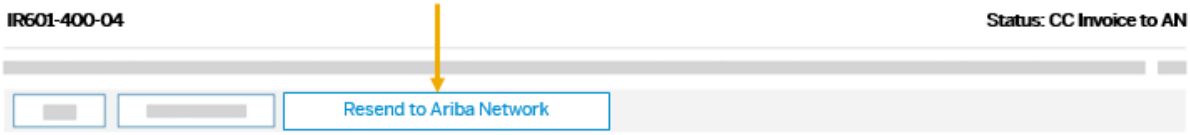
Following are examples of situations in which this procedure is useful:

- Buyer-entered invoices with a status of **CC Invoice to AN**
- IR documents that failed to be sent to SAP Business Network.
- IR documents where a status update request failed to be sent to SAP Business Network

When you resend an invoice or IR document, it's added to the queue for sending to SAP Business Network. Queue processing occurs at regular intervals based on an internal scheduled task.

Procedure

1. Open the invoice or IR document to resend.
2. Choose **Resend to Ariba Network**.



The **Resend to Ariba Network** button is shown only in situations where a network error prevented the invoice or IR document from being sent.

A comment popup opens, along with a message saying the document will be requeued.

3. Enter a comment, if desired, and choose **OK**.

Results

The document is added to the queue for sending to the network.

A record of the action is added to the document's history.

Important Disclaimers and Legal Information

Hyperlinks

Some links are classified by an icon and/or a mouseover text. These links provide additional information.
About the icons:

- Links with the icon : You are entering a Web site that is not hosted by SAP. By using such links, you agree (unless expressly stated otherwise in your agreements with SAP) to this:
  - The content of the linked-to site is not SAP documentation. You may not infer any product claims against SAP based on this information.
  - SAP does not agree or disagree with the content on the linked-to site, nor does SAP warrant the availability and correctness. SAP shall not be liable for any damages caused by the use of such content unless damages have been caused by SAP's gross negligence or willful misconduct.
- Links with the icon : You are leaving the documentation for that particular SAP product or service and are entering an SAP-hosted Web site. By using such links, you agree that (unless expressly stated otherwise in your agreements with SAP) you may not infer any product claims against SAP based on this information.

Videos Hosted on External Platforms

Some videos may point to third-party video hosting platforms. SAP cannot guarantee the future availability of videos stored on these platforms. Furthermore, any advertisements or other content hosted on these platforms (for example, suggested videos or by navigating to other videos hosted on the same site), are not within the control or responsibility of SAP.

Beta and Other Experimental Features

Experimental features are not part of the officially delivered scope that SAP guarantees for future releases. This means that experimental features may be changed by SAP at any time for any reason without notice. Experimental features are not for productive use. You may not demonstrate, test, examine, evaluate or otherwise use the experimental features in a live operating environment or with data that has not been sufficiently backed up.
The purpose of experimental features is to get feedback early on, allowing customers and partners to influence the future product accordingly. By providing your feedback (e.g. in the SAP Community), you accept that intellectual property rights of the contributions or derivative works shall remain the exclusive property of SAP.

Example Code

Any software coding and/or code snippets are examples. They are not for productive use. The example code is only intended to better explain and visualize the syntax and phrasing rules. SAP does not warrant the correctness and completeness of the example code. SAP shall not be liable for errors or damages caused by the use of example code unless damages have been caused by SAP's gross negligence or willful misconduct.

Bias-Free Language

SAP supports a culture of diversity and inclusion. Whenever possible, we use unbiased language in our documentation to refer to people of all cultures, ethnicities, genders, and abilities.

www.ariba.com

Copyright © 2026 Ariba, Inc. All rights reserved.

This documentation, as well as the Ariba solutions, software and/or services described in it, contain proprietary information. They are provided under a license or other agreement containing restrictions on use and disclosure and are also protected by copyright, patent and/or other intellectual property laws. Except as permitted by such agreement, no part of the document may be reproduced or transmitted in any form by any means, electronic, mechanical or otherwise, without the prior written permission of Ariba, Inc.

Ariba, Inc. assumes no responsibility or liability for any errors or inaccuracies that may appear in the documentation. The information contained in the documentation is subject to change without notice.

Ariba and Ariba products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of Ariba, Inc. in the United States and other countries. Please see http://www.ariba.com/legal/trademarks for additional trademark information and notices.

Ariba Sourcing solutions (On Demand and software) are protected by one or more of the following patents, including without limitation: U.S. Patent Nos. 6,199,050; 6,216,114; 6,223,167; 6,230,146; 6,230,147; 6,285,989; 6,408,283; 6,499,018; 6,564,192; 6,871,191; 6,952,682; 7,010,511; 7,072,061; 7,130,815; 7,146,331; 7,152,043;7,225,152; 7,277,878; 7,249,085; 7,283,979; 7,283,980; 7,296,001; 7,346,574; 7,383,206; 7,395,238; 7,401,035; 7,407,035; 7,444,299; 7,483,852; 7,499,876; 7,536,362; 7,558,746; 7,558,752; 7,571,137; 7,599,878; 7,634,439; 7,657,461; 7,693,747; 8,364,577; and 8,392,317. Patents pending.

Other Ariba product solutions are protected by one or more of the following patents:

U.S. Patent Nos. 6,199,050, 6,216,114, 6,223,167, 6,230,146, 6,230,147, 6,285,989, 6,408,283, 6,499,018, 6,564,192, 6,584,451, 6,606,603, 6,714,939, 6,871,191, 6,952,682, 7,010,511, 7,047,318, 7,072,061, 7,084,998; 7,117,165; 7,225,145; 7,324,936; 7,536,362; 8,364,577; and 8,392,317. Patents pending.

Certain Ariba products may include third party software or other intellectual property licensed from a third party. For information regarding software or other intellectual property licensed from a third party, go to http://www.ariba.com/copyrights.cfm.