Oracle® Fusion Middleware

WebCenter Forms Recognition AP Project 2803 Installation and Configuration Guide

12c (12.2.1.4.200714)

F33281-02

September 2020

Describes the AP Project installation and configuration concepts and procedures

ORACLE®

Oracle Fusion Middleware Oracle WebCenter Forms Recognition AP Project 2803 Guide, 12c (12.2.1.4.200714)

F33281-02

Contents

1 About AP Packaged Project for Oracle WebCenter Forms Recognition

The AP Packaged Project automates data entry for invoices, purchase orders, non-purchase orders, and credit notes.

Information in fields such as invoice number, purchase order, dates, vendors, and so on, is automatically captured and stored as an invoice document. The document is then available for use with your business's information system and can be shared among different departments and archived. AP Packaged Project also includes processes for tax determination and validation, automatic general ledger account coding, and solutions reporting.

AP Packaged Project can be integrated with almost any environment and includes the following components:

- Reporting feature for auditing and reporting purposes
- Thick Verifier for document quality-assurance purposes

1.1 Supported Accounts Payable Documents

AP Packaged Project supports the following document types:

- Vendor invoices
- Vendor credit memos
- Third party freight invoices

Additional document types, such as statements or travel and expense forms, need to be configured within the solution as new document classes.

2 About the Installation and Setup Process

The following steps are a high-level overview of the procedures that you need to perform to install and configure AP Packaged Project:

- Create the AP Packaged Project folder structure.
- Install AP Packaged Project.
- Create the pool directory.
- Run the database creation script.
- Configure the INI file and database, CSV file, and registry settings.
- Configure data export.
- Configure reporting and custom base classes.
- Set up and add clients.
- Create vendor master partitions, or create identification fields without using a partition.
- Set up business rules relating to predefined data fields and document types.
- Configure and activate the Dynamic Verifier Form.
- Set up users.
- Set up review states.
- [Optional] Create a connection to Reporting database.

2.1 Prepare for Installation

Before you install AP Packaged Project, verify that you have completed the following steps:

1. You have installed WebCenter Forms Recognition 12.2.1.4.0 or later.
2. You have obtained the AP Packaged Project installation files. The *AP 2803* is located inside <*WFR1221420714.zip*>*Projects*.

2.1.1 Configure the Local Settings for Decimal Separators

AP Packaged Project functions on any machine with a language setting that uses the western alphabet. Though English or English (US) is recommended for the server configuration. It also runs independent of whether the machine and system localization uses a period or a comma as the decimal separator, even if the server is set to one option, even if one or more Verifier stations use different separators.

Amount fields are outputted using a period as the decimal separator in all instances. If database output is required, you need to configure the language and decimal separator preferences against that database accordingly.

For locations that use a space as the thousand separator, such as the French location, this must be changed to a comma or period, whichever is appropriate.

Dates are handled internally in a manner that is entirely independent of the system locale. The Verifier display and output formats are configurable.

2.2 Create the AP Packaged Project Folder Structure

The first step in installing AP Packaged Project is to create the underlying folder structure within Windows Explorer. To create the folder structure, complete the following steps:

1. Create a folder directory on the hard drive, for example, `C:\AP Projects\AP Project 2803`.
2. Within this directory, create these folders if they do not exist:

- Import
- Batch
- Global
- Export

2.2.1 Create the Pool Directory

The pool directory stores the internal representation of the vendor master data, which is accessed by the system at runtime when determining the document vendor. To create the pool directory, in the Global directory, create a new directory and name it Pool.

2.3 Install the AP Packaged Project Project File

To install the AP Packaged Project project file, complete the following steps:

1. Navigate to the Global directory.
2. Copy the following files from the installation directory to the Global directory:
   - The SDP file
   - All contents of the Train directory
   - The INI file
3. Rename the SDP and INI files as appropriate for your business needs. The SDP and INI files must have the same name, but ensure that you retain the SDP and INI file extensions.

Note: In the AP Packaged Project help, the INI file is referred to as `<project>.ini`.

2.4 Run the Database Table Creation Script

To create the database and AP Package Project tables, complete the following steps. We recommend that the tables exist in their own database/tablespace.

1. On your Microsoft SQL or Oracle Server, create a new database.
   a. For Microsoft SQL Server:
   Please follow this link: *https://docs.microsoft.com/en-us/sql/relational-databases/databases/create-a-database?view=sql-server-ver15*.
   b. For Oracle Database:
   Execute the following commands in the SQL command prompt with a database administrator user:

```
SQL> create user <username> identified by <password>;

SQL> grant CREATE SESSION, ALTER SESSION, CREATE DATABASE LINK,
CREATE MATERIALIZED VIEW, CREATE PROCEDURE, CREATE PUBLIC SYNONYM,
CREATE ROLE, CREATE SEQUENCE, CREATE SYNONYM, CREATE TABLE, CREATE
TRIGGER, CREATE TYPE, CREATE VIEW, UNLIMITED TABLESPACE to
<username>;
```

2. Run the `MasterSQL - <Oracle/SqlServer>.sql` script against the new database. This script is included in the <*WFR12214200714.zip*>*Projects**AP 2803**DB Scripts*\.
   a. For Microsoft SQL Server
      i. Open Microsoft SQL Server Management Studio.
      ii. Click **File > Open** and navigate to `MasterSQL - SqlServer.sql`. Click **OK** to open the file.
      iii. On the toolbar, select the appropriate database from the database drop-down list.

iv. Click **Execute** on the toolbar to run the script.
v. A message appears at the bottom which indicates the execution of the script completed successfully.

b. For Oracle Database:
i. Create a new connection in SQL Developer for the database created in previous step.
ii. Connecting to the database would open an empty SQL Worksheet.
iii. Click **File > Open** and navigate to `MasterSQL - Oracle.sql`. Click **Open** to open the file in the SQL Developer.
iv. Click on *Run Script (F5)* icon on the SQL Worksheet toolbar or press the [F5] key on the keyboard to execute the script. A dialox box opens which prompts for the database against which the script has be executed. The dialog box now associates `MasterSQL - Oracle.sql` with the database.
v. A message appears at the bottom which indicates the execution of the script completed successfully.

Note: Change the Encoding to *UTF8* in the Oracle SQL Developer Preferences before running the Oracle version of the database script.

2.5 Configure the <project>.ini File

To configure the `<project>.ini` file, complete the following steps. You complete the AP Packaged Project installation by performing a basic configuration of the file.

1. Navigate to the Global directory you created.
2. Open the `<project>.ini` file and configure the `GRL_VL_ProjectName` settings**.**

(Optional) GRL_VL_ProjectName

The value is set to *AP Packaged Project* by default. This configured value will be stored in the reporting database for display in a reporting application. If there are multiple AP Projects configured for reporting, then update this value as required.

(Optional) GRL_VL_Version

We recommend setting its value to 1, but it can be left blank or changed to assign document records to different testing cycles. This is the version number of the project, which is recorded in the reporting database.

(Optional) GRL_VL_ClientName

Set this to the default client name that is recorded in the reporting database for each processed document. This is superseded by a setting in the BRWClient table and is therefore optional.

GRL_OP_ReadSettingsFromDB

Set this to Yes. This controls whether the project reads configuration settings from the database created in the previous step.

GRL_VL_SQLConnectionGroup

This must be set to 01. It is the reference to the SQL connection group in the SQL section of the INI file that contains the connection string to the AP Packaged Project database.

GRL_OP_BatchInDatabase

This must be set to Yes. This specifies whether the AP Package Project document batches are to be held in the AP Packaged Project database or held as a batch root in the file system.

GRL_VL_BatchSQLConnection Group

This must be set to 02. This is the reference to the SQL connection group in the SQL section of the INI file that contains the connection string to the main AP Packaged Project database.

SQL_VL_01_ConnectionString

Set this to the connection string for the AP Packaged Project database. For example:

- Microsoft SQL Server

```
SQL_VL_01_ConnectionString=Provider=SQLOLEDB.1;Password=<User
Password>;Persist Security Info=TRUE;User ID=<User ID>;Initial
Catalog=ICT;Data Source=<DataSource>
```

- Oracle Database

```
SQL_VL_01_ConnectionString=Provider=OraOLEDB.Oracle;Password=<User
Password>;User ID=<User ID>;Data Source=<DataSource[:Port]/ServiceName>
(for Oracle)
```

SQL_VL_02_ConnectionString

Set this to the connection string for the core engine database. For example:

- Microsoft SQL Server

```
SQL_VL_02_ConnectionString=Provider=SQLOLEDB.1;Password=<User
Password>;Persist Security Info=TRUE;User ID=<User ID>;Initial Catalog=
<WFR_Database>;Data Source=<DataSource>
```

- Oracle Database

```
SQL_VL_02_ConnectionString=Provider=OraOLEDB.Oracle;Password=<User
Password>;User ID=<User Id>;Data Source=<DataSource[:Port]/ServiceName>
(for Oracle)
```

3. Save the changes and close the file.

2.6 Activate Asian Language Recognition

If your system processes documents using a supported Asian language (Chinese, Japanese, Korean or Thai), complete the following steps to configure the recognition.

Prerequisite
It is recommended that you process projects that contain documents with Chinese, Japanese, Korean or Thai (CJKT) characters using a separate instance of the `<project>.sdp` file. The Abbyy Finereader 10 engine has been noted to exhibit slower performance when CJKT languages are added to the system and a lower process rate for documents that do not contain CJKT characters.

1. Using WebCenter Forms Recognition Designer, open the `<project>.sdp` file.
2. Navigate to **Project** settings.
3. On the **Definition Mode** tab, select the **Use multi-byte encoding** check box.
4. Save the `<project>.sdp` file.
5. In WebCenter Forms Recognition Designer definition mode, at the project level node, open the OCR Settings.
6. To activate Japanese or Chinese, on the **Recognition** tab, select **Japanese + English**, and add it to the **Used** list.
7. To activate Korean, on the **Recognition** tab, in the **Installed** list, select **Korean** and add it to the **Used** list.

8. To activate Thai, on the **Recognition** tab, in the **Installed** list, select **Thai** and add it to the **Used** list.
9. Open Windows registry editor.
10. Using the Registry Editor, add a new key called **CJKT Support** under the existing **HKEY_LOCAL_MACHINE\SOFTWARE*[Wow6432Node\]*Oracle** key.
11. Create a new **DWORD (32-bit)** value within the **CJKT Support** key, called **CJKT_MinimalSymbolSequenceLengthForWordsSplit.**
12. Set the value data to **1** as in the example given below:

```
HKEY_LOCAL_MACHINE\SOFTWARE \Wow6432Node\Oracle\CJKT Support
CJKT_MinimalSymbolSequenceLengthForWordsSplit       REG_DWORD      0x00000001 (1)
```

13. Close the Windows registry editor.

2.7 Registry and Verifier Message Settings

The following sections contain the recommended registry and thin client configuration settings.

2.7.1 Configure Registry Settings

To configure the registry settings, complete the following steps:

1. Open the Windows registry editor.
2. Using the Registry Editor, add a new key called **CJKT Support** under the existing **HKEY_LOCAL_MACHINE\SOFTWARE*[Wow6432Node\]*Oracle** key.
3. Create a new key and name it **Cedar**.
4. In the new Cedar folder, create two DWORD values and name them `AnalyzeLinesOptionally` and `ASEnginePoolAllowedCharDifference`.
5. Assign the new registry key `AnalyzeLinesOptionally` a value of 1 using a hexadecimal base.
6. Assign the new registry key `ASEnginePoolAllowedCharDifference` a value of 0 using a hexadecimal base.
7. Close the Windows registry editor.

2.7.2 Configure Verifier Message Settings

If you deploy the thin client Verifier, then you need to update the web.config file so that the dialog box information and optional messages are activated within the Verifier interface. This is not required for the thick client Verifier application. To configure the Verifier message display setting, complete the following steps:

1. Navigate to *[C:\Program Files (x86)]\Oracle\WebCenter Forms Recognition\WebCenter Forms Recognition Web Server* and open the `web.config` file.
2. In the `web.config` file, set the `mouseClicked` enabled and `tabPressed` enabled properties to `true`.
3. Save and close the file.

3 Solution Features

3.1.1 Duplicate Invoice Number Checking

The AP Packaged Project provides the ability to identify a duplicate invoice, by performing a lookup against a configured data source; typically the customer's ERP database. This check is performed as part of the document validation procedure, which means it will be performed both on the RTS instance after data extraction, and also within the Verifier application (and Web Verifier). Therefore, both extracted Invoice Number values, and manually entered Invoice Number values will be subject to this check for duplicates.

Different organizations may have varying definitions of what constitutes a 'duplicate' invoice. Typically, a duplicate invoice is defines as one coming from the same vendor (including the vendor site) with the same invoice number and issued to the same company. The AP Project provides configuration options to define which of these values should be used to define a duplicate invoice for a particular implementation.

Where a duplicate invoice is identified, the document will be prevented from being exported, and will require manual verification so the appropriate action can be taken. The system will not allow an invoice with a duplicate invoice number to be released and exported unless the configuration allows the Verifier user to forcefully validate and release the document with the duplicate invoice number in place.

This feature can be enabled and configured through the *WFR section* of the project configuration file.

3.1.2 Credit Note Export: Negative Quantity and Totals

Some ERP systems, such as Oracle E-Business Suite, require that when a credit note is inserted into the system through its automatic entry interface the header-level total and the quantity and total at line-level are expressed as negative values.

The AP Packaged Project exports numeric values as positive numbers by default, but this feature allows the project to be configured to always export the quantity and total amounts as their negative values when the document is a credit note. If enabled, this will occur for all credit notes, irrespective of whether the original document expressed those values as positive or negative values.

This feature can be enabled through the *WFR section* of the project configuration.

3.1.3 Metadata Pass-through

The AP Packaged Project provides the ability to parse the image filename, and write component values to the CSV and/or XML files that are created during export. This is in addition to the basic import functionality described in the IMP section of the project configuration.

Unlike the basic import function, where image filename component values are assigned to fields within the Invoices class, this Metadata Pass-through feature does not require fields to be created to hold the component values.

Both the basic import function and this Metadata Pass-through feature can be enabled within the same project, and work independently of each other.

This feature requires that the image filename use the underscore character as a separator to delimit the components. For example:

Assume the image filename: **007_00000011_204_05-26-2011_EMEA.tif**

The image filename in this example has five components:

- 007
- 00000011
- 204
- 05-26-2011
- EMEA

You can configure up to 99 components through the project configuration file, and for each configured component, you can specify whether it should be written to the CSV file, the XML file or both files at export. An example configuration could be:

```
CXP_VL_01_FilenameComponent=1
CXP_VL_01_CSVFieldID=<%ZXX>
CXP_VL_01_XMLHeaderFieldName=imageComponent1

CXP_VL_02_FilenameComponent=2
CXP_VL_02_CSVFieldID=<%ZYY>
CXP_VL_02_XMLHeaderFieldName=

CXP_VL_03_FilenameComponent=3
CXP_VL_03_CSVFieldID=
CXP_VL_03_XMLHeaderFieldName=imageComponent3
```

In the example configuration given above, component 1 will be written to both the CSV file and the XML file, component 2 will be written only to the CSV file, and component 3 will be written to only the XML file.

This feature can be enabled and configured through the *WFR section* of the project configuration.

3.1.4 PO Number Removal for Non-PO Invoices

By default with the AP Packaged Project, if a document is manually processed in the Verifier application, and the user changes the Invoice Type from **PO** to **NO-PO**, any existing value in the PO Number field will remain intact.

For some ERP integrations, this situation, where a non-PO invoice has a PO number value, can cause the invoice to be created in the ERP system erroneously as a PO-based invoice.

If enabled, the PO Number Removal feature will ensure that during document validation any PO Number value is removed if the Invoice Type is set as **NO-PO**.

This feature can be enabled through the *WFR section* of the project configuration file.

3.1.5 Separator Page Detection for Supporting Documents

Invoices are commonly received with supporting documentation attached. While customers typically want to retain these as a single document in the downstream content repository after Forms Recognition has processed them, the supporting pages should not be considered during extraction.

While WebCenter Forms Recognition does allow for the first *n* pages and /or the last *n* pages of a document to be OCR'd, this functionality cannot be effectively used in this situation, where invoices are a variable number of pages.

Instead, all pages of the document should be OCR'd as normal, then this Separator Page Detection feature allows the project administrator to configure one or more phrases, which, if found on the document, should be considered as a separator. In this case, all OCR text following this separator will be discarded from the workdoc, and will not be considered during the

extraction step of the Forms Recognition workflow. This feature can also be configured to remove the page containing the separator phrase(s) from the exported image if required.

Note: If separator page removal is enabled, additional settings must be configured in the *BRW*EXP section and the CSV section of the project configuration to ensure that the modified image file is exported from Forms Recognition instead of the original image that was imported. These settings are described in *WFR section*.

This feature can be enabled and configured through the *WFR section* of the project configuration.

3.1.6 Export Custom Unit of Measure Value to XML

There may be situations where a customer may wish to export a line item *Unit of Measure* value that differs from that on the purchase order line. For example, the unit of measure value may be expressed on the invoice, and extracted by the AP Packaged Project, as **EACH**, but the output requirement may be that the exported value is written as **Each** in the XML output file.

In some scenarios the required output may be entirely different, for example an extracted value of **EACH** may be required to be written to the XML as **EA**.

This feature allows a project administrator to specify whether one or more units of measure should be exported as a different value to the one that was extracted, and to configure the value that should be written to the XML in each case.

This feature can be enabled and configured through the *WFR section* of the project configuration.

3.1.7 Force Validation of Documents Using Custom Invalid Reasons

Standard functionality of the Verifier application provides the user with a couple of options for handling invalid documents, or documents that cannot be successfully validated and released for export:

1. The user can set the document to an exception state and (optionally) move it to a separate batch for processing later.
2. The user can manually reclassify the document to the *Void* class.

While these options are sufficient in the majority of cases, there may be a business requirement to allow an invalid document to pass verification and be released for export. This is particularly true where the downstream process will be able to identify this type of document and process it accordingly. Again, standard functionality within Verifier will often allow users to set an appropriate *Invalid Reason* to allow documents that could otherwise not be validated to continue through to export.

However, the *Invalid Reason* approach is limited because each invalid reason is linked to a rule, and none of the available rules make all fields in a document valid, so there may still be some fields that require user entry or correction. In the case of a bad image or a document that is not an invoice, such manual verification may not be appropriate, or even possible.

This feature allows an administrator to configure the project so that the Verifier user can forcibly validate a document, and allow it to be released to export, by selecting one of a defined list of invalid reasons.

This feature can be enabled and configured through the *WFR section* of the project configuration.

3.1.8 Format Line Items in XML for the E-Business Suite Open Interface

In some cases where E-Business Suite is the downstream ERP system, the XML output from the AP Packaged Project may not meet the specific validation criteria of the EBS open interface. This may result in the import failing or invoice lines being imported with EBS holds applied unnecessarily.

For example, in a scenario where the **PURCHASE_BASIS** EBS field is configured to determine a service PO, and the value of that field is **SERVICES**, that value is exported to the XML file as the line type, e.g.:

```
<lineType>SERVICES</lineType>
```

However, **SERVICES** is not a valid line type value for the EBS open interface, and this value should actually be exported in the XML file as **ITEM**.

Additionally, for certain line types, such as **FREIGHT** and **MISCELLANEOUS**, the EBS open interface may expect the quantity, unit price and unit of measure to be zero or empty. Finally, for invoices where the PO type is **SERVICE** the EBS open interface may require the quantity and unit price values to be transposed in the XML file.

This feature allows the project administrator to configure how the line item data is written to the XML file in all of the examples described above, to ensure that the resulting file content will not cause any unnecessary rejections or failures during import into E-Business Suite through the open interface.

This feature can be enabled and configured through the *WFR section* of the project configuration.

4 Configure Data Export

You can configure the following export options for AP Packaged Project:

- Export TIFF images.
- Export a PDF file.
- Configure export to database tables.
- Configure the XML file.
- Configure the CSV file.
- Configure output to ProcessIT for Oracle EBS.
- Configure output of the OCR XML file.
- Set up a custom export.

Data export is controlled by settings in the following tables. Each of the tables is keyed by an export profile ID that can be assigned to a client.

- BRWEXP
- BRWEXPHeader
- BRWEXPLINES
- BRWEXPGL
- BRWEXPTax

4.1 Export TIFF Images

Runtime Server settings can be used to export a copy of the original TIFF image to the export directory. It is recommended that this setting must always be used over custom settings. However, custom settings provide additional options for outputting a second TIFF image and must be used when one or more of the following applies:

- A second TIFF file is required during document export.
- The TIFF file name must be set to the document URN rather than the original image file name.
- The TIFF image resolution needs to be changed from the original image resolution.
- The TIFF image compression ratio needs to be changed from that of the original image.

4.2 Export an Additional TIFF Image

To export an additional TIFF image, complete the following steps:

1. Open the **BRWEXP** table.
2. Set the **OutputTiffFile** column to `TRUE`.
3. To set the name of the TIFF file to match the document URN in the <project>.ini file, set the **TiffName** column to `URN`. Otherwise, the name used matches the original imported document.

Result The additional TIFF file is always written to the directory specified as the export directory on the runtime server instance settings for the instance that is carrying out the document export. If you do not specify an export directory, the default export directory in the `DefaultExportPath` setting is used. If the setting is left blank, the document export fails and the batch goes to a status of 750.

4.2.1 Configure DPI and the Compression Type for TIFF Images

Configuring the DPI and image compression type is optional. To change the image DPI and compression, complete the following steps:

1. In the **BRWEXP** table, set **TiffDPI** column to a DPI of your choice. The default image compression is 300 dots-per-inch.
2. In the same column, set image compression to one of the following. The default is standard Grade 4 compression.
    - G4FAX This is standard Grade 4 compression.
    - G3FAX This is standard Grade 3 compression.
    - LZWFAX This is LZW compression.
    - HUFFAX This is HUF compression.

4.3 Export PDF File

You can output an invoice as a searchable PDF file. To export a PDF file, complete the following steps:

1. Open the **BRWEXP** table.
2. Set the **OutputPDF** column to `TRUE`.
3. To set the name of the PDF file to match the document URN in the `<project>.ini` file, set the **PDFName** column to `URN`. Otherwise, the name used matches the original imported document.

Result
The PDF file is always written to the directory specified as the export directory on the runtime server instance settings for the instance that is carrying out the document export. If you do not specify an export directory, the default directory in the `DefaultExportPath` setting is used, the export fails, and the batch goes to status 750.

4.4 Export Data to Database Tables

You can configure database exports and customize the export of header data to add additional fields to documents. Data exports can be written to the following types of tables:

- Invoice header data table
- Invoice line-item data table
- Invoice general ledger-account coding table
- Invoice tax line item table

4.4.1 Configure Database Export

To activate output to a database, complete the following steps:

1. Navigate to the **BRWEXP** table.
2. Set the **ExportToDB** setting to `TRUE`.
3. Save the changes.
4. Navigate to the SQL section.
5. In the **NN_ConnectionString** parameter, enter a valid connection string for the SQL group.
6. Save the changes.

4.4.2 Configure Data Export to the Invoice Header Table

Configuring data export to the Invoice Header table is optional. Invoice Header table information includes data that is applied to the document as a whole, such as the invoice number, the invoice date and the vendor. The structure of the invoice header table must be such that each row has a single column that represents that unique key for the database record. By default, the system populates this column with the name of the image file name minus the file extension. However, the document URN can be mapped in the IMP section of INI file. To configure data export to the Invoice Header table, complete the following steps:

1. Open the `<project>.ini` file.
2. Navigate to the IMP section.
3. Set the DBKey parameter to `URN`.
4. Open the **BRWEXP** database table.
5. Enter a unique name for the invoice header table in the **DBHeaderTable** column.
6. To set the name of the column in the Invoice Header table which represents the unique record identifier, in the **DBHeaderKey** column, enter a unique name for the header column.
7. To verify whether invoice header information is written into the table as a new insertion or as an update to an existing record, in the **DBHeaderOperation** column, enter either `Insert` or `Update`.
8. Save the changes and then navigate to the BRWEXPHeader database table.
9. To configure which fields must be outputted and their destination columns, for the DBColumnName column, enter the names for the fields you want to have data output to. For example, to write an extracted invoice number with a name of INVNO, enter INVO in the **InvoiceNumber** row in the **DBColumnName** field.
10. Repeat the above step for all header fields for which database output is required, ensuring that the `DBColumnName` parameter is left blank for any fields that are not relevant for database output.
11. Save the changes.

Note: After configuring the field names in the BRWEXPHeader table, never change the field names.

4.4.3 Configure Data Export to the General Ledger Coding Items Table

The output of general ledger line items is optional. The system expects the destination General Ledger Coding Items table to have a composite key consisting of two fields. One is the document identifier, which is populated with a value identical to that passed into the Invoice Header table's unique key field, which must have the same technical column name. The second is a line item number index column, which the system sets from 1-n, where n is the number of general ledger line items. The line item fields that are outputted are controlled in the BRWEXPLines table, where there are more than 30 line item fields available for mapping. If a table is not specified, line items are not written exported. If a table is specified, but items key field are not populated, then the export fails and the batch goes to a state of 750.

To configure the output of line items to the General Ledger Coding Items table, complete the following steps:

1. Open the **BRWEXP** database table.
2. In the **DBHeaderKey** column, enter a unique document identifier name.
3. In the **DBLineItemsTable** column, enter the database table where the line items data is written.
4. In the **DBLineItemsKey** column, enter the line item number field.

5. Open the **BRWEXPGL** database table.
6. In the **DBColumnName** column, to indicate the field that data must be outputted to, enter the name of the destination database table column.
7. Repeat the above step for all line item fields for which database output is required, ensuring that the parameters are left blank for any fields that are not relevant for database output.
8. Save the changes.

4.4.4 Configure Data Export to the Tax Items Table

The output of tax line items is optional. The system expects the Destination Tax Line Item table to have a composite key consisting of two fields. One is the document identifier which is populated with a value identical to that passed into the Invoice Header table's unique key field, which must have the same technical column name. The second is a tax line item number index column, which the system sets from `1-n`, where `n` is the number of tax line items. If a table is not specified, tax line items are not exported. If a table is specified, but a tax line items key field is not populated, then the export fails and the batch goes to a state of 750.

To configure the output of line items to the Tax Items table, complete the following steps:

1. Open the **BRWEXP** database table.
2. In the **DBHeaderKey** column, enter the document identifier column.
3. In the **DBTaxTable** column, enter the database table where the tax item data is written.
4. In the **DBTaxKey** column, enter the tax line item number field.
5. Open the **BRWEXPTax** database table.
6. In the **DBColumnName** column, to indicate that a field must be outputted to, enter the name of the destination database table column.
7. Repeat the above step for all tax item fields for which database output is required, ensuring that the parameters are left blank for fields that are not relevant for database output.
8. Save the changes.

4.4.5 Add Aditional Fields to a Database Header Export

In addition to the default fields and values, you can add additional fields and values to a database header export.

To add a new fields to the database export, complete the following steps:

1. Open the **BRWEXPHeader** database table.
2. Add a new row for the export profile ID which represents the new field.
3. Open the **<project>.sdp** file with the Designer module.
4. Navigate to the `UserExitDBHeaderExport` subroutine and add the following line into the subroutine.

```
fnWriteDBHeaderField("InvCode",pWorkdoc.Fields("InvoiceCode").Text,
str.INIfileKeyName, strFieldValue)
```

If the database field being added is a date field, and that date field is in the Verifier output format that has been configured in the BRWDAT table, then the `fnWriteDBHeaderDateField` function must be used instead. This function has an identical interface, but converts the date into the export output format configured in the BRWDAT table.

5. Save the changes and close the `<project>.sdp` file.

4.5 Configure the XML File

The XML file can contain custom fields that you can configure based on your document types. You can divide the file into separate sections based on these type.

4.5.1 XML File Sections

The standard XML output file is divided into separate sections. Some sections are the same regardless of the document type, while other sections are dependent on the document type. The following sections are the same for all document types:

Document section

This section includes global document fields that are separate from the type of business document they represent, such as the scan date, priority flag, the URN, and so on.

Invoice header section

This section includes fields associated with the invoice header, such as the invoice number, the invoice date, the vendor ID, as so on.

Invoice line items section

This section includes invoice line item information written in line-by-line, such as the line item quantity, the unit price, the line item total, and material number.

General ledger coding section

This section includes all of the general ledger account entries associated with the invoices, such as the general ledger account number, the cost center, the tax code, and the amount.

Tax line items section

This section includes all of the tax line items associated with the invoices, such as the tax base amount, the tax code, and the tax amount.

Each of the sections has a specific tag, as does each field within those sections. All tags are configurable within the system configuration, as are those fields that are written into the file and which fields are not. The structure of the file is as follows:

<XML Encoding Header>

Configured using the `XMLEncodingHeader` parameter in the BRWEXP table.

<Document>

Configured using the `XMLFileHeader` parameter in the BRWEXP table.

[Document level fields]

<InvoiceHeader>

Configured using the `XMLInvoiceHeader` parameter in the BRWEXP table.

[Invoice header fields]

</InvoiceHeader>

<InvoiceLines>

Configured using the `XMLLineItemsHeader` parameter in the BRWEXP table.

[Invoice line item fields – one set of entries per line item]

</InvoiceLines>

<GLLines>

Configured using the `XMLGLLinesHeader` parameter in the BRWEXP table.

[GL item fields – one set of entries per GL line]

</GLLines>

<TaxLines>

Configured using the `XMLTaxHeader` parameter in the BRWEXP table.

[Tax item fields – one set of entries per tax line]

</TaxLines>

</Document>

4.5.2 Activate Output to an XML File

To set up output to an XML file, complete the following steps:

1. Open the **BRWEXP** table.
2. To set the name of the XML file to match the document URN, in the `<project>.ini` file, set the **XMLFileName** column to `URN`. Otherwise, the name used matches the original imported document.
3. In the **DefaultExportPath** column, enter the path to the output directory.
4. Optional. To set the file extension to XML, set the **XMLFileType** column to `TRUE`.
5. Optional. To support UTF-16 for non-western characters, such as Chinese, set the **XMLEncodingHeader** column to `UTF-16`. An XML file is not generated if the output data contains non-western characters and the encoding header is set to `UTF-8`.
6. The XML file is always written to the directory specified as the export directory on the runtime server instance settings for the instance that is carrying out the document export. If no export directory has been specified, the default export directory of `C:\Export`, which is set in the **DefaultExportPath** column, is used. If you want to use a different export directory, enter the directory in the `DefaultExportPath`.
7. Save the changes.

4.5.2.1 Additional XML File Section Options

You can configure the following document fields and section tags in the BRWEXPHeader table:

Document level field in BRWEXPHeader	Description
BatchName	This is the batch name as derived from a mapping to the document file name in the IMP section of the `<project>.ini` file.
ClientID	This is the client ID for the document.
DocClass	This is the AP Packaged Project document class.
DocumentLink	This is the link to the document image.
PriorityFlag	This is the flag to indicate whether the document has a high priority.
ScanDate	This is the scan date as derived from a mapping to the document filename in the IMP section of the `<project>.ini` file.
Status	This is the document status as set against the `XMLStatusExported` column in the BRWEXP table.

Document level field in BRWEXPHeader	Description
URN	This is the document URN as derived from a mapping to the document filename in the IMP section of the `<project>.ini` file. If no mapping has been set, the document name, minus the file path and the file extension, is used.

4.5.3 Define Field Output

After the XML output has been activated and you have configured the section tags, configure the fields that are written into the file and define how they are tagged. The field output is configured in the following tables. Tags for each field and section are defined in the XML file sections in the corresponding tables. To configure a field in the XML file, complete the following steps using the parameters in the BRWEXPHeader table, BRWEXPLines table, BRWEXPTax table, and BRWEXPGL table.

1. Populate the **XMLTag** column with the XML tag parameter for the given field.
2. Repeat the above step for all fields in the **XML** section.
3. Save the changes.

4.5.3.1 BRWEXPHeader Table

The following table contains parameters for mapping the AP Packaged Project header export fields into fields in the XML file or columns in a database:

Parameter	Type	Description
EXPProfileID	Integer	This is the export profile ID.
FieldName	Freetext	This is the name of the field.
XMLTag	Freetext	This column represents the tag that is used for the field in an exported XML file. If left blank, the field is not exported.
DBColumnName	Freetext	This is the technical name of the target field in the export database.
ProcessITag	Freetext	This column represents the tag that is used for the field in an exported XML file for ProcessIT. If left blank, the field is not exported.

4.5.3.2 BRWEXPLines Table

The following table contains the parameters to map the AP Packaged Project line item fields in the XML file or columns in a database:

Parameter	Type	Description
EXPProfileID	Integer	This is the export profile ID.
FieldName	Freetext	This is the name of the field.
XMLTag	Freetext	This column represents the tag that is used for the field in an exported XML file. If left blank, the field is not exported.
DBColumnName	Freetext	This is the technical name of the target field in the export database.

Parameter	Type	Description
ProcessITTag	Freetext	This column represents the tag that is used for the field in an exported XML file for ProcessIT. If left blank, the field is not exported.

4.5.3.3 BRWEXPTax Table

The following table contains the parameters for mapping the AP Packaged Project tax line item fields into fields in the XML file or columns in a database:

Parameter	Type	Description
EXPProfileID	Integer	This is the export profile ID.
FieldName	Freetext	This is the name of the field.
XMLTag	Freetext	This column represents the tag that is used for the field in an exported XML file. If left blank, the field is not exported.
DBColumnName	Freetext	This is the technical name of the target field in the export database.
ProcessITTag	Freetext	This column represents the tag that is used for the field in an exported XML file for ProcessIT. If left blank, the field is not exported.

4.5.3.4 BRWEXPGL Table

The following table contains the parameters for mapping the AP Packaged Project general ledger line item fields into fields in the XML file or columns in a database:

Parameter	Type	Description
EXPProfileID	Integer	This is the export profile ID.
FieldName	Freetext	This is the name of the field.
XMLTag	Freetext	This column represents the tag that is used for the field in an exported XML file. If left blank, the field is not exported.
DBColumnName	Freetext	This is the technical name of the target field in the export database.

4.6 Add a Field to the XML File

You can insert custom fields into any section of the XML file.

The process of adding a field to the XML file consists of two steps. The first step is to add a new row to the appropriate export table for the export profile ID to represent the new field. The second step is to add a script to the XML Export User Exit.

The export table that is used depends on the type of field that needs to be written out. The BRWEXPHeader table is used for fields that are written either to the document or invoice header level in the XML file. ; For line item fields, the BRWEXPLines table is used; for GL items, the BRWEXPGL table is used; and for tax line items, the BRWEXPTax table is used.

The exact naming of the field in the FieldName column of the BRWEXPHeader, BRWEXPLines, BRWEXPGL, and BRWEXPTax tables does not need to be the same as the technical name of the relevant field, but the names must be meaningful. The name of the field also must be unique for the export profile ID.

4.6.1 Add a New Row to the BRWEXPHeader Table

To add a new row to the BRWEXPHeader table, complete the following steps:

1. Open the **BRWEXPHeader** table.
2. Add a row for the export profile ID which represents the new field. The name of the new field in the **Fieldname** column must be unique for the export profile ID.
3. Populate the **XMLTag** column with the XML label text that is required for the field in the XML file. If it is left blank, the field is not written to the file.
4. Save the changes.

Note: No field in the row is permitted to have a `NULL` value. The Delete key can be used to create a blank entry, which is allowed by the database.

4.6.2 Add a Script into the XML Export User Exit

To add a script into the XML export user exit, complete the following steps:

1. With the WebCenter Forms Recognition Designer module, open the `<project>.sdp` file.
2. In the **Definition mode** section, highlight the **UserExits** class.
3. Right-click on the class and select **Show Script**.
4. Navigate to the UserExitXMLOutput user exit subroutine.
5. Add the following line into the sub-routine in the invoice header section of the sub-routine's select case structure:

   ```
   fnWriteXMLField("HCInvoiceCode", pWorkdoc.Fields("InvoiceCode").Text)
   ```

   The above code looks for a row in the BRWEXPHeader table with a FieldName of `InvoiceCode`, and then uses the associated XMLTag to write the value of the technical field `InvoiceCode` into the XML file.

   The field parameter for the field which appears in the document header or the invoice header section of the XML file must always be prefixed by `HC`. Hence, `InvoiceCode` is passed as `HCInvoiceCode` into the `fnWriteXMLField` function.

   Similarly, new line items fields entered in the BRWEXPLines table must be prefixed by `Table`, the new GL line item fields entered in the BRWEXPGL table must be prefixed with `GLTable`, and the new tax line item fields entered in the BRWEXPTax table must be prefixed with `TaxTable`. These prefixes are not case sensitive.

6. Save the changes.

   The `fnWriteXMLField` function is specially provided to condense the operation into a single command. The new field is added to the XML output in the invoice header section, and if the field content is `12345`, it appears as follows:

   ```
   <InvCode>12345</InvCode>
   ```

   If the field being added is a date field, and the date field is configured in the BRWDAT table in the Verifier output format, then the `fnWriteXMLDateField` function must be used instead. This function has an identical interface, but converts the date into the export output format configured in the BRWDAT table.

4.6.3 Insert Additional Fields Derived from the Document File Name

Additional values inserted into the XML file need not be tied to actual fields within the project file. You can also insert hard-coded values, or derive values from the document properties, for example, the document filename.For example, the document file name is `12345_ABCD_20090901_ACD.tif`. The business requirement is that the ACD component of the file name, which denotes a special accounting routing code, is passed into the XML file in the document header section.

To insert additional fields derived from the document file name, complete the following steps:

1. In the **BRWEXPHeader** table, add a row for the routing code with a `FieldName` of `RoutingCode`.
2. Use the global subroutine `SplitString` and the global function `fnGetFileName` to pass an additional parameter (`strRoutingCode`) from the file name into the XML document header section.

   ```
   fnWriteXMLField("HCRoutingCode", strRoutingCode)
   ```
3. Save the changes.

When the document is processed, the following line is written into the document header section of the XML file:

```
<RoutingCode>ACD</RoutingCode>
```

4.7 Write Data to a CSV File

The structure of the CSV file is configurable for up to five lines per document at the header level, and one additional line for each line item. It is also possible to configure the CSV file output at the document or batch level and to create multiple files depending on whether the invoice type is PO or NO-PO. You can output any number of different CSV files per document processed.

4.7.1 Configure the CSV File

Configuration of the CSV file output is performed in the BRWCSV table. Each row in the table is keyed by the export profile ID and an index column. To generate an additional file for each processed document, you can add new rows for each export profile ID.. This controls CSV file output at a global level to activate or deactivate all output files in one setting.

4.7.1.1 Activate CSV File Output at the Global Level

To activate CSV file output for the export profile ID, complete the following steps:

1. Open the **BRWEXP** table.
2. Set the **OutputCSVFile** column to `TRUE`.
3. Save the changes.

4.7.1.2 Activate CSV File Output for Each CSV File

To activate output for each individual CSV file, complete the following steps. The naming of the CSV file depends upon whether output is required on a per-document or a per-batch basis.

1. Open the **BRWCSV** table.
2. Set the **OutputFile** column to `TRUE`.

3. If one file per batch is required, set the **CombinedFilePerBatch** column to `TRUE`. If one file per document is required, set the column to `FALSE`.
4. If you set the **CombinedFilePerBatch** column to `TRUE` in step 3, set the Document Grouping option for import in the RTS instance settings to 1 folder per batch or 1 batch per subdirectory, 1 folder per batch.
5. To specify a date format for each individual CSV file output, in the DateFormat column, enter one of the following date formats:
   - `YYYYMMDD`
   - `MMDDYYYY`
   - `DDMMYYYY`
6. To specify the separator style for each individual CSV file output, in the DateSeparator column, enter one of the following separator formats:
   - – (dash)
   - / (forwared slash)
7. To restrict CSV file output depending on the contents of the Invoice Type field, which denotes whether the document is purchase order-related, complete one of the following:
   - If the field is purchase order related, in the **InvoiceType** column, enter `PO`.
   - If the field is not purchase order related, in the **InvoiceType** column, enter `NPO`.
   - If the field is for both purchase order and non-purchase order invoices, leave the column blank.
8. To output an additional copy of the original image to the destination directory, set the **OutputImage** column to `TRUE`.
9. Save the changes

The output file is written to the export directory configured against the **Filepath** column. If a file path is not specified, the system uses the export file path set against the RTS instance carrying out the export step in the AP Packaged Project workflow. If no export directory is specified there, the default export file path set in the BRWEXP table is used. If this is also blank, or any file path is invalid, export of the CSV file fails and the document goes to state 750.

4.7.2 Add a New Header Field to the CSV File

Custom fields can be written into the CSV file. To add custom fields to the file, complete the following steps:

Prerequisite

If the CSV field being added is a date field and that date field is in the Verifier output format configured in the BRWDAT table, then in the following steps you must use the `fnWriteCSVDateField` function instead. This function has an identical interface, but converts the date in the date format configured for the CSV file group. If no group format is set, the date is formatted according to the output date format setting in the BRWDAT table.

1. With the WebCenter Forms Recognition Designer module, open the `<project>.sdp` file.
2. Navigate to the **Definition mode** section and highlight the **UserExits** class.
3. Right click the class and select **Show Script**.
4. Navigate to the **UserExitCSVFile** user exit subroutine.
5. To add the **InvoiceCode** custom field to the file, add the following line to the subroutine:

```
 fnWriteCSVField(strRecordText,strKey,
"<%ZIC>",pWorkdoc.Fields("InvoiceCode").Text)
```

6. Save the changes and close the script.

4.7.3 Add a New Line Item Field to the CSV File

Custom line item fields can be written into the CSV file. To add a new line item field, complete the following steps:

1. With the WebCenter Forms Recognition Designer module, open the `<project>.sdp` file.
2. Navigate to the **Definition mode** section and highlight the **UserExits** class.
3. Right-click on the class and select **Show Script**.
4. Navigate to the **UserExitCSVFileLine** user exit subroutine.
5. To add the line item field to the file, add the following line to the subroutine:

```
 fnWriteCSVField(strRecordText,strKey,"<%myfield>",
"myvalue")fnWriteCSVFieldDateField(strRecordText,strKey,"<%myfield>",
"mydate")
```

6. Save the changes and close the script.

4.8 Export Data to ProcessIT

ProcessIT is a packaged account payable workflow solution for Oracle eBusiness Suite. AP Project includes a standard integration so that, upon export, documents may be sent in XML format to the ProcessIT interface table within the Oracle database. It is also possible to export the XML as a flat file in addition to, or instead of, writing the data into the interface table.

4.8.1 ProcessIT XML File Sections

The ProcessIT XML file is divided into three sections, one for the header, one for the line items and one for the tax line items.

The format of the file is as follows:

```
<PITXML Header 1>
<PITXML Header 2>
<DOCTYPE>
--- Header fields
<LINEITEM>
       ---Line                          item                          fields
</LINEITEM>
<TAXITEM>
---                    Tax                     item                    fields
</TAXITEM>
</ARCDOC>
```

A `<LINEITEM>` block is created for each line item exported.

In the above structure, the PITXML Header 1 and PITXML Header 2 are configurable in the BRWEXP table via parameters `PITXMLHeader1` and `PITXMLHeader2` respectively.

The value set against `PITXMLHeader2` must follow the convention of `<ARCDOC … >` to match the closing `</ARCDOC>` XML tag or else the XML file is not generated correctly and data export fails.

4.8.2 Set the DOCTYPE Field

Within the ProcessIT XML file, the `DOCTYPE` field, which is different from the AP Project Document Type field (which denotes whether the document is an invoice or a credit memo), can be used to categorize and control the process flow of the document when it reaches ProcessIT. The `DOCTYPE` field is always written into each and every XML file generated.

Within the BRWEXP table, the `DOCTYPE` for each export profile ID can be set to a hard value via the `PITDocType` parameter. If this is left blank, the client name is used.

A user exit `UserExitSetPITDocType` is also available for a developer to use a custom naming convention for the `DOCTYPE` field by changing the value of the user exit interface parameter `strPITDocType`.

4.8.3 SelectFields for Inclusion in the ProcessIT XML File

Fields for output into the ProcessIT XML file are configurable both at header and line item level. The XML tag that is used can also be changed.

For document header fields, the parameter `ProcessITTag` is used in the BRWEXPHeader table. If a value is populated in this column, this designates the field as relevant for output. The entry in this field controls the value of the tag in the XML file.

For example, if, in the BRWEXPHeader table, the row for the invoice number field has a value of `INVNO` in the **ProcessITTag** column in the PIT XML file, and if `1234` is extracted as the invoice number, the following is written:

```
<INVNO>1234</INVNO>
```

The control of line item field output and tags is done in exactly the same way, except in the BRWEXPLines table.

The default installation of the AP Project database includes the standard tag names that ProcessIT expects at both header and line item level.

It is possible to add new fields at the header level only simply by inserting a new row into the BRWEXPHeader table.

4.8.4 Document Type Field Settings

Within AP Project, the standard Document Type field is set to either `INVOICE` or `CREDIT`. The ProcessIT interface expects `STANDARD` for an invoice and `CREDIT` for a credit. These values can be configured using parameters `PITInvoice` and `PITCredit` respectively in the BRWEXP table and the system performs the appropriate substitution at time of export.

If no values have been configured, then the AP Project defaults of `INVOICE` and `CREDIT` are used.

4.8.5 Provide the Image Details within the ProcessIT XML File

Integration with ProcessIT expects an early archiving strategy. That is, the document will already have been archived prior to its receipt in AP Project.

The details of the archived image are communicated to ProcessIT via the XML file, specifically via the `XXIT_FILENAME` tag. This must always be mapped to the URN field in the BRWEXPHeader table. The URN is set to the image file name (minus file extension), or the component of the image file name that is designated as the URN in the IMP section of the AP Project INI file.

4.8.6 ProcessIT Amount and Date Output Formats

The formatting of dates and amounts within the ProcessIT XML file is configurable.

For the amount fields, the decimal and thousand separators can be configured in the BRWAMT table via parameters `ExportThousandSeparator` and `ExportDecimalSeparator`. The number of decimal places is configured by the DecimalPlaces column against the field in the BRWFLD table.

For date fields, the format of the date (e.g. YYYYMMDD, DDMMYYYY or MMDDYYYY) is controlled via parameter ExportFormat in the BRWDAT table, the separator (if any is required) is controlled via parameter ExportSeparator also in the BRWDAT table.

4.8.7 ProcessIT Priority Flag

The ProcessIT XML schema includes a priority flag using the Urgent tag. This corresponds to the AP Project standard PriorityFlag flag, except that ProcessIT requires a single character Y or N as opposed to the AP Project standard of YES or NO. To accommodate this difference, the system only sends over the first character of the PriorityFlag field content.

4.8.8 ProcessIT External Site ID

Oracle EBS uses external and internal IDs for both the vendor ID and the site ID. Within AP Project, the external vendor ID is accommodated, but the Verifier display is restricted to the internal site ID. ProcessIT requires the external vendor ID and site ID to be exported against the SUPPLIER and SUPPLIER_SITE XML attributes.

The external vendor ID is available and can be mapped in the BRWEXPHeader table against the VendorID row. The mapping for the site ID must continue to be against the row for SiteID. At the time of writing out the file, the system looks for an external site ID in the vendor master data as mapped in the BRWSRC table, hence this configuration needs to be in place. If it is found, this external site ID is written into the XML file; if no external site ID is available, the internal site ID, as captured in the AP Project file SiteID is passed.

4.8.9 ProcessIT Line Item Data

ProcessIT requires the quantity, unit price and total columns to be populated for all line items.

Hence, during export to ProcessIT, if no quantity is available. AP Project puts in 1; if no unit price is available, the system divides the line item total by the quantity and this value is used for the unit price.

4.8.10 Include Additional Field Metadata in the ProcessIT XML File

AP Project includes an option to include metadata for each field written into the ProcessIT XML file where available. This metadata includes the following:

- The field page number (PageNumber)
- The field co-ordinates (Position - left, top, width, height) measured in pixels
- The field validity status (Valid – set to TRUE or FALSE)
- Best candidate confidence weighting (Confidence)
- Second best candidate confidence weighting (Confidence2)

This metadata is included within the file as part of the field XML tag. For example, for the output of an invoice number where the extracted value is 2679.

```
<InvoiceNumber          Valid="TRUE"          Confidence2=".282649517059326"
Confidence="1.61400547027588"                                  PageNumber="1"
Position="2102,627,89,29">2679</InvoiceNumber>
```

Internally, AP Project uses a zero-based index for the page number, so 0 denotes page 1, 1 denotes page 2, and so on. For the purposes of the ProcessIT metadata, the page number is incremented so that 1 denotes page 1, 2 denotes page 2 etc.

If no positional data is available, the PageNumber and Position attributes are not included. If line pairing is being used, field positional information is not available for paired line items.

To activate the output of additional field metadata, the parameter `PITIncludeFieldPositions` must be set to `TRUE` in the BRWEXP table.

4.8.11 Write the ProcessIT XML Output to a Directory

To output the Process IT XML file into a directory, the parameter `OutputPITXML` in the BRWEXP table must be set to `TRUE`. The system subsequently writes the XML file to the export directory specified in the properties of the Runtime Server instance carrying out the export.

The naming of the XML file is controlled using the `PITKey` parameter. If this is set to blank, then the entire image file name is used as the XML output filename. If it is set to `URN`, then the portion of the file name mapped to the URN in the IMP section of the INI file is used. The XML output file extension is set using the parameter `PITXMLFileType`, which is `XML` by default if left blank.

The document export fails if the XML file cannot be created.

There is also an option to write out a *ready* file for each ProcessIT XML file outputted. This can be useful to ensure that a downstream system does not pick up an XML file before AP Project has finished writing it out in full. To activate the *ready* file output, the parameter `OutputPITRDYFile` in the BRWEXP table must be set to `TRUE`.

The *ready* file is provided the same file name as the corresponding XML file, but with a file extension of `RDY`. The AP Project export fails if the RDY file cannot be written out.

4.8.12 Write the ProcessIT XML Output to the ProcessIT Interface Table

To configure the system to send the ProcessIT output XML file to the ProcessIT interface table within the Oracle database, the following must be configured within the BRWEXP table:

- Parameter `SendToProcessIT` must be set to `TRUE`.
- The SQL connection group that points to the Process IT Oracle database must be maintained against the parameter `PITSQLConnectionGroup`.
- The name of the Process IT interface table must be maintained against the parameter `PITDBTable` – the default table name is `XXIT3_INTERFACE`.

When sending the data to the interface table, the following columns are updated:

- INTERFACE_ID – This is the unique key for the row in the ProcessIT interface table – it is set to the next value of a standard ProcessIT trigger `XXIT3_INTERFACE_SEQ.NEXTVAL`.
- SOURCE – This is read from the parameter `PITSourceName` in the BRWEXP table – if it is blank, `PERCEPTIVE` is used.
- SOURCE_ID – This is set to the document URN, which is the whole image file name (minus extension) if no component of the file name has been set as the `URN` in the IMP section of the AP Project INI file;
- FLOW_STATUS – This is hard-coded to a set value of `UNPROCESSED` – ProcessIT subsequently changes this status as the document is picked up for processing.
- INTERNAL_XML – This is populated with the XML file.

If a problem is encountered when writing the data to the interface table, document export fails. Detailed information regarding the error can be found in the export RTS instance log file.

4.8.13 ProcessIT XML User Exit

User exit `UserExitPITXMLOutput` is provided to allow a developer to edit the body of the XML file created for ProcessIT.

It is called once the XML document is assembled by the system based on settings within the AP Project configuration tables, but before it is written out to the export directory or sent to the

ProcessIT interface table. The XML document is passed into the user exit via the `xmlDoc` parameter.

4.9 OCR XML File Export

The OCR XML file export outputs a file that contains the following information:

- All the document OCR words with positional information
- All the document field data (excluding tmp fields) with positional information and validity status, including cell-by-cell content of table data
- All the field candidate information (optional)

For example, the file can be used by a downstream system that has a component similar to the Verifier application, thus avoiding the need for a second OCR.

4.9.1 Write the ProcessIT XML Output to a Directory

To output the Process IT XML file into a directory, the parameter `OutputPITXML` in the BRWEXP table must be set to `TRUE`. The system subsequently writes the XML file to the export directory specified in the properties of the Runtime Server instance carrying out the export.

The naming of the XML file is controlled using the `PITKey` parameter. If this is set to blank, then the entire image file name is used as the XML output file name. If it is set to `URN`, then the portion of the file name mapped to the URN in the IMP section of the INI file is used. The XML output file extension is set using the parameter `PITXMLFileType`, which is `XML` by default if left blank.

The document export fails if the XML file cannot be created.

There is also an option to write out a *ready* file for each ProcessIT XML file outputted. This can be useful to ensure that a downstream system does not pick up an XML file before AP Project has finished writing it out in full. To activate the *ready* file output, the parameter `OutputPITRDYFile` in the BRWEXP table must be set to `TRUE`.

The *ready* file is provided the same file name as the corresponding XML file, but with a file extension of `RDY`. The AP Project export fails if the RDY file cannot be written out.

4.9.2 Write the ProcessIT XML Output to the ProcessIT Interface Table

To configure the system to send the ProcessIT output XML file to the ProcessIT interface table within the Oracle database, the following must be configured within the BRWEXP table:

- Parameter `SendToProcessIT` must be set to `TRUE`.
- The SQL connection group that points to the Process IT Oracle database must be maintained against the parameter `PITSQLConnectionGroup`.
- The name of the Process IT interface table must be maintained against the parameter`PITDBTable` – the default table name is `XXIT3_INTERFACE`.

When sending the data to the interface table, the following columns are updated:

- INTERFACE_ID – This is the unique key for the row in the ProcessIT interface table – it is set to the next value of a standard ProcessIT trigger `XXIT3_INTERFACE_SEQ.NEXTVAL`.
- SOURCE – This is read from the parameter `PITSourceName` in the BRWEXP table – if it is blank, `PERCEPTIVE` is used.
- SOURCE_ID – This is set to the document `URN`, which is the whole image file name (minus extension) if no component of the file name has been set as the URN in the IMP section of the AP Project INI file;

- FLOW_STATUS – This is hard-coded to a set value of `UNPROCESSED` – ProcessIT subsequently changes this status as the document is picked up for processing.
- INTERNAL_XML – This is populated with the XML file.

If a problem is encountered when writing the data to the interface table, document export fails. Detailed information regarding the error can be found in the export RTS instance log file.

4.9.3 ProcessIT XML User Exit

User exit `UserExitPITXMLOutput` is provided to allow a developer to edit the body of the XML file created for ProcessIT.

It is called once the XML document is assembled by the system based on settings within the AP Project configuration tables, but before it is written out to the export directory or sent to the ProcessIT interface table. The XML document is passed into the user exit via the `xmlDoc` parameter.

4.10 OCR XML File export

The OCR XML file export outputs a file that contains the following information:

- All the document OCR words with positional information
- All the document field data (excluding tmp fields) with positional information and validity status, including cell-by-cell content of table data
- All the field candidate information (optional)

For example, the file can be used by a downstream system that has a component similar to the Verifier application, thus avoiding the need for a second OCR.

4.10.1 OCR XML File Structure

The structure of the XML file is as follows:

```
<OCR XML Header>
<Document>
<Words>
       <Word>
              <ID>
              <Text>
              <Top>
              <Left>
              <Height>
              <Width>
       </Word>
</Words>
<Fields>
<Field>
       <Name>
       <Text>
       <Valid>
       <ErrorDescription>
       <PageNr>
       <Top>
       <Left>
       <Height>
       <Width>
```

The following appears for table fields only:

```
       <Rows>
              <Row>
```

```
                        <ID>
                        <Columns>
                               <Column>
                                      <Name>
                                      <Text>
                                      <Valid>
                                      <ErrorDescription>
                                      <PageNr>
                                      <Top>
                                      <Left>
                                      <Height>
                                      <Width>
                               </Column>
                        </Columns>
                 </Row>
          </Rows>
```

The following appears if candidate information is required (non table fields):

```
          <Candidates>
                 <Candidate>
                        <ID>
                        <Text>
                        <PageNr>
                        <Top>
                        <Left>
                        <Height>
                        <Width>
                 </Candidate>
          </Candidates>
   </Field>
   </Fields>
   </Document>
```

4.10.2 OCR XML File Configuration Options

To activate the output of the OCR XML file, the `OutputOCRXMLFile` parameter must be set to `TRUE` in the BRWEXP table. The output file is created in the export directory set against the export RTS instance.

The naming of the file can be set to either the image file name (excluding the file extension) or the part of the image file name that is set as the document URN in the IMP section of the project INI file. This is done using the `OCRXMLFileKey` parameter, which must be set to `URN` if the URN is used, or left blank if the image file name is used.

The file extension is also configurable using the `OCRXMLFileType` parameter, which is `XML` by default if no value is set here.

The OCR XML header can be configured via the `OCRXMLHeader` parameter. This subsequently appears as the first line of the exported XML file.

The `IncludeCandidateInfo` parameter must be set to `TRUE`, if for each non-table field, the candidate data must also be included for each field.

4.11 Set Up a Custom Export

If you have a required data export and the existing export options do not support the data export's format, or if you need to export data for a custom base class, you must create a custom export. The custom export must be scripted and executed within a special user exit. The following sections describe how to implement a custom export.

The user exit is called once for each document that is exported. Once a document is exported, the export history is updated against the document so that it is not unintentionally exported a second time. The history can be cleared by resetting the document back to state 200. If an export is not successful, the user exit is called again during the next attempt.

The script contents of a user exit can be set to anything that your business needs require.

You must check the document class before developing any script that refers to fields using hard-coded field names, particularly if the project uses custom base classes. If a field that does not exist is referenced against the document class, it results in a runtime error. The `fnGetBaseClass` global function, described in the Global Variables Script Class, can be used to check the document class.

4.11.1 Configure and Activate a Custom Export

The script contents of the user exit can be set to anything that is required. To configure and activate a custom export, complete the following steps:

1. With the WebCenter Forms Recognition Designer module, open the `<project>.sdp` file.
2. Navigate to **Definition** mode and highlight the **UserExits** class.
3. Right-click on the class and select **Show Script**.
4. Navigate to the **UserExitCustomExport** user exit subroutine.
5. Configure the parameters listed in the following table:

Parameter Name	Description
pWorkdoc	This is the standard Workdoc object that provides access to all document field information (including the originally extracted line item data), the document class name, the document OCR text and the document file name.
ExportPath	This is the destination folder for file output. This value is taken from the export file path configured on the RTS instance responsible for document export. If the RTS instance path is blank, then the export path is set to the value held against the system configuration parameter `EXP_VL_DefaultExportPath`.
strDocLink	This is the path to the image of the document, which could be stored either in a storage director or the batch directory, or could be a URL to retrieve the image from an archive.
LineData	This multi-line array contains the line items available for export. This is populated for all documents classified as invoices where line items are relevant. Refer to the LineData structure section in this document for more information.
GLData	This is the array based on the accounting data type defined on the global variables script level. This array is populated if the system determines that general ledger coding entries are required for the document being processed. In the standard solution, this is only populated for invoices where a miscellaneous charge extracted either at header or line item level is configured to be posted to a GL account in the BRWMSC table of the system configuration. Refer to the AccountingData structure section in this document for more information.
TaxData	This is the array containing the total tax amount that corresponds to each tax code determined during the automatic tax calculation procedure. Refer to the AccountingData structure section in this document for more information.
blLinesRequired	This is the flag indicating whether line item export is relevant for the invoice based on the invoice characteristics, any invalid reasons set, and the configuration in the BRWTAB table. If set to `TRUE`, line items must be exported.
Address	This is the vendor address structure. This contains the address details for the document vendor. Refer to the Address structure section in this document for more information.

Parameter Name	Description
Flags	These are the document validation flags. This structure contains document-specific flags which can be used to determine what data must be exported. Refer to the Flags structure section in this document for more information.

6. To activate the custom export, in the BRWEXP table, set the **CustomExport** column to `TRUE`.
7. To overwrite any history checks, set the **RedoAllExports** column to `TRUE`.
8. Save the changes.

5 Add Custom Scripts

Adding custom script must always be the last resort for the correction of extraction problems. Each individual vendor learnset has its own script class, which means that the code can be added without interfering with the operation of any other part of the project.

Example of custom scripts include the following:

- Correcting OCR problems where the correct result are known for a given vendor.
- Defaulting mandatory field values which the vendor does not actually state on the invoice.
- Improving line item extraction for vendors who present the information in a way that is not fully supported by the a table extraction engine.

6 Configure and Add Clients

AP Packaged Project supports multiple configuration types to operate within a single installation.

A basic installation creates a single client with a client ID of 0 (zero), and this is the default client the system uses.

Each document that passes through the system is preassigned to a client, and is the client that controls the following:

- The overall document flow.
- The fields that are extracted.
- The mandatory and optional fields, and their corresponding validation rules.
- The data sources that are used for field validation.
- How data is exported.

When you design an AP Packaged Project client, you must consider how the client is utilized for your business needs. For example:

- If the end user is a BPO, a client can be used to represent a single customer of the BPO or a division of a single customer.
- If you have one user working in multiple regions or with multiple divisions with their own requirements, a client can be used to represent each region or division.
- If you have one user working with multiple ERP systems, each ERP system can be set up as an individual client for the different ERP-system connections and processing rules.

Client settings and properties are contained in the BRWClient table in the AP Packaged Project database. A basic installation creates a single client with a client ID of 0 (zero), and this is the default client the system uses.

The columns contained in the BRWClient table and their uses are described in the following table. The sections that follow provide more information about configuring clients.

Column	Description
ClientID	This is the unique ID of the client and must always be set to an integer value.
ProfileID	This is the ID of the profile assigned to the client. The profile controls what fields are extracted and how they are validated. More than one client may share the same profile ID if the extraction and validation requirements are identical.
ExportProfileID	This is the ID of the export profile assigned to the client. The export profile ID controls how data is exported for a client. More than one client may share the same export profile ID if the export requirements are identical.
ClientName	This is a free text string containing the name of the client. This data is written to the reporting database for each document assigned to a client.
InstructionsProfileID	AP Packaged Project includes a button on the dynamic verifier form which, when pressed, displays specific processing instructions for a particular client. The instructions profile ID is the ID assigned to a particular set of instructions that are held in the BRWINSTR table.
ForceVerify	This is a flag that controls whether all documents for a client must be routed to the Verifier. If this column is set to `TRUE`, all documents are routed to the verifier. If it is set to `FALSE`, only documents requiring review by a user attention are routed to the verifier.
ClientGroup	This is the ID of the client group to which the client belongs. It is an integer value that can be set by the system administrator. You use this group to give users access to documents belonging to specific clients.

Column	Description
RequiresReview	This is a flag that indicates whether documents assigned to a client must always be subject to review by a user after the document has been routed through the Verifier.
VendorPartition	The ID of the vendor master data partition to be used by the client.
EmployeePartition	The ID of the employee master data partition to be used by the client.
POPartition	The ID of the purchase order data partition to be used by the client.
TaxPartition	The ID of the tax code data partition to be used by the client.
Priority	When documents are imported into WebCenter Forms Recognition, they are placed in batches and each batch is assigned a priority. This priority controls the order by which the runtime server component of WebCenter Forms Recognition processes the batches and the order in which the documents appear in the Verifier. The priority scale runs from 1 to 9, and 1 is the highest level of priority. If you set this to 1, all batches from this client have a priority of 1

6.1 Add a New Client with New Requirements

Client settings and properties are contained in the client configuration table within Solution Configuration Manager. Using this table, you can edit an existing client, copy a client, or create a new client.

To create a new client, complete the following steps within Solution Configuration Manager:

1. Select the project to be used.
2. Select **Client Settings** from the **Settings** drop-down list. The client configuration table is displayed.
3. Scroll to the last row in the table where you have the option to create a new entry.
4. Populate **Client ID** with the client ID you would like to use. The ID must be a unique numeric integer.
5. Enter a short description of your client in **Client Name**.
6. Choose the processing profile you wish to assign to the client by selecting the processing profile ID number from the **Processing Profile ID** drop-down list.
7. Choose the export profile you wish to assign to the client by selecting the export profile ID number from the **Export Profile ID** drop-down list.
8. Choose the instructions profile you wish to assign to the client by selecting the instructions profile ID number from the **Instructions Profile ID** drop-down list.
9. Select the **Force Verify** check box if you want all documents belonging to this client to stop in Verifier for a user to review.
10. Enter the client group you want the client to be assigned to in **Client Group**. A client can only be assigned to a single client group. The client group is used to assign users to client documents in the user management table.
11. Select the **Requires Review** check box if you want all documents belonging to this client to be subject to the document supervisor review step.
12. Choose the vendor partition you wish to assign to the client by selecting the vendor partition ID number from the **Vendor Partition** drop-down list. Use of vendor partitions must be active in **Global Settings > General Settings** for the selection to take effect.
13. Choose the employee partition you wish to assign to the client by selecting the employee partition ID number from the **Employee Partition** drop-down list. Use of employee partitions must be active in **Global Settings > General Settings** for the selection to take effect.
14. If you are using a partitioned database table to validate purchase order numbers, choose the purchase order partition you wish to assign to the client by selecting the purchase

order partition ID number from the PO Partition drop-down list. Use of PO partitions must be active in **Processing Settings > PO Number Settings > PO Number Validation** for the selection to take effect.

15. If you are using a partitioned tax table for determining the VAT type tax code during line pairing, choose the tax partition you wish to assign to the client by selecting the tax partition ID number from the **Tax Partition** drop-down list. Use of tax partitions must be active in **Processing Settings > Tax Settings > Tax Configuration** for the selection to take effect.
16. Select a priority indicator to be assigned to client documents by selecting a number from the **Priority** drop-down list. A value of 0 indicates the highest level of priority. A value of 9 indicates the lowest level of priority. The default value assigned is 5. Any value outside of the range 0-9 is ignored. Documents set to a high priority appear at the top of the Verifier worklist and receive priority processing from the Runtime Server. If you want to use this feature, you must configure the Runtime Server instances to import documents as batches into the database. The client priority level overrides any priority level set in the Runtime Server instance configuration as long as the **Set Batch Priority From RTS** check box is not selected in **Global Settings** > **General Settings**.
17. Click the **Insert** hyperlink to save your changes

6.2 Assign Documents to a Client

Documents must be preassigned to clients prior to being captured by the system. You can assign documents using the document file name or by using a database lookup.

6.2.1 Assign a Client ID using a Document File Name

WebCenter Forms Recognition uses a parameter in the image file name to identify the client a document must be assigned to. Therefore, a client ID must be embedded within the image file name and separated by an underscore. The part of the file name that represents the client ID is specified in the IMP section in the `<project>.ini` file in the `IMP_VL_ClientID` parameter with the word *COMPONENT* followed by a number that indicates the client ID's position in the file name.

To assign a client ID using a document file name, complete the following steps:

1. Open the `<project>.ini` file.
2. Navigate to the IMP section.
3. Set the `IMP_VL_ClientID` parameter to `COMPONENT2`.

   For example, processing a document using the component assigned to client 2 has a client ID embedded in the file name as follows: `12345_2_20120901.tif`. The first component of the file name is `12345`, the second component and client identifier is `2`, and the third component is `20120901`.

4. If the file name for client 2 is `12345_20120901_2.tif`, set the `IMP_VL_ClientID` parameter to `COMPONENT3`.

   Note: If this parameter is not set or the file component does not exist, the system processes the document using the default configuration assigned to client zero.

5. Save the changes and close the file.

6.2.2 Assign a Client ID using a Database Lookup

You can derive the client ID with a database look-up based on either the document file name or the URN component of the document file name. If the client ID database look-up is configured, it

takes precedence over any client ID file name mapping configured elsewhere in the IMP section of the INI file.

To assign a client ID using a database lookup, complete the following steps:

1. Open the `<project>.ini` file.
2. Navigate to the IMP section.
3. Set the client database lookup in the `IMP_OP_ImportClientIDFromDB=YES` parameter.
4. Set the `IMP_VL_SQLConnectionGroup=01` connection group parameter. If no connection group is specified, then connection group is defaulted to 01.
5. Set the `IMP_VL_ClientKey=` parameter. If left blank, then the entire file name is used to query the database table. If only the `URN` component of the file name must be used, then set this parameter to `URN`.
6. Map the database table name and column as follows:

```
ID IMP_VL_DBClientID=Client

IMP_VL_DBTableName=Client

URN IMP_VL_DBURN=Document
```

6.2.3 Client ID Errors

The system displays an error if any of the following conditions occur:

- The client ID lookup table is not mapped or is incorrect.
- The column names for the `URN` and client ID have not been mapped or are incorrect.
- The `ClientKey` parameter has been set to `URN`, but `IMP_VL_URN` has no component assigned or the component does not exist in the file name.
- A connection to the client lookup database cannot be established.
- There is no entry in the lookup table for the document file name or `URN`.
- There is an entry in the lookup table for the document file name or `URN`, but the client ID is not populated or is not numeric.
- The client ID read from the table does not exist in the BRWClient table.
- There are multiple entries in the look-up table for the same document file name or `URN` with different client IDs.

6.3 Working Without a Vendor Partition

If the project requires no vendor partition, either because it is for a single client or multiple clients that pool the same set of vendor data, complete the following steps:

1. Open the `<project>.ini` file.
2. Set the `GRL_OP_ActivateVendorFiltering` parameter to `NO`.
3. Save and close the INI file.

6.4 Set Up the Vendor Master Partition

AP Packaged Project supports multiple sets of vendor data within the same project file. Each set of vendor master data is referred to as a *vendor partition* and is assigned its own partition ID within the system configuration. Partitions IDs are assigned to clients in the BRWClient table. Multiple clients may share the same vendor partition.

When the vendor is being determined by the system at runtime, the system only takes into account vendors that belong to the vendor partition assigned to the document client. Within the

Verifier application, when the user executes a search, only vendors assigned to that client are included in the results.

The choice to implement the vendor master partition must be made during the installation and setup process.

6.4.1 Implement Partitioning

The following is a high-level overview of the steps for implementing partitions:

1. Activate partitioning within the `<project>.ini` file.
2. Register the vendor partition in the BRWVendorPartition table.
3. Assign the partition ID to the client.
4. Populate the vendor master table.
5. Create a user DSN for the vendor master table.
6. Configure the ASA sections in the `<project>.ini` file.
7. Create the ASE pool.
8. Configure the BRWSRC table.

6.4.1.1 Activate Partitioning

To activate partitioning, complete the following steps:

1. Navigate to the Global directory.
2. Open the `<project>.ini` file.
3. Set the `GRL_OP_ActivateVendorFiltering` parameter to `Yes`.
4. Set the `GRL_VL_VendorFilterColumn` parameter to `PartitionID`.
5. Save and close the INI file.

6.4.1.2 Register the Partition

To register the partitions, complete the following steps:

1. In the AP Packaged Project **database**, open the BRWVendorPartition table.
2. Populate a row with a partition ID. This value must be an integer.
3. Optional. Add a description. Its naming must indicate what it represents.
4. Save the changes.

6.4.1.3 Assign the Partition ID to the Client

To assign the partition ID to the client, complete the following steps:

1. In the AP Packaged Project database, open the BRWClient table.
2. In the **VendorPartition** column, enter the ID of the newly registered partition for the appropriate client.
3. Save the changes.

6.4.1.4 Populate the Master Tables

Included within the AP Packaged Project database are sample BRWVendorMaster master table. These contain an example structure that the tables must follow. To populate the master tables, complete the following steps:

1. Open the **BRWVendorMaster** table.
2. Populate the columns with the parameters listed in the following table:

Column	Usage
Index	This is a unique identifier for the record in the table. This value must be set to the partition ID followed by a hyphen and then the client's vendor ID.
PartitionID	This is the partition ID for the vendor master as set in the BRWClient and BRWVNDPartition tables.
ID	This is the client's vendor ID.
Address1	This is the street address of the vendor.
City	This is the city of the vendor.
Zip	This is the zip code of the vendor.
Country	This is the two-character ISO-code representing the vendor's country of origin, such as GB = United Kingdom; US = United States; DE = Germany; CN = China.

3. (Optional) Populate additional columns according to your business needs.
4. Save the changes.

6.4.1.5 Create a User DSN for the Vendor Master Tables

WebCenter Forms Recognition requires a user DSN to be created which reflects a connection to the AP Packaged Project database using SQL Server-based authentication.

Create the user DSN using Administrative Tools on a Windows machine. For more information about creating a user DSN, refer to Windows documentation.

6.4.1.6 Configure ASA Section in the <project>.ini File

To configure the ASA section in the `<project>.ini` file, complete the following steps:

1. Open the `<project>.ini` file.
2. In the **ASA** section, configure the following settings, replacing myDSN with the name of the user DSN, and set `myUSERNAME` and `myPASSWORD` with the appropriate database credentials.

```
ASA_VL_01_Class=Invoices
ASA_VL_01_Fieldname=VendorASSA
ASA_OP_01_AlphaNum=YES
ASA_OP_01_PoolRelative=YES
ASA_VL_01_PoolPath=
ASA_VL_01_PoolDirectory=Pool
ASA_VL_01_PoolName=Vendor
ASA_OP_01_FileRelative=YES
ASA_VL_01_ImportPathFilename=
ASA_VL_01_ImportFilename=
ASA_VL_01_ImportODBCDSN=myDSN
ASA_VL_01_ImportODBCSelect=select * from BRWVendorMaster
ASA_VL_01_ImportODBCUser=myUSERNAME
ASA_VL_01_ImportODBCPWD=myPASSWORD
ASA_VL_01_AutoImportOption=ODBC
```

3. Save and close the INI file.

6.4.1.7 Create the ASE Pool

To create the ASE pool, complete the following steps:

1. Using the Designer, open the `<project>.sdp` file and navigate to the **VendorASSA** field in the Invoices class.
2. Display the field settings.
3. On the **File Import** tab, to import the pool from the table, click **Import**.
4. Result A message stating that the pool was created is displayed. If an error message is displayed, the vendor master table has not been configured correctly. Correct any configuration errors and reimport the pool.
5. On the **General** tab, complete the following substeps:
    a. In the **ID** column, set the radio button to the field that is the unique identifier for the Vendor row in the database.
    b. In the **Filter** column, set the radio button to the **PartitionID** field.
    c. In the **Search** column, only select those field values that are strong and unique criteria for selecting the invoice.
6. On the **File import** tab, complete the following substeps:

a. Reimport the pool.

b. Set the class name settings to `[Name]_[Index]`.

c. Configure the following field settings, set the first line to the unique identifier for the record in the vendor extract. It is recommended that the field uses the following structure, but this is optional depending on what is appropriate for the client as long as the first line is the unique identifier.

```
[Index]

[Name]

[Address1]

[City]

[State][Zip]
```

Result The vendor field configuration is complete, and a green light with the message **Engine Is Ready** must appear in the field status box.

7. Save and close the file.

6.4.1.8 Configure the BRWSRC Table

The BRWSRC table informs the system which column in the vendor master or employee master pool corresponds to the internal field that is used in processing. This is a global table and must only have a single row. To configure the tables, complete the following steps:

1. Map the parameters in the SRC table to the column name of the database column within the master table or the CSV file column header for the respective ASE pool.
2. If the BRWVendorMaster table is being used as a source for the vendor master table, then configure the `SRC` parameters as listed in the table.
3. Columns **Custom1** to **Custom5** are available for a developer to map additional fields that may be available within a source of vendor master data to standard AP Packaged Project data structures. These items are available via the `Custom1` to `Custom5` properties of the Address structure used in numerous user exit interfaces.

Column in BRWSRC table	Value
ID	IndexID
SiteID	SiteID
Name	Name
Address1	Address1
Address2	Address2
City	City
Zip	Zip
State	State
Country	Country
POBox	POBox
POBoxZip	POBoxZip
EUMember	EUMember

Column in BRWSRC table	Value
Currency	Currency
TaxID1	TaxID1
TaxID2	TaxID2
VATRegNo	VATRegNo
TaxJurCode	TaxJurCode
TelNo	TelNo
InvoiceType	InvoiceType
PaymentMethods	PaymentMethods
BankDetails	BankDetails
WithholdingTaxDetails	WithholdingTaxDetails
CompanyCodes	CompanyCodes
UtilityFlag	UtilityFlag
PORSubscriberNo	PORSubscriberNo
ExternalVendorID	ExternalVendorID
ExternalSiteID	ExternalSiteID
VendorAccountGroup	VendorAccountGroup
AlternatePayee	AlternatePayee
PermittedPayee	PermittedPayee
SiretID	SiretID
VendorIdentifier	VendorIdentifier
PartitionID	PartitionID
EUMemberAlias	X
Custom1 to Custom 5	Use these fields to store additional information about the vendor that is specific to your needs.

4. Save the changes.

7 INI File Configuration Settings

The <project>.ini file resides in the same directory as the project SDP file. The configuration contains settings which controls the way in which the project file behaves. The following sections describe the configuration settings available in the <project>.ini file.

7.1 INI File Nomenclature

The <project>.ini file is subdivided into sections that control different aspects of the project file behavior. Each INI file parameter is made up of the following nomenclature:

```
XXX_YY_DDDDD=ZZZ
```

Or

```
XXX_YY_NN_DDDDD=ZZZ
```

Where the nomenclature is the following:

- XXX is the INI file section ID, such as REP, GRL, ITY, EXP, and so on.
- YY is the type of setting where VL denotes a value or list of values. OP denotes an on/off switch and must be set either to YES or NO.
- NN is an optional INI file group ID used to tie multiple individual settings together to form a settings group. This is similar to a database table where XXX is table name, NN represents the unique table row and DDDDD represents the unique table column name.
- DDDDD is the parameter name, which must be at least 5 characters.
- ZZZ is the parameter setting, which can be completed by the individual configuring the project and can be more or less than 3 characters. Only ZZZ values must ever be changed in the file, though additional NN settings groups may also be added as appropriate.

7.2 Field Configuration

The extraction and validation of fields is controlled in the WebCenter Forms Recognition database in the BRWFLD table.

Extraction and validation rules are set at the profile ID level and assigned to clients.

Each row in the database represents a field, and the table is keyed by the profile ID, the document class name and the technical name of the field. During installation, the table is populated with a full list of the fields available within the project for Client 0 (zero).

The entries in BRWFLD table allow you to do the following:

- Switch fields on and off
- Set fields to mandatory or optional
- Set default field values
- Set different field rules depending on the document classification result
- Set a field type, such as date, amount, table, text, and corresponding validation rules
- Allocate usage of custom fields

Note: The name of the field is displayed in the **FieldName** column. This name must not be changed.

7.2.1 Switch Fields On and Off

Fields that are not active do not appear on the Dynamic Verifier form. If a standard field in the project is not listed in the BRWFLD database table for a profile, it is considered inactive. To turn a field on or off, complete the following steps:

1. Open the BRWFLD database table.
2. To turn a field on, set the **Active** column to `TRUE`.
3. To turn a field off, set the **Active** column to `FALSE`.
4. Save the changes.

7.2.2 Mandatory or Optional Fields

Whether a field is mandatory or optional is controlled using the **RequiredInRTS** and **RequiredInVerifier** columns in the BRWFLD database table. The following describes the effect of setting these columns to `TRUE` and `FALSE` in isolation and in tandem:

RequiredInRTS	RequiredInVerifier	Effect
FALSE	FALSE	The population of the field is optional within the project.
TRUE	FALSE	If the system does not extract a value into this field automatically, the field is marked invalid and the document is sent to Verifier. The user is permitted to leave a blank value in Verifier.
TRUE	TRUE	If the system does not extract a value into this field automatically, the field is marked invalid and the document is sent to Verifier. The user must enter a value in Verifier.
FALSE	TRUE	If the system does not extract a value into this field automatically, the field is marked invalid and the document is sent to Verifier. The user must enter a value in Verifier.

To make a field mandatory or optional, complete the following steps:

1. Open the **BRWFLD** table.
2. To make a field mandatory, set the **RequiredInRTS** and **RequiredInVerifier** columns to `TRUE`.
3. To make a field optional, set the **RequiredInRTS** and **RequiredInVerifier** columns to `FALSE`.
4. Save the changes.

7.2.3 Apply a Country Filter

Some fields required by the solution are mandatory or optional depending on the country to which the invoice relates. For example, the payment reference field needs to stop in Verifier if nothing is extracted in case the document is a Danish, Swedish, Finnish or Norwegian invoice, but not in case it is from any other country.

The country filter permits a country-specific dimension as to whether a field is needed or not.

Within the table BRWFLD, the column **CountryFilter** can be populated with a comma-separated list of country ISO-codes. Examples of ISO-codes for the four Nordic countries are DK, SE, FI, NO.

This means that, if a field is configured as mandatory, and the extracted vendor's country of origin is not specified in the list of countries, the field is permitted to pass as a blank value.

7.2.4 Force a Field to Require Verification

To configure a field so it is always marked invalid and then reviewed by a user in Verifier, complete the following steps:

1. Open the **BRWFLD** table.
2. Set the **ForceVerify** column to `TRUE`.
3. Save the changes.

7.2.5 Label a Field in Dynamic Verifier

To control how a field is labeled in Dynamic Verifier, complete the following steps:

1. Open the **BRWFLD** table.
2. Note the text ID that has been assigned to the field in the **VerifierTextID** column.
3. Open the table BRWFLDTexts and find the row or rows that match the text ID in the above step for the appropriate display language.
4. Change the content of the **Message** column in BRWFLDTexts to reflect the desired field label. If you need to add a new language for an existing label, you can add a new row into BRWFLDTexts where the **TextID** column matches the existing labels for that field and the language column is populated with the appropriate language ISO-Code.
5. If no text ID is assigned in BRWFLDTexts or the BRWFLDTexts table has no content for the Verifier user language, the system uses the text in the VerifierLabel column in the table BRWFLD as a default. This can also be changed, if required.
6. Save the changes.

7.2.6 Set Field Default Values

There are two default settings for every field, and the field usage depends upon how the default is applied. To set the default values for a field, complete the following steps:

1. If a field is always set to a fixed value irrespective of extraction, set **the DefaultValue** column to the desired field default value.
   or
   If a field must have a default value because the system has not extracted anything into that field, set the **DefaultIfNothingExtr** column to the desired field value.
2. Save the changes.

Field defaults apply irrespective of whether the field is active or not.

7.2.7 Set Field Types

The field type governs which of the additional settings in the table affect the validation of the field.

The following are the four field types you can assign to each field:

- Date
- Amount
- Text
- Table

The field type is assigned by populating the **FieldType** column for the field in table BRWFLD. The content of the field is not case-sensitive. The different types, along with their respective configuration options, are described in the following sections.

7.2.7.1 Configure Date Fields

For a DATE type field, any value extracted must be in a legitimate date format in the Gregorian calendar. If it is not, and the system is unable to convert it, then the field is marked as invalid and the document is sent to Verifier.

To configure date fields, complete the following steps:

1. Open the **BRWFLD** table.

   In the **FutureDays** column, enter a numerical value that indicates the number of days in the future from the present date that an extracted date is considered valid. For example, if today's date is March 20 and a date is extracted as March 31, and the value is set to 10, then the system marks the field invalid as the extracted date is 11 days in the future. If future dates are not permitted, then this must be set to 0 (zero). To disable the check entirely, the column value must be set to -1. You can pass any value in Verifier, as long as it is a valid date.

   In the **NoDaysInPast** column, enter a numerical value that indicates the number of days in the past from the present date that an extracted date is considered valid. For example, if today's date is March 20 and a date is extracted as March 9, and the value is set to 10, then the system marks the field invalid as the extracted date is 11 days in the past. If past dates are not permitted, then this must be set to 0 (zero). To disable the check entirely, the column value must be set to -1. You can pass any value in Verifier, as long as it is a valid date.

2. To force an extracted date to stop in Verifier if the displayed date is not in the current month, set the **DateOnlyInCurrentMonth** column to TRUE.
3. To display the date as MM/DD/YYYY in Verifier, enter MMDDYYYY in the **VerifierFormat** column in table **BRWDAT**. To display the date as DD/MM/YYYY, enter DDMMYYYY. To display the date as YYYY-MM-DD, enter YYYYMMDD.

   Note: The date output format settings are set for each profile ID, so a different configuration is permitted for each client.

4. The date output format for export is controlled using ExportFormat in table BRWDAT. Entries in this column can be MMDDYYYY, DDMMYYYY, or YYYYMMDD. An optional separator is configurable using the ExportSeparator column, for example '/' or '-'.
5. Save the changes.

7.2.7.2 Configure Amount Fields

The following settings apply if the field type is set to AMOUNT. These settings take effect only during data export.

To configure amount fields, complete the following steps:

1. Open the **BRWFLD** table.
2. Set the **Amount** field to Amount.
3. To control the number of decimal places assigned to the amount during data export, in the **DecimalPlaces** column, enter a numeric value.
4. To set the export output format of an extracted value that is less than zero, in the **NegativeType** column, enter one of the following integers (1-3):
   a. 1 - this forces a minus sign to appear after an amount, such as 100.00-.

b. 2 - this forces a minus sign to appear before an amount, such as -100.00.
c. 3 - this forces a value to appear in parentheses, such as (100.00).

5. If an alternative value must be exported if the extracted value is either empty or zero, enter the alternate value in the **OutputForZero** column.
6. If an alternative value must be exported if the extracted value is greater than zero, enter the alternate value in the **SubstituteValueIfOver0** column.

7.2.7.3 Configure Field Calculations

In the following extraction, the standard amount fields are validated mathematically to check that the invoice is in balance. The following calculation is applied if line items are required for the document:

```
AmounTotal = Sum of LineItems + ( AmountTax + PST + HST ) + (
AmountFreightPrepaidAndAdded + AmountMisc ) – AmountDisc – (
AmountWithholdingTax + ISRRetention )
```

If no line items are required, the following calculation is applied:

```
AmounTotal = AmountSubtotal + ( AmountTax + PST + HST ) + (
AmountFreightPrepaidAndAdded + AmountMisc ) – AmountDisc – (
AmountWithholdingTax + ISRRetention )
```

If the VAT table is activated, the following additional check is carried out:

```
AmounTotal = ( Sum of tax amount column + sum of taxable amount column ) –
( AmountWithholdingTax + ISRRetention )
```

To set the initial validity of the field, this calculation is always applied on the server side, but it can be switched off in Verifier.

To switch off the mathematical validation in Verifier, complete the following steps:

1. Open the **BRWAMT** table.
2. Set the **DeactivateCrossValidation** column to `TRUE`.
3. Save the changes.

Note We recommend setting this value to `FALSE` if all fields included in the calculation above are not in scope for the client. Otherwise, a Verifier user is unable to process a document because information needs to be entered or corrected on a field that does not exist on the form.

7.2.7.4 Format Separators in Amounts for Output

To format the separators that appear in amounts during export, complete the following steps:

1. Open the **BRWAMT** table.
2. Set the **ExportThousandSeparator** and **ExportDecimalSeparator** fields with the parameters that are appropriate for your business needs. For example, the following table displays the effect upon the output of the amount 10,000 if the number of decimal places is set to 2.

ExportThousandSeparator	ExportDecimalSeparator	Output for a value of TEN THOUSAND to 2 decimals
,	.	10,000.00
.	,	10.000,00

ExportThousandSeparator	ExportDecimalSeparator	Output for a value of TEN THOUSAND to 2 decimals
(blank)	.	10000.00
(blank)	(blank)	10000.00
‘	.	10’000.00

7.2.7.5 ConfigureText Fields

Text fields are fields that can contain numeric and alphanumeric characters. Examples of text fields within the project include the invoice number, the line item description, and the vendor ID.

To configure text fields, complete the following steps:

1. Open the **BRWFLD** table.
2. In the **MinLength** column, enter a numeric value that represents the minimum permitted length of the field. Any extracted value or user input below this length is not permitted.
3. In the **MaxLength** column, enter a numeric value that represents the maximum permitted length of the field. Any extracted value or user input that exceeds this length is not permitted.
4. In the **PadChar** column, enter a numeric value by which a field is padded to the right when a maximum field length is not met. For example, if a value `1234` is extracted and the maximum field length is set to `10`, the padded value is displayed as `123400000`.
5. To have a value padded by the character entered on the left, such as `000001234`, set the **RightJustify** column to `TRUE`. If the **RightJustify** column is set to `FALSE`, the padded value displayed is `1234000000`.
6. To remove all special characters from the extracted or user-entered value, set the **RemoveAllSpecials** column to `TRUE`.
7. To remove all spaces from a value and retain a non-comma separated list of special characters, set the **RemoveBlanks** column to `TRUE`.
8. To remove any special characters at the beginning and at the end of an extracted or user-entered value, set the **KeepCertainSpecials** column to `TRUE`.
9. To define the starting character when trimming an extracted value, set the relevant starting character in the **SubstringStartPos** column and the length of the substring in the **SubstringLength** column. Positive numbers start from the left side of a field and negative numbers start from the right side. For example, the invoice number must only be 5 characters, but the extracted value is `123456789`. `SubstringStartPostion` of `1` and `SubstringLength` of 5 equals to `12345`. `SubstringStartPostion` of -5 and `SubstringLength` of 5 equals to `56789`.
10. To remove any leading zeros from an extracted or user-entered value, set the **RemoveLeadingZeros** column to `TRUE`.
11. To enable a field to have a comma-separated list of valid entries that contains only approved characters for an extracted or user entered value, enter any letter, number or wildcard character in the **FieldMask** column. For example, if the content of this column is set to `ABCD,WXYZ`, then no value is permitted in this field unless it is equal to either `ABCD` or `WXYZ`. In case of wildcard characters, a *hash* symbol (#) is used to represent any number, an *at* symbol (@) is used to represent any letter, and a *question mark* (?) is used to represent either a number or a letter.

The following is a list of special fields that have additional configurable validations and extraction influences on top of the regular configuration specified in table BRWFLD:

Fieldname	Additional validations
PONumber	Optional validation against an ODBC datasource or stored procedure is configured in table BRWPON. Format settings for a valid PO are configured in table BRWPONFormats. In order to extract a purchase order number field, profile ID entries must be present in this table.
InvoiceNumber	Optional validation against a history of invoice numbers is provided by the same vendor. This is configured in table BRWNUM.
BillToName	Format settings for a valid bill-to name are entered in table BRWBTOFormats. In order to extract a bill-to name, profile ID entries must be present in this table.
IBAN	Checksum validations are applied.
PaymentReference	Checksum validations are applied for any extracted value if the vendor is from Denmark, Sweden, Finland, or Norway.
VendorID	Validation against vendor master data pool can be activated in table BRWVND.
Currency	Validation against an ODBC data source can be activated in table BRWCUR. Currencies for extraction are set up globally for the project (independent of the profile) in table BRWCurrency.
CompanyCode	Validation against an ODBC datasource can be activated in the table BRWCCO.
AmountTax / VAT Rate (in the line items table)	In the table BRWTAXCONFIG, a comma-separated list of valid tax rates can be entered against the `PrimaryRates` and `SecondaryRates` columns.. In order to extract a VAT rate in the line items table, the rate must be listed in either of these columns. The primary rates column is intended for the headline rates of VAT in the countries within the scope for the profile ID. The secondary rate column is intended for the reduced rate of VAT. It is not necessary to include zero in the list of valid rates. We recommend maintaining this list of values for the benefit of extraction of the invoice tax amount.
Alternate Payee	This field operates in conjunction with the Vendor ID field. In the table BRWVND, if the parameter `CheckForAlternatePayees` is set to `TRUE`, then the system tries and identifies the party receiving payment from the invoice as well as the regular invoice vendor. Cross-validation between the vendor ID and alternate payee is also activated so that a payee cannot be selected unless it has been assigned to the main vendor ID.

7.2.7.6 Use Table Fields

AP Packaged Project has the following standard table fields:

- LineItems
- VATTable
- DeliveryNotes

The field type is shown as `TABLE` for each of these fields, and it must be activated for the table to be relevant for a specific profile.

For the LineItems table, each of the individual column names are registered as separate rows in the BRWFLD table so they can be switched on and off, relabelled, and configured like a regular field.

The Delivery Note table, which has a single column, is validated according to the settings in the standard DeliveryNote field configuration in table BRWFLD.

The VAT table has three columns that are formatted and handled as amounts, which cannot be changed.

7.2.7.7 Use Document Classes for Special Field Rules

You can configure the system to use alternative field rules depending on the result of the document classification as it stands prior to extraction. For example, if a custom child class of the main `Invoices` class, called `SpecialVATInvoice` is created and this class requires a certain length and formatting rules for the Vendor and Bill-to VAT registration number fields, then two additional rows are added into the BRWFLD table for `VendorVATRegNo` and `BillToVATRegNo`, but with the class name column set to `SpecialVATInvoice`. The class name column is not case-sensitive.

Thus, if a document is classified to `SpecialVATInvoice`, the field rules are lifted from the BRWFLD entries for that class name. For other invoices, or for other fields that exist on the `SpecialVATInvoice` class that do not have special class-specific entries, the system continues to use the BRWFLD entry where the class name is set to `INVOICES`, which is the default setting.

Hence, the logic for loading the settings for each individual field is that the system checks for class-specific rules using the document class name in the first instance, then, if none exist, the field settings for the `INVOICES` class are used.

7.2.7.8 Work with Custom Fields

The following custom fields are included in the `<project>.sdp` file and can be set up as per your business needs:

- Custom1
- Custom2
- Custom3
- Custom4
- Custom5

This means that for each client, five additional fields are available on top of the standard fields. You can activate and configure these as either a date, an amount, or a text value field on a profile-by-profile basis. These fields cannot be configured as a table type field. Automatic extraction can be configured for these fields by assigning an extraction profile via the **ExtractionProfile** column in table BRWFLD. However, any additional configuration within the project file, such as field training or scripting would apply globally across the project for all clients.

7.2.7.9 User Exit For Text Fields

To permit specific formatting and validation coding that cannot be achieved using the regular configuration options, configure the `UserExitTextFieldFormatting` parameter, which is available in the project user exit events. For more information, contact your administrator.

7.2.8 Use Substitution Rules

Within the BRWFLD table, a text field can be assigned a substitution rule, which permits an extracted text value to be substituted in part, or as a whole, with another value. This works in a similar way to the standard VB replace command. One substitution rule can be assigned per field. These settings take effect only during data export so the effects of a substitution is not seen within the Verifier application.

Substitution rules are held in global the BRWSubstitution table, which has the following structure:

Fieldname	Explanation of usage
Index	Substitution rule index.
Original	String value to be replaced.
Replace	String value to be substituted.

Note: The values contained in the Original and Replace columns of the BRWSubstitution table are case-sensitive.

8 Dynamic Verifier Forms

A Dynamic Verifier form is a window presented to users in the Verifier application that contains fields that are active and that can be updated by users.

A Dynamic Verifier form is activated in the `<project>.ini` file. When a Dynamic Verifier form is activated, all documents imported into WebCenter Forms Recognition use the Dynamic Verifier form from that point forward.

8.1 Activate a Dynamic Verifier Form

To activate a Dynamic Verifier form, complete the following steps:

1. Open the `<project>.ini` file.
2. Set the `GRL_OP_UseDynamicVerifierForm` parameter to `Yes`.
3. Save the changes and close the INI file.
4. Restart the Verifier application on each client machine.

8.2 Turn Off Dynamic Verifier Logging

Diagnostic logging for Dynamic Verifier is on by default and exports logging information to the WebCenter Forms Recognition `V_ log` file. To turn the logging off, complete the following steps:

1. Open the `<project>.ini` file.
2. Set the `GRL_OP_DynamicDebug` parameter to `No`.
3. Save the changes and close the file.

8.3 Change the Dynamic Verifier Field Order

To change the order of the fields as they appear on the Verifier a user exit is provided.

The standard order is as follows:

1. InvalidReason
2. VendorID (which includes the address display, search button and the equivalent items for the alternate payee field)
3. DocumentType
4. InvoiceType
5. POType
6. CompanyCode
7. InvoiceNumber
8. InvoiceDate
9. DueDate
10. YourRef
11. DeliveryDate
12. DeliveryNote
13. DeliveryNotes (table)
14. AccountNumber
15. PONumber
16. PONumbers (table) - not available
17. PORNumber
18. PORSubscriberNo
19. BillToName

20. VendorVATRegNo
21. BillToVATRegNo
22. EmployeeID (which includes the address display and search button)
23. PaymentReference
24. BankAccount
25. BankAccountCode
26. IBAN
27. BIC
28. AmountSubtotal
29. AmountFreightPrepaidAndAdded
30. AmountMisc
31. AmountDiscount
32. AmountTax
33. HST
34. PST
35. ICMS
36. AmountWithholdingTax
37. ISRRetention
38. VATTable (table)
39. AmountTotal
40. Currency
41. ExchangeRate
42. LocalVATAmount
43. Custom1
44. Custom2
45. Custom3
46. Custom4
47. Custom5
48. InvoiceCodeCN
49. PasswordCN
50. MexicanUUID
51. LineItems (table)

This order may be changed by adding code to the `UserExitDynamicVerifierFields` parameter. Passed into that function is the 1-based vFields array, the content which can be edited to reflect the desired field order.

For example, the enduser requires that the invoice number and the invoice date appear in reverse order. As per the list above, the invoice number has an index of 7 within the array as the 7th field in the list; the invoice date has an index of 8. To reverse the order, the fields need to be switched around. This can be done using the following script:

```
Public Sub UserExitDynamicVerifierFields(vFields() As String)
vFields(7) = "InvoiceDate"
vFields(8) = "InvoiceNumber"
End Sub
```

8.4 Dynamic Verifier Form Language Translations

AP Packaged Project contains a dynamic language translation feature for the Verifier application. If the project uses the dynamic language form, you can select the language you want. When the user logs in, the system translates the Verifier user interface to the language selected. This

translation includes all field and column labels, dialog box and message boxes, search dialogs, drop-down options for the document type, invoice type, Purchase Order type, invalid reason fields, and standard system error messages. The Verifier language preference is set in the **LanguageID** column in the BRWUser table. If no language preference is set, the system defaults to English. AP Packaged Project includes two language packs for English and Chinese, but additional translations can be configured within the AP Packaged Project database.

The following table lists each type of text element and the corresponding database table in which they are managed:

Text element	Table
Field and Column labels	BRWFLDTexts. The field and column labels defined in the BRWFLDTexts table are linked to their corresponding fields in BRWFLD through the VerifierTextID column. If VerifierTextID is populated within the BRWFLD table and a populated entry exists for the Verifier language preference in the BRWFLDTexts table, then this over-rides any other label defined in the VerifierLabel column.
Standard system error messages	BRWERR. If no error message is defined for the Verifier language preference, the system by default displays the corresponding message in English. If the message is not defined, a generic default error message appears. Each error message is presented to the user suffixed by an error code, for example, CN201. This code is present to facilitate easy cross-referencing to the standard system error message. Standard error messages written to the system export log are in English.
Invalid reason drop down	BRWIVRTexts. Each entry in the BRWIVRTexts table must correspond to an entry in the BRWIVRType table. The entry in the column TextID in BRWIVRTexts table must be the same as the entry in the column IndexID in the BRWIVRType table. If no invalid reason text is defined in BRWIVRTexts table for the Verifier language preference, the default text defined for the corresponding entry in the BRWIVRType table is used.
Document type drop down	BRWDTYTexts. The valid values for the TextID column are table 1 or 2, where 1 indicates an invoice and 2 indicates a credit memo. The system rejects any other entries.
Invoice type drop down	BRWITYTexts. The valid values for the column TextID is table 1 or 2, where 1 indicates an invoice type of PO, and 2 indicates an invoice type of NO-PO. The system rejects any other entries.
PO Type drop down	BRWPTYTexts. The valid values for the column TextID is table 1 or 2, where 1 indicates PO type of MATERIAL and 2 indicates a PO type of SERVICE. The system rejects any other entries.
Class names, search dialog labels and information dialog labels	BRWTexts. If no entry exists for the Verifier language preference, the system uses English as the default entry.
Information box texts	BRWINFType. If no entry exists for the Verifier language preference, the system uses English as the default entry.

9 Set Up Users

The AP Packaged Project database contains the BRWUser user table. Use this table to configure the following parameters for your users:

- The AP Packaged Project permissions for each user.
- The log in authentication for a user using Windows authentication, or supply a static user name and password.
- The client groups a user can access and the corresponding documents a user can process.
- The users that are subject to quality reviews.
- The Verifier user language preference for the dynamic Verifier form.

After you configure the user table, you configure a server job to automatically import the user table into AP Packaged Project.

9.1 Populate the User Table

The BRWUser table is keyed by a unique combination of users' user names and the client groups that they are assigned to. If a user is assigned to multiple client groups, then multiple rows need to be added into the table. For example, if user JSMITH needs to be assigned to client groups 1 and 2, the UserID column has two entries in the BRWUser table.

To populate the user table, complete the following steps:

1. Open the **BRWUser** table.
2. In the **UserID** column, enter a user name.
3. In the **ClientGroup** column, enter the client group to which the user belongs.
4. To have the user log in with a user name and password, enter a password in the **Password** column. If you want the user to log in using Windows Authentication, leave the column blank.
5. If the user is using Windows Authentication to log in, in the **Domain** column, enter the Windows domain information.
6. To configure whether verification work performed by the user is subject to quality control review, in the **RequireReview** column, enter **Yes**.
7. In the **PrimaryGroupName** column, enter the group name for the group the user is assigned.
8. The AuthorityLevel column controls what a user is permitted to do within the Thick Verifier application. To configure the Thick Verifier activities for a user, enter one of the entries listed in the following table in the **AuthorityLevel** column. All users automatically inherit the FLT permission.

AuthorityLevel value	Description
ADM	This is the administrator role that manages users, groups, and user-to-group assignments. Administrators install AP Packaged Project, configure applications, and manage data. They also design and maintain projects. This role is the most powerful of the roles, because it encompasses the permissions for all other authority objects. For a user that is granted the ADM role, the client group may be left blank. If it is left blank, the administrator is able to see documents in Verifier belonging to all clients.
SLM	This is the Supervised Learning Manager role that is used to define, modify, and maintain the Learnset. This functionality is accessible only through Verifier.
SLV	This is the Supervised Learning Verifier role used to collect and manage local training data. Supervised Learning Verifiers are subject-matter experts who train Learnset candidates to improve system performance. This functionality is accessible only through Verifier.

AuthorityLevel value	Description
VER	This is the Verifier role that verifies documents that could not be automatically processed. Typically, members of the Verifier group are clerks. This functionality is accessible only through Verifier.
SET	This is the Verifier Settings role used to change the AP Packaged Project for Verifier configuration. This role is given to users who are considered to have enough knowledge of the application to make changes that are beneficial to all Verifier users.

9. Type the Verifier user language preference into the **LanguageID** column. Enter the two character, standard ISO-code. For example, enter `EN` for English and `CN` for Chinese.
10. Save the changes.

9.1.1 Configure an Automatic Import Job

After the BRWUser table is populated, an automatic import job is configured to import users into the main WebCenter Forms Recognition database. The automatic import job is configured in the RTS Management Console against the RTS instance that is carrying out document import.

To configure an automatic import job, complete the following steps:

1. Open the **RTS Management Console**.
2. On the **General** tab, complete the following steps:
    a. Configure the frequency of the system security updates using the drop down options provided. For example, every 2 days, or every 30 minutes.
    b. Configure the user update starting date and time.
    c. Ensure that the **Update system security** check box is selected.
3. On the **General** tab, click **OK**.

9.1.1.1 Automatic Import Job Errors

An error is displayed and the import fails if any of the following conditions are met. Error messages are located in the WebCenter Forms Recognition log file for the RTS instance performing the user import job.

- You do not supply a connection string for the WebCenter Forms Recognition database referenced by the SQL connection group in the GRL section of the `<project>.ini` file.
- The system is unable to connect to the WebCenter Forms Recognition database.
- The BRWUser table is empty.
- A user name column is blank.
- The Client Group column is blank and the user is not an administrator.
- The Client Group column does not contain a numeric value.
- The primary Group ID column is not populated.
- You have not allocated clients to the Client group in the BRWClient table.
- The user the system is trying to add already exists as a user that was created through the Designer application – the Designer-created user must be deleted.

9.2 Solution Features

The following section details the features available within the AP Packaged Project solution.

9.2.1 Document Management System (DMS) Integration

AP Packaged Project supports integration with document management systems in both the early and late archiving scenarios.

Early Archiving. Early archiving means that the image has already been archived prior to reaching AP Packaged Project. In this scenario, AP Packaged Project requires a copy of the archived image with the unique archive document ID embedded into the document file name.

Configuration options in the IMP section define whether this unique archive document ID constitutes the entire file name or a component of that file name.

At time of document export, the archive document ID is passed downstream via the AP Packaged Project URN field.

Late Archiving. Late archiving means that the image is to be archived after processing in AP Packaged Project.

The standard CSV file output can be configured to produce an import file compliant with Oracle ECM and Mobius. The standard late archiving options are available for the Reporting Database.

9.2.2 ERP System Integration

Integration to downstream ERP systems with AP Packaged Project is possible using the following interfaces:

- Flat file transfer
- Export to database staging tables

The various export options can be activated in the BRWEXP table.

9.2.3 Solution Reporting

AP Packaged Project contains out-of-the-box connectors to populate the reporting tables for solution reporting.

9.2.4 Automatic General Ledger Account Coding

AP Packaged Project provides a feature by which non-purchase order related invoices are coded automatically for general ledger entry and the appropriate cost object assigned.

Both the general ledger coding and cost object determination are carried out using the unique search technologies of AP Packaged Project, which has the ability to reconcile the free-text information on an invoice to structured data.

By virtue of the extracted line item description, the search technology is used to allot an appropriate general ledger code based on the code that has been used for that item in the past. The cost object information is derived through contact names, department names, and ship-to addresses that the vendor may have included on the invoice document.

These two items together provide a complete set of coding strings for the invoice in order that they can be submitted to the downstream ERP system.

Currently, the implementation of the auto GL-coding feature requires customization, typically within UserExitPostLinePairing, but this is converted into a more comprehensive set of configuration options in a future release of the AP Packaged Project product.

10 Configuration Settings

The configuration settings control the way in which the project files behave.

The project files can be configured using the `<project>.ini` file that resides in the same directory as the `project.sdp` file and the AP Packaged Project database.

10.1 INI File Settings

The `<project>.ini` is subdivided into sections that control different aspects of the project file behavior.

Each `<project>.ini` file setting parameter is comprised of the following nomenclature:

```
XXX_YY_DDDDD=ZZZ or XXX_YY_NN_DDDDD=ZZZ
```

- `ZZZ` is the parameter setting.
- `XXX` is the INI file section ID, such as REP, GRL, ITY, EXP.
- `YY` is the type of setting, and VL denotes a value or list of values. OP denotes an on/off switch and must be set either to YES or NO.
- `NN` is the optional INI file group ID used to tie multiple settings together to form a settings group.
- `DDDDD` is the parameter name.

Note: Only the parameter settings indicated as `ZZZ` in the above example and the type of setting indicated as `YY` in the above example must be changed in the file. Although, additional file group ID settings may be added when appropriate.

10.1.1 GRL Section

This section contains global settings for the project that are used for the purposes of solution reporting.

The first three parameters are set primarily from the point of view of reporting, and these are stored against all database records created from documents processed through the project.

You can set the parameters listed in the following table:

Parameter	Type	Description
ProjectName	Freetext	This is the name of the project.
Version	Number	This is the name version number of the project implementation at the client.
ClientName	Freetext	This is the name of the client.
VerifierFormStyle	Freetext	This is the name color scheme applied to the Verifier form. The option is BW. This is the WebCenter Forms Recognition logo color Scheme. If any other setting is applied (including blank), then the system displays the default Verifier color scheme, such as gray form with valid fields marked in green and invalid fields marked in red.
UseDynamicVerifier Form	YES/NO	This is the name flag to indicate whether the project must use the dynamic verifier form.
ReviewState	Freetext	This is the name RTS state that a document must be set to if it is subject to review.
ReadSettingsForDB	YES/NO	This is the flag to indicate whether the AP Packaged Project settings must be read from the database. This value must always be set to YES.

Parameter	Type	Description
SQLConnectionGroup	NN	This is the numeric reference to the SQL connection group which represents the database in which the DFI configuration tables have been created. This should be set to 01, 02, and so on.
DynamicDebug	YES/NO	If this is set to YES, logging for the internal mechanics for the layout of the dynamic verifier form is written into the standard Thick Verifier log file.
BatchInDatabase	YES/NO	This is the flag to indicate whether the batch containing the production documents exists within a database or within a batch root folder. This must always be set to YES.
BatchSQLConnection Group	NN	This is the numeric reference to the SQL connection group which represents the primary AP Packaged Project database. This must be set to 01, 02, and so on.
ActivateBatchSecurity	YES/NO	This setting is redundant.
ActivateVendor Filtering	YES/NO	This is the flag to specify whether the system utilizes multiple sets of vendor master data/partitions.
VendorFilter Column	Freetext	This is the case-sensitive name of the database column in the vendor master table that contains the partition ID.
ActivateEmployee Filtering	YES/NO	This is the flag to specify whether the system utilizes multiple sets of employee master data/partitions.
EmployeeFilter Column	Freetext	This is the case-sensitive name of the database column in the employee master table that contains the partition ID.
BufferClientSettings	YES/NO	If set to YES, the system buffers the configuration settings for the current client in order to reduce repeated calls to the database. For example, if the system is processing a document for client 1, then following an initial read of the database, those settings are stored in memory. If the next document is also for client 1, then the system pulls the settings from memory, rather than reading the database. Only settings for the current client are held in memory. This means that any changes made to the database may not take effect instantly for all clients, hence a restart of the RTS would be required. If this is not desirable, this option must be set to NO.

10.1.2 IMP Section

This section contains settings for document import, specifically the mapping of values contained within the image file name to fields in AP Packaged Project. This provides a simple means to pass data to AP Packaged Project from an upstream system.

Filename components must be separated by a configurable separator, for example, an underscore such as `COMPONENT1.tif`, `COMPONENT1_COMPONENT2.tif`, `COMPONENT1_COMPONENT2_COMPONENT3.tif`.

For example, `IMP_VL_ScanDate=COMPONENT1` for the file `12022008_1234_123456.tif` would put `12022008` into the ScanDate field in AP Packaged Project.

The table also includes settings for importing a document client ID from a database.

You can set the parameters listed in the following table:

Parameter	Type	Description
URN	Freetext	This is the document unique reference number.
BatchName	Freetext	This is the document batch name.
ScanDate	Freetext	This is the document scan date.
PriorityFlag	Freetext	This is the document priority flag.
InvoiceType	Freetext	This is the document invoice type.
DestinationArchive	Freetext	This is the document destination archive.
CompanyCode	Freetext	This is the document company code.
InputSource	Freetext	This is the document input source, such as SCAN, EDI, and EMAIL.
ClientID	Freetext	This is the document client ID. In a multiclient project, this value must be mapped. If not, the configuration set associated with client zero is used.
LocationID	Freetext	This is the document Location ID. This field can be used to hold the BPO operation location ID that is relevant for the document, for example, the ID of a shared service center. The value contained within the file name is written into the reporting tables for that particular document to enable location level reporting.
ExternalBatchID	Freetext	This is the external batch ID.
TransactionID	Freetext	This is the transaction ID.
TransactionType	Freetext	This is the transaction Type.
Separator	Freetext	This is the component file name separator. If this is left blank, the separator defaults to an underscore.
PriorityFlagYes	Freetext	This is the value that denotes a positive setting for the priority flag.
DateFormat	Freetext	This is the format of a date contained within the document file name. Options are DDMMYYYY, MMDDYYYY or YYYYMNDD.
ImportClientIDFromDB	YES/NO	This is the flag to denote whether the document client ID is imported from a database table based upon the document filename/URN.
SQLConnectionGroup	NN	This is the SQL connection group specifying the client ID look-up database connection string as set in the SQL section. If no connection group is specified, the system uses group 01.
ClientKey	Freetext	This contains the component that is used as the database table key for the record in the client ID lookup table. If left blank, the key is set to the image file name (minus the file extension). If just a component of the filename is required, then this value is populated with the URN, then the URN component of the file name is mapped in the IMP section.
DBTableName	Freetext	This is the name of the database table holding the relationship between the document file name/URN and the client ID.
DBURN	Freetext	This is the technical name of the column in the database table above that holds the document file name/URN.
DBClientID	Freetext	This is the technical name of the column in the database table above that holds the corresponding client ID.

10.1.3 REP Section

This section contains the configuration settings for WebCenter Forms Recognition reporting.

You can set the parameters listed in the following table:

Parameter	Type	Description
ConnectToReportingDB	YES/NO	This is the flag to set whether the project writes out reporting data or not.
SQLConnectionGroup	NN	This is the SQL connection group specifying the reporting database connection string as set in the SQL section. If no connection group is specified, the system uses group 01.
ReportingInDesigner	YES/NO	This is the flag to indicate whether documents processed or analyzed in the WebCenter Forms Recognition Designer Module must have the results written to the reporting database.
StartNewRecordFor ImportedDocument	YES/NO	If this is set to YES, WebCenter Forms Recognition will create a new reporting record for each document imported into Designer, removing any old ones for the same document key. If this is set to NO, WebCenter Forms Recognition only writes to the reporting database if an entry exists for the same document key. This can be used in the event that the reporting trail begins at the scan station.
ReportingDBDocument Table	Freetext	This is the name of the document header table in the reporting database.
ReportingDBFieldTable	Freetext	This is the name of the document field table in the reporting database.
ReportingDBHistory Table	Freetext	This is the name of the document history table in the reporting database.
ReportingDBImage Table	Freetext	This is the name of the document image table in the reporting database.
StoreImageInReporting Tables	YES/NO	This indicates whether the document image must be stored in a binary type field in the reporting database.
ReportingKey	Freetext	This contains the component to be used as the database table key for the document record. If left blank, the key is set to the image file name (minus the file extension). If just a component of the file name is required, then this value must be populated with URN, then the URN component of the file name must be mapped correctly in the IMP section.
ArchiveURL	Freetext	This contains the mask for the URL associated with a document link. XXXXX must denote the part of the URL which must be substituted with the unique document ID from the point of view of the archiving system to form a valid URL which retrieves the document.
StorageDirectory	Freetext	This is the path to the directory which is used as a repository to store images subsequent to document export.

10.1.4 SQL Section

This section contains the SQL connection strings that are used by AP Packaged Project.

The solution supports Oracle and Microsoft SQL Server databases. You can set the parameters listed in the following table:

Parameter	Type	Description
NN_ConnectionString	Freetext	This is the connection string for the SQL group NN.
NN_EncryptedPassword	Freetext	This is the encrypted password for the SQL group NN.

10.1.5 ASA Section

The following settings control the ASE pools used for the vendor, employee code, and company code lookups in AP Packaged Project.

If an ASE field is not required, it can be removed from the INI file as long as the ASE field is also disabled in the project file.

You can set the parameters listed in the following table:

Parameter	Type	Description
Class	Freetext	This is the name of the AP Packaged Project class on which the field was created.
Fieldname	Freetext	This is the technical name of the AP Packaged Project field.
AlphaNum	YES/NO	This indicates whether the key field for the pool record is alpha-numeric if set to YES. If set to NO, the field is assumed to be numeric. This must be set correctly in order to generate the pool correctly.
PoolRelative	YES/NO	This indicates whether the location of the pool directory is relative to the project file.
PoolPath	Freetext	This is the UNC path to the pool directory if it is not relative to the project file.
PoolDirectory	Freetext	This is the name of the pool directory.
PoolName	Freetext	This is the name of the pool.
FileRelative	YES/NO	This indicates whether the location of the pool import CSV file is relative to the project file.
ImportPathFilename	Freetext	This is the UNC path to the pool import CSV file if it is not relative to the project file.
ImportFilename	Freetext	This is the name of the pool CSV import file.
ImportODBCDSN	Freetext	This is the name of the user DSN for the ODBC pool import.
ImportODBCSelect	Freetext	This is the select statement used to create the pool.
ImportODBCUser	Freetext	This is the user ID for connecting to the database. This can be left blank and specified in the AP Packaged Project project file if security requires it.
ImportODBCPWD	Freetext	This is the user password to access the database. This can be left blank and specified in the AP Packaged Project project file if security requires it.
AutoImportOption	'FILE', "NONE' or 'ODBC'	This indicates the source from which the pool must be created via the WebCenter Forms Recognition Runtime Server (RTS). If set to NONE, the pool is not be updated automatically by RTS.
FirstPageOnly	YES/NO	This is the flag to indicate whether only the OCR text on the first page of the document must be used to determine the field result.
PageZoneALeft	0-100	This is Zone A left search %.
PageZoneAWidth	0-100	This is Zone A width search %.
PageZoneATop	0-100	This is Zone A top search %.
PageZoneAHeight	0-100	This is Zone A height search %.
PageZoneBLeft	0-100	This is Zone B left search %.
PageZoneBWidth	0-100	This is Zone B width search %.
PageZoneBTop	0-100	This is Zone B top search %.
PageZoneBHeight	0-100	This is Zone B height search %.

10.1.6 WFR Section

This section contains configurations for the project features specific to Oracle AP processing.

Parameter	Type	Description
FilterVendorsByCompany Code	YES/NO	If this option is set to **YES**, only vendors that belong to the invoice's company code will be considered as candidates and returned in the results of a vendor search in Verifier.
SetNegativeAmountsFor Credit	YES/NO	If this option is set to **YES,** the following fields will always be exported to the XML file as negative values when the Document Type is **CREDIT**: ▪ Invoice Total ▪ Line Item Quantity ▪ Line Item Total
RemovePONumberFor NoPO	YES/NO	If this option is set to **YES**, any value in the PO Number field will be removed during document validation if the Invoice Type is **NO-PO**.
PerformDuplicateInvoice Check	YES/NO	If this option is set to **YES**, a lookup will be performed against the configured data source to determine whether the Invoice Number value already exists (i.e. whether this is a duplicate invoice). This check is performed during document validation.

Parameter	Type	Description
CDIERPName	Freetext	The name of the ERP (or other) system that will be used in the information messages displayed as a result of the duplicate invoice check. The default value for this setting is **E-Business Suite**.
CDISQLConnectionGroup	NN	SQL connection group specifying the database connection string, which will be used to perform the duplicate invoice check. If no connection group is specified, the system will use group **01**.
CDIDBTableName	Freetext	The name of the database table or view containing invoice data that will be used to perform the duplicate invoice check. The default value for this setting is **XX_OFR_INVOICES_V**.
CDIDBInvoiceNumber	Freetext	The name of the field in the database table that contains the invoice number. The default value for this setting is **INVOICE_NUMBER**.
CDIDBSupplierID	Freetext	The name of the field in the database table that contains the supplier ID. The default value for this setting is **VENDOR_ID**.
CDIDBSupplierSite	Freetext	The name of the field in the database table that contains the supplier site ID. The default value for this setting is **VENDOR_SITE_ID**.
CDIDBCompanyCode	Freetext	The name of the field in the database table that contains the company code. The default value for this setting is **ORG_ID**.
CDIIgnoreBlankValues	YES/NO	If this option is set to **YES,** the duplicate invoice check will be performed using only the configured database fields where the invoice has an extracted (or entered) value. For example, if the **CDIDBCompanyCode** setting is configured, but the document does not have a Company Code value, the lookup will be performed with that field omitted from the query.
CDIAllowForceValidation	YES/NO	If this option is set to **YES,** a message box will be displayed in Verifier stating that this is a duplicate invoice number. The user will have the option to validate and release the document anyway. If this option is set to **NO** (default), the document cannot be released in Verifier while there is a duplicate invoice number.
EnableMetadataPassthrough	YES/NO	If this option is set to **YES,** the configured file name components will be written to the CSV and/or XML output file(s) during export.
NN_CMPFilename Component	Number	This setting identifies which component of the file name should be written to the corresponding field(s) in the CSV and/or XML files. An underscore character must be used to separate the components in the image file name. For example, assume an image file name of **AA_BB_CC_DD_EE.tif**. The value of component 4 is **DD**.
NN_CMPCSVFieldID	Freetext	The unique identifier used in the *BRWCSV section* to determine where the component value will be written in the CSV output file. Oracle recommends that custom identifiers should begin with a **Z**, e.g. **<%ZAA>**. This setting may be left blank if it is not required for the component value to be written to the CSV file (i.e. the value is only required to be written to the XML file).
NN_CMPXMLHeader FieldName	Freetext	The tag that will be used to contain the component value in the header section of the XML output file.

Parameter	Type	Description
		For example, if you want the value to appear in the XML file as **<myComponent4>DD</myComponent4>,** the value for this setting would be **myComponent4**. This setting may be left blank if it is not required that the component value be written to the XML file (i.e. the value is only required to be written to the CSV file).
SPDEnableSeparator Detection	YES/NO	If set to **YES,** the Separator Page Detection feature will be enabled.
SPDRequireAllPhrases	YES/NO	If set to **YES**, all of the configured separator phrases configured in NN_SPDSeparatorPhrase must appear on the same page in the document for that page to be considered as a separator page. If set to **NO,** then the first page of the document that contains any of the configured separator phrases will be considered to be a separator page.

SPDDeleteSeparatorPage	YES/NO	If set to **YES,** the separator page will be removed from the document in Forms Recognition.
		
		To ensure that this modified document (with the separator page removed) is exported instead of the original image (that still contains the separator page), the following additional settings must be specified in the *BRWEXP section* and *BRWCSV section*:
		
		`EXP_OP_OutputTiffFile=YES`
		`EXP_VL_TiffName=XXX`
		`EXP_VL_TiffDPI=300`
		`EXP_VL_TiffFormat=G4FAX`
		`EXP_OP_RedactInvoiceNumber=NO`
		
		`CSV_OP_01_OutputImage=NO`

SPDSeparatorPhrase	NN	A phrase that signifies a separator page. Separator phrases should not include any of project's configured word segmentation characters, and other special characters such as asterisks should be avoided. Separator phrases should be a text string that is not likely to legitimately appear in an invoice document, for example, **XXX WFR SEPARATOR PAGE XXX**. Where more than one separator phrase is configured, the **NN** component must begin at **01** and run sequentially.
XCUEnableCustomUOM	YES/NO	This parameter should be set to **YES** to the Export Custom Unit of Measure Value to XML feature.

XCUUseISOCode	YES/NO	If this parameter is set to **YES** the value that will be written to the <*UOM*> tag for the line item in the output XML file will be the corresponding *ISOCode* for the extracted UOM value, as defined in the *BRWUOM section* of the project configuration.
		
		For example, assume the following settings:
		
		`UOM_VL_02_ISOCode=EA`
		`UOM_VL_02_Alias=Each,EACH`
		`UOM_VL_02_ExportValue=Each`
		
		If a line item unit of measure was extracted as **EACH**, and this parameter is set to **YES**, the value written to the XML file will be **EA**, because that is the *ISOCode* setting that corresponds to the *Alias* parameter for this UOM group, where the extracted value exists in the comma-separated list of aliases.
		
		If this parameter is set to **NO,** the value that will be written to the <*UOM*> tag for the line item in the output XML file will be the corresponding (optional) *ExportValue* for the extracted UOM value, as defined in the

Parameter	Type	Description
		BRWUOM section of the project configuration. In the example given above, the value written to the XML file would be **Each**. If the Export Custom Unit of Measure Value to XML feature is not enabled, or if the extracted *Alias* cannot be found in the project configuration, or if the *ExportValue* setting is not configured, the extracted unit of measure will be written to the XML file.
IFVEnableIVRForce Validation	YES/NO	If this parameter is set to **YES,** the Verifier user can forcibly validate a document by selecting one of the *Invalid Reason* options defined in the *IFVInvalidReasonGroups* parameter. This allows the document to pass through to export with known invalid values in one or more of the extraction fields.

IFVInvalidReasonGroups	Freetext	Comma-separated list of Invalid Reason groups that, if selected by the Verifier user, will cause the document to be considered valid and ready for export, irrespective of whether one or more fields failed validation.
		
		For example, assume the project administrator added the following custom invalid reasons to the project configuration:
		
		`IVR_VL_10_Rule=SETAMOUNTSTOVALID`
		`IVR_VL_10_VerifierDisplay=COULD NOT VALIDATE DOCUMENT`
		`IVR_VL_10_ExportCode=10`
		`IVR_VL_11_Rule=SETAMOUNTSTOVALID`
		`IVR_VL_11_VerifierDisplay=KNOWN INVALID DOCUMENT`
		`IVR_VL_11_ExportCode=11`
		
		To enable the Verifier user to select either of the above Invalid Reasons to forcibly validate a document and release it for export, this parameter should be configured as follows:
		
		`WFR_VL_IFVInvalidReasonGroups=10,11`

FXIFormatXMLforOIT	YES/NO	If set to **YES**, one or more functions of the Format XML for OIT feature will be enabled, depending on which are configured.
		
		If set to **NO**, none of the functions provided by this feature will be enabled.
		
		When this setting is set to **YES,** it is important that the following settings in the *BRWEXP section* have their values removed:
		
		`EXP VL XMLTableLineType=`
		`EXP_VL_XMLTableQuantity=`
		`EXP_VL_XMLTableUOM=`
		`EXP_VL_XMLTableUnitPrice=`
		
		Failure to remove those values will not prevent this feature from working correctly, but will result in duplicate tags being written to the XML output. Instead, the following settings in the *BRWEXP section* must be set:
		
		`EXP_VL_XMLTableOITLineType=lineType`
		`EXP VL XMLTableOITQuantity=quantity`
		`EXP VL XMLTableOITUOM=UOM`
		`EXP VL XMLTableOITUnitPrice=unitPrice`

FXIDefaultLineType	Freetext	Defines the line type value that should be used by default when the sample is configured to set blank values to the default or to replace defined values with the default.
FXISetBlanksToDefault	YES/NO	If set to **YES**, the line type tag in the XML output will be populated with the value defined in **FXIDefaultLineType**. If set to **NO,** any lines where the line type is blank will have an empty tag written to the XML.
FXILineTypesToDefault	Freetext	Comma-separated list that defines which line type value(s) should be replaced with the value defined in **FXIDefaultLineType** when it is written to the XML file. This setting can be useful in implementations where the item category is being used for Service PO determination, in which case the value from the

Parameter	Type	Description
		defined item category field would typically be written to the XML file as the line type. Leave this setting blank if no line type values should be set to the defined default value.
FXIEmptyQUPUOMforLine Type	Freetext	Comma-separated list that defines which line types should have empty quantity, unit price and UOM tags in the XML output file. The value(s) specified here should match the values in the line type field of the line item. Leave this setting blank if no line types should be written to the XML file with empty quantity, unit price and UOM fields (i.e. the actual values should be written to the XML file).
FXIFlipLineQUPforPOType	Freetext	Comma-separated list that defines which line types should be written to the XML file with the quantity and unit price values transposed. If enabled, the quantity value will be written to the unit price tag in the XML file, and vice versa. Leave this setting blank if this flip should not occur for any line types.

10.2 AP Packaged Project Database Settings

The AP Packaged Project database is subdivided into a series of tables, each containing settings that relate to various aspects of the solution. These settings can be configured to meet your business requirements.

10.2.1 BRWPONFormats

In this table, the purchase order number formats are defined for the purposes of extraction. The table is keyed by both the profile ID and the index column. You can set the parameters listed in the following table:

Parameter	Type	Description
ProfileID	Integer	This is the profile ID.
IndexID	Integer	This is the index of the PO format. This must be unique per profile ID.
Format	Freetext	This is the format string for a possible PO number. Multiple formats can be entered per profile ID, with one per index. # is a wildcard character representing any number; @ represents any alpha character. For example, 45######## would denote a possible purchase order number as a ten-digit number beginning with 45. @@###### would denote a possible purchase order number as being two alpha characters followed by 5 digits. We recommend defining these format strings as tightly as possible.
IgnoreCharacters	Freetext	This is the list of characters that may appear in a purchase order number in any position, such as a hyphen or a period/full stop, that the system must be tolerant of in the corresponding format string specified. This list does not need to be comma-separated.

10.2.2 BRWPON

This section controls the validation of purchase order numbers, as well as how service purchase orders are identified. You can set the parameters listed in the following table:

Parameter	Type	Description
ProfileID	Integer	This is the profile ID.

Parameter	Type	Description
MaxWordCount	Freetext	Set this value to the maximum number of OCR words that are permitted to form a candidate for the purchase order number. It is recommended to set this value to 3. This allows for the following: 45 0000 0020 and 12345 – OP to be recognized as possible purchase order numbers, where each value separated by a space is recognized as an individual OCR word. A hyphen (-) is considered by the system to be a separate OCR word, regardless of the geometric distance to neighboring values on either side.
SetCompanyCodeFromPO	Boolean	If set to TRUE, the system overwrites any existing content in the company code field with the company code derived from the purchase order.
ValidateFromDB	Boolean	This flag denotes whether the extracted purchase order is validated against a database table. If this validation is activated, the extracted purchase order must exist in the database table and the vendor identified on the invoice must be the vendor on the purchase order. In the case of Oracle implementation, the invoice site ID address does not have to be the same as the PO order from vendor site ID. In all other cases, the purchase order number field is set to invalid.
ReadPOHeaderViaUserExit	Boolean	Set this flag to TRUE if the purchase order header details are retrieved using a custom lookup in UserExitReadPODetails.
POKeyIncludes CompanyCode	Boolean	Set this value to TRUE if the unique key to identify a single purchase order in the PO header database lookup table consists of the purchase order number and the company code in tandem. Do not set this value to TRUE for JD Edwards implementations.
UseStoredProcedure	Boolean	Set this value to TRUE if the PO header details are retrieved from a database using a stored procedure.
StoredProcedureName	Freetext	This is the technical name of the stored procedure used to retrieve the PO header details.
StoredProcedure Parameters	Freetext	This is a comma-separated list of parameters relevant to calling the stored procedure. The parameters listed must correspond to entries maintained within the table BRWSPC.
SQLConnectionGroup	NN	This is the SQL connection group that specifies the purchase order database connection string as set in the SQL section. If no connection group is specified, the system uses group 01.
UsePOPartition	Boolean	This value must be set to TRUE if the purchase order database lookup must be set to use a purchase order partition. Purchase order partitions are set against the client in the BRWClient table.
DBTableName	Freetext	This is the name of the database table that contains the purchase order header information. This setting is mandatory for database validation where no stored procedure is used.
DBPartitionID	Freetext	This is the technical name of the purchase order header database table column which holds the record partition ID. If the UsePOPartition column is set to TRUE, this value must be populated to prevent the system from raising an error.
DBPO	Freetext	This is the name of the database table field that holds the purchase order number.
DBVendorID	Freetext	This is the name of the database table field that holds the vendor ID for a given purchase order. This setting is mandatory for database validation. This must always be set to the internal vendor ID that the ERP system uses.
DBSiteID	Freetext	This is the name of the database table field that holds the site ID for a given purchase order. This must be mapped when working with ERP systems that use a vendor ID and a site ID to identify a unique vendor address. (e.g. Oracle Financials) The column must not be mapped under any other circumstances.
DBCurrency	Freetext	This is optional and is the name of the database table column that holds the purchase order document currency.

Parameter	Type	Description
DBCompanyCode	Freetext	This is the name of the database table column that holds the company ID for which the purchase order is created. This is only mandatory for JD Edwards purchase orders or where the POKeyIncludesCompanyCode parameter is set to YES.
DBStatus	Freetext	(Optional) This is the name of the database table column that holds the status of the purchase order, such as released, closed, and so on.
DBDocType	Freetext	This is optional and is the name of the database table column that holds the purchase order document type. This is only mandatory for JD Edwards purchase orders.
DBBusinessUnit	Freetext	This is the name of the database table column that holds the purchasing business unit. This is only mandatory for PeopleSoft purchase orders.
JDEPO	Boolean	If this value is set to TRUE, the system looks for and validates purchase order numbers based on the JD Edwards ERP system formats and data structures. A unique purchase order is identified by the combination of the company code, purchase order number, and purchase order type. In Verifier, the PO Extension field is considered as a mandatory field for PO-based invoices where the JD Edwards purchase order type is entered.
JDEPOTypes	Freetext	This is a comma-separated list of valid JD Edwards purchase order types. If JDEPO is set to TRUE, purchase order numbers are not extracted unless a valid purchase order type is also specified on the document, before or after the actual purchase order number itself. Typical JD Edwards purchase order types are OP and UX.
PeopleSoftPO	Boolean	If this value is set to TRUE, the system looks for and validates purchase order numbers based on the PeopleSoft ERP system formats and data structures. A unique purchase order number is identified by a combination of the purchase order number itself and the purchasing business unit. In Verifier, the PO Extension field is considered as a mandatory field for PO-based invoices where the purchasing business unit is entered.
PeopleSoftBusinessUnits	Freetext	This is a comma-separated list of valid PeopleSoft purchasing business units. If PeopleSoftPO is set to TRUE, purchase order numbers are not extracted unless a valid purchasing business unit is also specified on the document, before or after the actual purchase order number itself.
ServicePOTypes	Freetext	This is the comma-separated list of purchase order document types that denote a service purchase order.
ServicePOItemCategories	Freetext	This is the comma-separated list of item categories and line types at the purchase order line item level that denote a service.
ServicePOPrefixes	Freetext	This is the comma-separated list of purchase order prefixes that exclusively identify a service purchase order.
ServicePOUOMs	Freetext	This is the comma-separated list of units of measure at the purchase order line item level that exclusively identifies a service purchase order.
StopERSPO	Boolean	If set to TRUE, the system stops a purchase order in Verifier if it is marked as an ERS/self-billing purchase order.
StopERSPOInVerifier	Boolean	If set to TRUE, the system does not permit a purchase order to pass Verifier if it is marked as an ERS/self-billing purchase order.

Parameter	Type	Description
SkipDuplicatePOCheck	Boolean	If set to FALSE, the system stops a document in Verifier if a database lookup is activated and the purchase order number exists more than once in the purchase order header table. The user may choose to accept the PO in Verifier, but line pairing is not carried out. If set to TRUE, the system does not reject a purchase order if a duplicate record is found in the purchase order number header table, but instead the system uses the first one found, and the line pairing is carried out as usual. If the purchase order is keyed using a combination of the purchase order number and a company code/business unit/PO document type, which can be the case for implementations involving JD Edwards or Peoplesoft, then a duplicate is flagged only if multiple records are found using all keys defined.
KeepCaseSensitive	Boolean	If set to TRUE, the system retains lower case characters in an extracted purchase order number.

10.2.3 BRWBTO

This table is used to specify format strings for possible bill-to names on the invoice which are extracted into the bill-to name field.

You can set the parameters listed in the following table:

Parameter	Type	Description
ProfileID	Integer	If set to TRUE, the system does not set the bill-to name field to invalid if a valid bill-to name has not been extracted on server side.
Distance	Numeric	Value between 0 and 1 representing the fuzzy tolerance that must be applied when finding a match to the formats set in BRWBTOFormats. 1 is the most fuzzy and 0 requires a precise match.

10.2.4 BRWBTOFormats

This table is used to specify format strings for possible bill-to names on the invoice which are extracted into the bill-to name field.

You can set the parameters listed in the following table:

Parameter	Type	Description
ProfileID	Integer	This is the profile ID.
IndexID	Integer	This is the index for bill-to name format. This must be unique for each profile ID.
Format	Freetext	This is the format string for a possible bill-to name. Multiple formats can be entered per profile ID, one per index. # is a wildcard character representing any number and @ represents any alpha character. The format string entered is used to help the system home in on the correct bill-to name. For example, if ACME is specified as a possible bill-to, the system uses this to help anchor the bill-to name on the invoice, but the field is extracted as it appears on the document. Therefore, if the bill-to was actually ACME UK Office, then this would be the value extracted.
IgnoreCharacters	Freetext	This is the list of characters that may appear in a bill-to name in any position (e.g. a hyphen or a period) that the system must be tolerant of in the corresponding format string specified. This list does not need to be comma-separated.

10.2.5 BRWAMT

The settings in this table specify the format string and ignore characters for a valid amount used by the project.

We do not recommend that this setting is changed from the following default format:

```
Format=#[2-10] Ignore=,.'-_$£€¥^
```

You can add more Ignore characters to cover a wider variety of currency symbols.

You can set the parameters listed in the following table:

Parameter	Type	Description
ProfileID	Integer	This is the profile ID.
Format	Freetext	This is the simple expression denoting the format of an amount.
IgnoreCharacters	Freetext	This is the special characters that may appear in the amount.
DeactivateCrossValidation	Boolean	If set to TRUE, the mathematical checks applied to the extracted invoice amounts in Verifier are deactivated.
ExportThousandSeparator	Freetext	This is the character representing the thousand separator that must be used when formatting the amount during data export.
ExportDecimalSeparator	Freetext	This is the character representing the decimal separator that must be used when formatting the amount during data export.

10.2.6 BRWDTYFormats

This table is used to specify the words that are displayed on an incoming document to denote that the document is a credit memo. For example, CREDIT NOTE, CREDIT MEMO, AVOIR, or GUTSCHRIFT. The table is client and profile independent. You can set the parameters listed in the following table:

Parameter	Type	Description
IndexID	Integer	This is the unique index for the table row.
Format	Freetext	This is the format string for a possible credit note indicator. # is a wildcard character representing any number and @ represents any alpha character.
IgnoreCharacters	Freetext	This is the list of characters that may appear in a credit note indicator in any position, such as a hyphen or a period/full stop, that the system must be tolerant of in the corresponding format string specified. This list does not need to be comma-separated.

10.2.7 BRWDTY

This table holds additional settings relating to the document type field. The parameters listed in the following table can be set.

Parameter	Type	Description
ProfileID	Integer	This is the profile ID.
Distance	Freetext	This represents the fuzzy factor the system uses when determining document types based on the entries above. It is a value between zero and one where zero requires an exact match and one accepts values that do not match at all. The recommended value for this setting is 0.17.
StopAllCredits	Boolean	If this flag is set to TRUE, then all documents identified for credit stop will be put for manual review in Verifier.
IgnoreNegativeTotal	Boolean	If set to TRUE, the system does not use the presence of a negative invoice amount to determine whether the document type must be a credit memo, as opposed to an invoice. Only the format strings defined in the BRWDTYFORMATS table affect the credit memo recognition.

10.2.8 BRWDTYTexts

If the dynamic Verifier form is being used, this table contains the field text that is presented to a user within the Verifier application for the Document Type field. Do not change the existing

content of this table. However, you can add new rows that reference an existing TextID to support translations for additional languages.

Parameter	Type	Description
TextID	Integer	This is the unique ID for the text element. This must be set to either 1 for INVOICE or 2 for CREDIT.
LanguageID	Freetext	This is the two-character language ISO-code for the error message text, for example, EN is English, DE is German, and CN is Chinese.
DisplayText	Freetext	This is the display text for the language specified in the LanguageID column.

10.2.9 BRWITY

This table includes configuration settings that determine the invoice type field in AP Packaged Project, and whether it is set to `PO` or `NO-PO` Invoice Type. You can set the parameters listed in the following table:

Parameter	Type	Description
ProfileID	Integer	This is the profile ID.
DefaultValue	PO or NPO	This specifies the default value for the field, where PO = purchase order related; NPO = non-purchase order invoice. If the invoice type exists in the image file name and has been mapped in the INI file IMP section, then this overrides the default.
SetByVendor	Boolean	This specifies whether the invoice type field must be set by the extracted vendor. For this to occur, the invoice type value must be mapped to a column in the vendor extract in the BRWSRC table.
SetByVendorCC Exceptions	Freetext	This is the comma-separated list of the company codes that are exceptions to the SetByVendor parameter above. For example, if SetByVendor is set to TRUE, and SetByVendorCCExceptions is set to 1000,2000, then the system sets the invoice type according to the vendor except when the invoice belongs to company code 1000 or 2000. If SetByVendor is set to FALSE, then the system sets the invoice type according to the vendor only if the invoice belongs to the company code 1000 or 2000.
POValue	Freetext	This denotes the value held in either the document file name or in the vendor extract that represents a PO invoice.
NPOValue	Freetext	This denotes the value held in either the document file name or in the vendor extract that represents a NO-PO invoice.
SetToPOIfPOFound	Boolean	On RTS, if this setting is set to TRUE and a purchase order number is found, the invoice type is set to PO irrespective of any default or vendor specific settings.
SetToPOIfValidPOFound	Boolean	On RTS, if this setting is set to TRUE and an extracted purchase order is found to exist in a validation database, the invoice type is set to PO irrespective of any default or vendor-specific settings.
SetToPOIfPOPopulated	Boolean	If this parameter is set to TRUE, the system always sets the invoice type to PO if any input is present in the purchase order number field, whether captured by RTS or entered manually by a Verifier user.

10.2.10 BRWITYTexts

If the Dynamic Verifier form is being used, this table holds the field text that is presented to a user within the Verifier application for the Invoice Type field. Do not change the existing content of this table. You can add new rows that reference an existing TextID to support translations for additional languages.

Parameter	Type	Description
TextID	Integer	This is the unique ID for the text element. This is set to 1 for PO or 2 for NO-PO.
LanguageID	Freetext	This is the two-character language ISO-code for the error message text. For example, EN is English, DE is German, and CN is Chinese. AP Packaged Project ships with English and Chinese languages pre-configured.
DisplayText	Freetext	This is the display text for the language specified in the LanguageID column.

10.2.11 BRWPTYTexts

This global table is used to hold the field text that is presented to a user within the Verifier application for the **PO Type** field if the dynamic Verifier form is being used.

i.e. `MATERIAL` or `SERVICE`

The existing content of this table must not be changed, but new rows can be added referencing an existing TextID in order to support translations for additional languages.

Parameter	Type	Description
TextID	Integer	This is the unique ID for the text element. This must be set to either 1 for MATERIAL or 2 for SERVICE.
LanguageID	Freetext	This is the two-character language ISO-code for the error message text. For example, EN is English, DE is German, and CN is Chinese.
DisplayText	Freetext	This is the display text for the language specified in the LanguageID column.

10.2.12 BRWNUM

This table contains the formatting and validation options available for the invoice number field. You can set the parameters listed in the following table:

Parameter	Type	Description
ProfileID	Integer	This is the profile ID.
SkipForUtilityVendor	Boolean	If set to TRUE and the vendor is identified as being a utility vendor (if the column is mapped in the BRWSRC table), then the invoice number is set to valid and an account number is required instead.
UtilityAlias	Freetext	This specifies the positive value in the vendor extract that denotes a utility vendor.
AcceptTwoCharacters	Boolean	If set to TRUE, the system does not automatically set a two-character invoice number to invalid on the server side. Otherwise, any invoice number extracted, which is two characters or less in length, is marked as invalid by the system automatically, and the document is sent to Verifier for a user to confirm the extracted value.
ValidateFromDB	Boolean	This is the flag to denote whether the invoice number must be validated against previous invoice numbers from the same vendor in a database table.
SQLConnectionGroup	NN	This is the SQL connection group specifying the invoice number history database connection string as set in the SQL section. If no connection group is specified, the system uses group 01.
DBTableName	Freetext	This is the name of the invoice number history database table.

Parameter	Type	Description
UseVendorPartition	Boolean	This column specifies whether a partition must be used for the invoice number history check. This functionality mirrors the vendor partition functionality, whereby it is possible to hold all invoice number history data within a single database table, then utilise partitions to ensure segregation of one client's data from another. For example, in an AP Project project, there are three clients – client A, client B, and client C. Vendor partitions are used. Clients A & B utilise vendor partition 1, and client C uses vendor partition 2. In previous versions of AP Project, it was necessary to create two invoice number history tables for the configuration above: one table for the vendor pool shared between clients A &B, a second for the vendor pool utilised by client C. In the event that the project involved many clients, this could be cumbersome from an administrational point of view. A single invoice number history pooled between all clients may be used. Hence, if UseVendorPartition is set to TRUE, when the system queries the invoice number history table, it now does so using the vendor partition ID specified for the client in the BRWClient table. Equally, if the invoice number history table needs to be updated at time of export, the system updates the table inserting the relevant vendor partition ID into the column set to represent the partition, which is specified using the PartitionID parameter below. If UseVendorPartition is set to TRUE and no vendor partition has been set in the BRWClient table, the system raises an error. It must be noted that, if an invoice number table with partitions is to be used, then all clients pointing to that table MUST be set to use the partition. If this is not the case, then, at the time of invoice number history update, the system could erase history data belonging to another client if the vendor number happened to be the same. If UseVendorPartition is set to FALSE, the system does not use a vendor partition for the invoice number history check.
PartitionID	Freetext	This is the name of the column that contains the partition ID in the invoice number history lookup table. This setting is not mandatory if invoice partitions are not being used (i.e. UseVendorPartition is set to FALSE). If vendor partitions are being used and this column is set to blank, the system raises an error message, sending the document to Verifier.
VendorID	Freetext	This is the column in the database table which holds the vendor number. This setting is mandatory.
RecID	Freetext	This is the column in the database table which holds the record ID. This setting is mandatory if the invoice number history table is to be updated upon export.
InvoiceNumber	Freetext	This is the column in the database table which holds the invoice number. This setting is mandatory.
DocumentType	Freetext	This is the column in the database table which represents the document type invoice or credit. This setting is mandatory.
InvoiceAlias	Freetext	This is the value in the document type column in the invoice number history table which denotes an invoice. This setting is mandatory.
CreditAlias	Freetext	This is the value in the document column in the invoice number history table which denotes a credit memo. This setting is mandatory.
MaxRecords	Freetext	This is the maximum number of records that must be considered when comparing an extracted invoice number to historic invoice numbers from the same vendor in the database table. The default is set to 20 records.

Parameter	Type	Description
NoOfHits	Freetext	This is the number of hits required to validate the extracted invoice number format with the invoice number history database table. The default is set to two records.
CorrectOCRMisreads	Boolean	If set to TRUE, and database validation of the invoice number is activated, the system looks to repair two specific OCR issues with the extracted invoice number. If 1 has been read in the extracted value, but the invoice history shows that this must be the letter I in as many records as set in the NoOfHits parameter, then the extracted value is changed to an I automatically on server side only. This also applies to instances where a zero has been extracted, and the history shows that a letter "O" is expected. No other OCR issues are handled.
CheckSequencing	Boolean	If this flag is set to TRUE, and the vendor supplies invoices numbers that are entirely numeric, then, when carrying out the invoice number format comparison against the vendor's prior history, the system does not consider prior invoice numbers if they are 100 ahead or more. For example, invoice number 1500 is read from an invoice. In the history table, invoice numbers 1598, 1599, 1600, and 1601 are present but 1600 and 1601 are not considered valid matches., Although they match the 4 numeric format, they are too far ahead in terms of the sequence. The sequence limit, in the above example set to 100, is configurable via the SequencingLimit parameter.
UpdateDBAtExport	Boolean	If set to TRUE, the system updates the invoice number history table with a record for the current document at the point of document export.
ExtractDeliveryNotes IntoTable	Boolean	If set to TRUE, the delivery note number field is deactivated and the table field DeliveryNotes is used to collect delivery note numbers instead.
RemoveInvNoFor Korean	Boolean	If set to TRUE, the system blanks out any extracted invoice numbers for a Korean language document.
SequencingLimit	Integer	This setting is used in tandem with the sequence check functionality described above. If no sequence is specified (i.e. the sequencing limit is set to zero), then the system uses the default sequencing limit of 100.
InvoiceNumberConfidence	Integer	This value is expressed as a percentage (that is 20 = 20%). It is the confidence value that an extracted invoice number must surpass lest it is marked invalid by the system. This check is carried out by the system if the invoice number history check is not in use.

10.2.13 BRWDAT

The settings in this table control the formatting and validation of the invoice date. You can set the parameters listed in the following table:

Parameter	Type	Description
ProfileID	Integer	ProfileID
VerifierFormat	DDMMYYYY, MMDDYYYY or YYYYMMDD	If set to DDMMYYYY, AP Packaged Project displays the date in Verifier as DD/MM/YYYY. If set to MMDDYYYY, AP Packaged Project displays the date in Verifier as MM/DD/YYYY. If set to YYYYMMDD, AP Packaged Project displays the date as YYYY-MM-DD.
ExportFormat	MMDDYYYY DDMMYYYY YYYYMMDD	This is the output date format for export. This setting applies to database output and all flat file exports.
ExportSeparator	Freetext	This is the separator that must be used when exporting a date value. For example, a slash (/), dot (.), or hyphen (-).
MMDDCountries	Freetext	Comma-separated list of countries that use MM/DD/YYYY as the date format preference (e.g. the US).

Parameter	Type	Description
YYMMDDCountries	Freetext	This is the comma-separated list of countries in which YY-MM-DD is a standard date format, such as Sweden, for example. If the vendor country of origin is included in this list and the invoice date is read as 12-01-11, this is formatted as 11/01/2012 (DDMM). If the invoice date year is either the current year, the previous year, or the following year, the invoice date is formatted as 01/11/2012 (MMDD).. If the invoice date year is something else, the format is assumed to be DD-MM-YY.
DeliveryDateIncludesShipDate	Boolean	If set to TRUE, this expands the scope of delivery dates appropriate for extraction to include ship dates.

10.2.14 BRWTAXCONFIG

This table contains the configuration settings relating to the extraction of tax and the automatic determination of tax codes for invoice creation. You can set the parameters listed in the following table:

Parameter	Type	Description
ProfileID	Integer	This is the profile ID.
PrimaryRates	Freetext	This is the comma-separated list of tax rates expected for invoices processed by the AP Packaged Project project. This list is optional, but does assist AP Packaged Project in finding the correct tax value on the document. Values matching the primary rate are considered ahead of values entered as secondary rates.
SecondaryRates	Freetext	This is the comma-separated list of tax rates expected for invoices processed by the AP Packaged Project project. This list is optional, but does assist AP Packaged Project in finding the correct tax value on the document.
ActivateVATComplianceCheck	Boolean	If set to TRUE, the system activates the VAT registration number compliance check, which means that if the value added tax is being charged on the invoice, the vendor and bill-to company VAT registration numbers become mandatory fields. If VAT is being charged in a currency which differs to the local currency of the company code for which the invoice is addressed, then the exchange rate and/or local VAT amount fields are also mandatory.
VATCheckCompanyCode Exceptions	Freetext	This is the comma-separated list of company codes that are an exception from the VAT registration number compliance rule. For example, if the ActivateVATComplianceCheck is set to TRUE, then any company codes listed in this parameter do not have the check carried out. If ActivateVATComplianceCheck is set to FALSE, then only company codes specified in this parameter have the check carried out.
VendorVATCheckOnly	Boolean	If this value is set to TRUE and the VAT compliance check has been activated, the system requires the VAT registration number of the vendor, rather than that of both the vendor and the bill-to party.

Parameter	Type	Description
VendorVATCheckCo CodeExceptions	Freetext	This is the comma-separated list of company codes that are an exception to the vendor-only VAT registration number check. If VendorVATCheckOnly is set to TRUE, then, for the company codes listed against this parameter, both the vendor and bill-to VAT registration numbers are mandatory. If VendorVATCheckOnly is set to FALSE, then only the vendor VAT registration number is required for invoices belonging to a company code specified in this list.
CheckVATCrossBorder	Boolean	If this value is set to TRUE, and the VAT compliance check has been activated, the system requires entry of both the vendor and the bill-to VAT registration numbers if the tax amount on the invoice is zero; both vendor and company code are based in different EU countries; and the vendor has a VAT registration number set in the vendor extract. The cross-border VAT registration number check operates independently of the vendor-only VAT check.
CrossBorderCoCode Exceptions	Freetext	This is the comma-separated list of company codes that are an exception to the EU cross border VAT registration number check. If CheckVATCrossBorder is set to TRUE, then, for the company codes listed against this parameter, the cross-border VAT registration number check is not carried out. If CheckVATCrossBorder is set to FALSE, then the cross-border VAT registration number check is only carried out for invoices belonging to company code specified in this list.
CheckVATRates	Boolean	If this value is set to TRUE, the system checks that the content of the VAT Rate column in the VAT table exists for the country where VAT is applied. The system sets the VAT rate column to invalid if the rate cannot be found. The check is only carried out if all of the following hold true: • The VAT compliance check is activated. • The VAT table is used for capturing the document tax information. • A vendor VAT registration number is read from the document. • The vendor is marked as belonging to an EU member country. • The country where VAT is applied is derived from the first two characters of the vendor VAT registration number, which represent the VAT country. If this is not available, the vendor's country of origin is used. • The valid VAT rates are read from the VAT rates column in the country table configured in the BRWCTR table and are presented as a comma-separated list. • The comma-separated list need not include a VAT rate of zero as this is always accepted.
ActivateTaxDetermination	Boolean	If set to TRUE, the system activates the tax determination and validation feature. Otherwise, no tax code allocations or validations of those allocations are carried out.
AlwaysUsePOTaxCode	Boolean	If set to TRUE, the system always uses the purchase order tax code.

Parameter	Type	Description
AlwaysUseCalculateTaxFlag	Boolean	If set to TRUE, the system always requires the ERP system to calculate the tax amount rather than pass the tax amount read from the invoice.
TaxFlagException CompanyCodes	Freetext	This is the comma-separated list of company codes that are an exception to the calculate tax flag rule. For example, if the AlwaysUseCalculateTaxFlag parameter is set to TRUE, then company codes listed in this parameter do not use the calculate tax flag. If the AlwaysUseCalculateTaxFlag parameter is set to FALSE, then the calculate tax flag is used for company codes specified in this list.
ValidateFromDB	Boolean	This is the flag to denote whether tax codes must be derived and/or validated against a database table. This table must not be used for countries where tax jurisdictions apply, for example the US, Brazil, and Canada.
SQLConnectionGroup	NN	This is the SQL connection group specifying the tax code database connection string as set in the SQL section. If no connection group is specified, the system uses group 01.
DBTableName	Freetext	This is the name of the tax code database table.
CheckForICMSTax	Boolean	ICMS is a form of sales tax used on Brazilian Nota fiscal documents. If this parameter is set to TRUE, then the system attempts to identify this tax amount on incoming documents and appies the validations to this field. It must be set to FALSE if the client is not using the solution to process Brazilian Nota fiscal documents.
DeriveShipToFrom CompanyCode	Boolean	If set to TRUE, the ship-to country is set to the country in which the company code is based.
ReadPlantFromDB	Boolean	If set to TRUE, the system reads plant information from a database in order to determine the ship-to location for the goods for tax calculation purposes.
PlantSQLConnectionGroup	NN	This is the SQL connection group specifying the plant database connection string as set in the SQL section. If no connection group is specified, the system uses group 01.
DBPlantTable	Freetext	This is the name of the database table containing the plant information.
DBPlant	Freetext	This is the name of the column in the plant database table holding the plant ID.
DBPlantCountry	Freetext	This is the name of the column in the plant database table holding the country in which the plant is located.
DBPlantState	Freetext	This is the name of the column in the plant database table holding the state in which the plant is located.
DBPlantTaxJurCode	Freetext	This is the name of the column in the plant database table holding the tax jurisdiction code associated with the plant's location.
ExtractTaxIntoVATTable	Boolean	If set to TRUE, the AmountTax, PST, HST, ICMS and AmountSubtotal fields is deactivated and the relevant columns in the VATTable are used instead.
BreakOutHSTForCanada	Boolean	This is the flag to indicate whether HST must be captured within the HST field as opposed to in the AmountTax field.

Parameter	Type	Description
UseTaxPartition	Boolean	This flag specifies whether the tax partition ID must be used when ascertaining the tax code from the database table specified against the DBTableName parameter. Usage of a tax partition allows tax code data relating to all clients to co-exist within the same database table, thus alleviating the administrative burden of maintaining multiple tax code tables if multiple clients are being used. The tax partition ID value is set in the BRWClient table for the relevant client. Hence, when the system reads the tax table in order to retrieve the tax codes relevant for the invoice company code country, the client tax partition ID is also used to ensure that the correct set of records for the client is being selected. Within the tax code database table, the column PARTITIONID is expected to contain the partition ID for each row. Export fails if this column does not exist. If the column is not populated, the system fails to retrieve any tax codes, and hence, coding of the invoice lines for tax is not successful.
CNPasswordSpecials	Freetext	This is a list of special characters that are permitted to appear in the password block that appears on a domestic VAT invoice used in mainland China. The password must only consist of digits 0-9, and the six special characters listed within this field, which default to the current standard of '+<>-*/'. This field is configurable to permit further characters to be added must the government standard change in the future. If characters fall outside of this range, the system rejects the password field content captured automatically or entered manually by a user..
CNPasswordValidLengths	Freetext	This is a comma-separated list of the valid lengths for a password block appearing on a domestic VAT invoice used in mainland China. The default values are 84,108, which means that a captured password must be either 84 or 108 characters in length before the system accepts it. This reflects the currency government standard, but the field is configurable in case this changes in the future.

10.2.15 BRWTXJCodes

This table is used to configure the behavior of tax codes for countries that use tax jurisdictions during the automatic tax determination procedure. You can set the parameters listed in the following table:

Parameter	Type	Description
ProfileID	Integer	This is the profile ID.
IndexID	Integer	This is the numeric tax code index.

Parameter	Type	Description
TaxCode	Freetext	This is the tax code for the tax code settings group. This is the settings group used for countries where tax jurisdictions exist, such as the US, Brazil, and Canada. In such instances, the tax code denotes whether the item is taxable or tax exempt, as well as an instruction for processing, such as self-assess. For other countries, the OFRTAX table must be used for automatic tax code determination and validation.
Country	Freetext	This is the country to which the group tax code belongs, such as US, CA.
IfTax	Freetext	If there is tax on the invoice and the purchase order line item tax code is set to the value in TaxCode and the company code country is equal to the value in Country, then this value is set as the new tax code for the invoice line.
IfNoTax	Freetext	If there is no tax on the invoice and the purchase order line item tax code is set to the value in TaxCode and the company code country is equal to the value in Country, then this value is set as the new tax code for the invoice line.
VendorState	Freetext	This is the comma-separated list of vendor states that would trigger the selection of a state-dependent tax code.
ShipToState	Freetext	This is the comma-separated list of ship-to states that would trigger the selection of a state-dependent tax code.
IfTaxState	Freetext	This is the tax code for use if the vendor is from a state listed in VendorState, or the goods were shipped to a state listed in ShipToState and tax is being charged on the invoice. If both VendorState and ShipToState are populated, then the vendor state must be in the vendor state list and the ship-to state must be in the ship-to state list, else the IfTax code is used.
IfNoTaxState	Freetext	This is the tax code for use if the vendor is from a state listed in VendorState, or the goods were shipped to a state listed in ShipToState and no tax is being charged on the invoice. If both VendorState and ShipToState are populated, then the vendor state must be in the vendor state list and the ship-to state must be in the ship-to state list, else the IfNoTax code is used.
PayTaxAsBilled	Boolean	If there is a tax amount read from the invoice and any of the invoice line tax codes are set to the value held in TaxCode and this flag is set to TRUE, the system books the tax amount as billed on the invoice, rather than allowing the system to calculate it automatically.
ShortPayIfTax	Boolean	If there is a tax amount on the invoice and all invoice line item tax codes belong to a group that have this flag set to TRUE, then the invoice is booked minus the tax.
AccountAssignment Categories	Freetext	This is the comma-separated list of purchase order line account assignment categories. If no tax code is present on the purchase order line and the company code country is equal to the value held in Country and the purchase order line has an account assignment present in this list, then the tax code held in TaxCode is defaulted as the initial purchase order line item tax code.

10.2.16 BRWCUR

This table contains configuration settings associated with the invoice currency.

You can set the parameters listed in the following table:

Parameter	Type	Description
ProfileID	Integer	This is the profile ID.
DollarSignIsUSD	Boolean	If this flag is set to TRUE and no currency has been unambiguously determined from the invoice, but a dollar sign has been found in the OCR text, the currency is set to USD.
DefaultPOCurrency	Boolean	If this setting is set to TRUE and no currency can be identified from the invoice, the system displays the currency from the purchase order by default.
DefaultVendorCurrency	Boolean	If this setting is set to TRUE and no currency can be identified from the invoice, the system displays the currency for the vendor's country of origin by default.
ValidateFromDB	Boolean	If this setting is set to TRUE, then the contents of the currency field in Verifier are validated against a database to check that the currency is valid.
SQLConnectionGroup	NN	This is the SQL connection group specifying the currency database connection string as set in the SQL section. If no connection group is specified, the system uses group 01.
DBTableName	Freetext	This is the name of the database table against which the currency is validated. This setting is mandatory if a database lookup is to be performed.
DBColumnName	Freetext	This is the name of the column in the database table containing the valid currency code. This setting is mandatory if a database lookup is to be performed.
Activate ExtendedValidation	TRUE	Setting this flag to TRUE performs an extended validation on a currency determined automatically by the system. The extended validation compares the system-extracted currency to the following, in the order specified. • Permitted global currencies (see GlobalCurrencies parameter below). • The vendor currency if present in the vendor extract. • The currency associated with the vendor country of origin. • The currency of the company code. • The currency associated with the company code country. If the invoice currency does not match any of the above, then the currency is set to invalid for a user to check within the Verifier application. Errors caused by missing, incomplete or incorrect configuration in connection to company code and country lookup codes also set the currency field to invalid. User input in Verifier is excluded from the checks above and are assumed to be correct in so far as a valid currency is entered. The extended validation is only carried out for invoice classified to the generic node.
GlobalCurrencies	Freetext	This is the comma-separated list of permitted global currencies used in the extended currency validation such as USD and EUR as described above.

10.2.17 BRWCurrency

This table contains global settings for invoice currencies.

This table functions independently of the client and/or profile ID and comes pre-populated with entries covering many world currencies as part of the solution install. Additional currencies may be added as needed.

You can set the parameters listed in the following table:

Parameter	Type	Description
IndexID	Integer	This is the currency index.
ISOCode	Freetext	This is the currency ISO code for the currency settings group.
Alias	Freetext	This is the comma-separated list of aliases for the currency represented by ISOCode, such as Pounds Sterling for GBP; U.S. Dollars or U.S. funds for USD and Swiss Francs for CHF. If any items on this list are found on the document, then the extracted invoice currency is set to the currency in ISOCode.
AmountPrefix	Freetext	This is the comma-separated list of values that may precede or follow an amount on the invoice that represent the currency specified in ISOCode. For example, US$ would be the prefix for US$1000.00.
Symbol	Freetext	This is the symbol associated with the currency held in ISOCode. This must only be set to a special character, not a single letter.
Country	Freetext	This is the country associated with the currency held in ISOCode. If the symbol for the currency settings group is found on the document and that symbol is unique across all currency settings groups and the vendor country matches the country held in this setting, then the value held in ISOCode is extracted as the invoice currency.
ToleranceGroup	Freetext	This is the tolerance group for the currency, as defined in the BRWTOL table of the system configuration. If no tolerance group is set, then tolerance group 01 is used by default. If tolerance group 01 does not exist, then a tolerance of 0.0001 is used at header level, at line item level, and for the validation of tax.

10.2.18 BRWCCO

This table contains configuration settings that control the validation of the company code.

You can set the parameters listed in the following table:

Parameter	Type	Description
ProfileID	Integer	This is the profile ID.
ValidateFromDB	Boolean	If set to TRUE, the company code is validated against a database.
SQLConnectionGroup	NN	This is the SQL connection group specifying the company code database connection string as set in the SQL section. If no connection group is specified, the system uses group 01.
DBTableName	Freetext	This is the name of the company code database table. This setting is mandatory if a database lookup is to be performed.

Parameter	Type	Description
DBColumnName	Freetext	This is the name of the column in the company code database table that holds the valid company codes. This setting is mandatory if a database lookup is to be performed.
DBCountry	Freetext	This is the name of the column in the company code database table that holds the country in which the company is legally based.
DBCurrency	Freetext	This is the name of the column in the company code database table that holds the currency for the company code. This is not mandatory if the currency is going to be validated against a database.
DBVATRegNos	Freetext	This is the name of the column in the company code database table that holds the VAT registration numbers for which the company is registered. If the company is registered for VAT in more than one country, the database field must contain a comma-separated list.

10.2.19 BRWSRC

This table contains the mapping between columns in the vendor master data table and the values used internally within the AP Packaged Project project. It is also used for the mapping of columns from the employee master data table, so these columns must have the same technical name as those in the vendor master table. BRWSRC is a global table and works independently of the client and profile. The table must consist of a single row.

You can set the parameters listed in the following table:

Parameter	Type	Description
ID	Freetext	This is the AP Packaged Project ASE column name denoting the vendor ID. For ERP systems where a vendor at a unique address is represented by a combination of the vendor ID and the site ID, the formula for the ID column must be set to the following: Vendor ID * 1000000 + Site ID If the vendor ID or site ID is alphanumeric, the formula is as follows: VendorID~SiteID The delimiter (~ in the previous example) is configurable with the AlphNumSiteSeparator parameter in the BRWVND table. The system raises a configuration error if any of the following takes place: • No delimiter is specified, or • it is more than one character, or • it does not occur, occurs more than once, or occurs as the first character in the unique ASE ID column. The site ID must be mapped to SITEID in the BRWSRC table, and the vendor ID stem must be mapped to EXTERNALVENDORID. If the ERP system uses an external vendor ID, such as the supplier number in Oracle Financials, then this value is mapped to EXTERNALVENDORID and the internal vendor ID stem can remain unmapped. But the vendor ID stem component of the ID field is the internal ERP system vendor ID.
SiteID	Freetext	This is the AP Packaged Project ASE column name denoting the vendor site ID. This must only be mapped if the site ID forms part of the ID column above.
Name	Freetext	This is the AP Packaged Project ASE column name denoting the vendor name.

Parameter	Type	Description
Address1	Freetext	This is the AP Packaged Project ASE column name denoting the first line of the vendor's address.
Address2	Freetext	This is the AP Packaged Project ASE column name denoting the second line of the vendor's address.
City	Freetext	This is the AP Packaged Project ASE column name denoting the vendor's city of origin.
Zip	Freetext	This is the AP Packaged Project ASE column name denoting the vendor zip/postal code.
State	Freetext	This is the AP Packaged Project ASE column name denoting the vendor's state/region.
Country	Freetext	This is the AP Packaged Project ASE column name denoting the vendor's country of origin.
POBox	Freetext	This is the AP Packaged Project ASE column name denoting the vendor's PO Box.
POBoxZip	Freetext	This is the AP Packaged Project ASE column name denoting the postal/zip code that relates to the vendor's PO box.
EUMember	Freetext	This is the AP Packaged Project ASE column name denoting whether the vendor's country of origin is a member of the European Union.
Currency	Freetext	This is the AP Packaged Project ASE column name denoting the currency for the vendor's country of origin.
TaxID1	Freetext	This is the AP Packaged Project ASE column name denoting the vendor primary tax ID.
TaxID2	Freetext	This is the AP Packaged Project ASE column name denoting the vendor secondary tax ID.
VatRegNo	Freetext	This is the AP Packaged Project ASE column name denoting the vendor VAT registration number. Multiple VAT registration numbers must be presented as a comma-separated list if the vendor is VAT registered in more than one country. VAT registration numbers must contain the two-character country prefix, but this is not mandatory.
TaxJurCode	Freetext	This is the AP Packaged Project ASE column name denoting the vendor's tax jurisdiction code (US).
TelNo	Freetext	This is the AP Packaged Project ASE column name denoting the vendor's telephone number.
InvoiceType	Freetext	This is the AP Packaged Project ASE column name denoting whether the vendor is permitted to submit a NO-PO invoice.
PaymentMethods	Freetext	This is the AP Packaged Project ASE column name denoting the vendor's payment methods.

Parameter	Type	Description
BankDetails	Freetext	This is the AP Packaged Project ASE column name denoting the vendor's bank account details. This is a colon-separated list that uses the following format: BankAccount,SortCode,ERPBankAccountCode A sortcode is the US equivalent of a routing number.
WithholdingTax Details	Freetext	This is the AP Packaged Project ASE column name denoting the vendor's withholding tax details. This must be a colon-separated list in the following format: CompanyCode,WithholdingTaxType,WithholdingTaxCode If there is withholding tax on the invoice, at time of posting, the system posts the full withholding tax amount to the entry with the relevant company code.
CompanyCodes	Freetext	This is the AP Packaged Project ASE column name denoting a comma-separated list of company codes for which the vendor is valid.
UtilityFlag	Freetext	This is the AP Packaged Project ASE column name denoting whether the vendor is a utility vendor.
PORSubscriberNo	Freetext	This is the AP Packaged Project ASE column name denoting the vendor's Post Office Reference number as used in Switzerland.
ExternalVendorID	Freetext	This is the AP Packaged Project ASE column name denoting the vendor's external ID, such as the supplier number as used in Oracle Financials. If no external vendor ID is used by the ERP system, but the combination of a vendor ID and a site ID is used to identify a unique vendor address, this column must be mapped to the vendor ID stem.
ExternalSiteID	Freetext	This is the AP Packaged Project ASE column name that represent the vendor's external site ID. This mapping is required if the data export involves sending XML to ProcessIT.
SiretID	Freetext	This is the AP Packaged Project ASE column name that represents the vendor SIRET ID number, as assigned to French vendors.
VendorIdentifier	Freetext	This is the AP Packaged Project ASE column name that represents a unique vendor identifier, such as a Chinese tax registration number.
PartitionID	Freetext	This is the AP Packaged Project ASE column name that represents the vendor partition ID.
EUMemberAlias	Freetext	This contains the value that denotes a positive identification of the vendor as an EU state member.
Custom1	Freetext	This is the AP Packaged Project ASE column name that represents an item of custom vendor information.
Custom2	Freetext	This is the AP Packaged Project ASE column name that represents an item of custom vendor information.
Custom3	Freetext	This is the AP Packaged Project ASE column name that represents an item of custom vendor information.
Custom4	Freetext	This is the AP Packaged Project ASE column name that represents an item of custom vendor information.

Parameter	Type	Description
Custom5	Freetext	This is the AP Packaged Project ASE column name that represents an item of custom vendor information.

10.2.20 BRWVND

This table contains settings for validating an extracted vendor number. You can set the parameters listed in the following table:

Parameter	Type	Description
ProfileID	Integer	This is the profile ID.
ValidateFromASSA	Boolean	This denotes whether an extracted vendor ID must be validated against the AP Packaged Project Associative Search Engine Pool / Vendor Extract. We recommend setting this parameter's value to TRUE.
AlphNumSiteSeparator	Freetext	This is the special character used to separate a vendor ID and site ID in the unique ID column in the vendor ASE pool.
IgnorePOVendor	Boolean	If set to TRUE, the system always uses the vendor determined by the system rather than defaulting to the remit-to vendor set on the purchase order. To set the purchase order to valid, it is no longer required that the purchase order vendor must be found on the document. Instead, both values are obtained and validated independent of one another, although an otherwise invalid vendor is set to valid if it is corroborated by the vendor on the purchase order. Therefore, the invalid reasons for PO VENDOR <> INVOICE VENDOR and THIRD PARTY FREIGHT no longer apply.
UseASSAIfPOVendorInvalid	Boolean	If set to TRUE, and if none of the vendors on the purchase order can be validated against the document, then the system evaluates the vendor it determined . If the system-determined vendor can be validated, this is displayed in the vendor ID field, and the invalid reason is automatically set to PO VENDOR <> INVOICE VENDOR. If the system-determined vendor cannot be validated, it is displayed in the field, but is set to invalid. The content of the invalid reason field is not changed. This parameter only takes effect if Ignore POVendor is set to FALSE.
DefaultCountry	Freetext	If no country column is available in the vendor extract used by the VendorASSA field or the value in the country column is blank, a default country for all vendors may be specified here. This must be a two-character ISO-code, such as United States = US, United Kingdom = GB, Germany = DE.
UseBillToBasedVendor Extraction	Boolean	If this value is set to TRUE and a bill-to name field has been captured from the invoice, the system dynamically modifies the search area for the invoice vendor to include the area of the first page of the document above where the bill-to name was detected, along with the bottom ten percent of the first page of the document.

Parameter	Type	Description
RefineVendorExtraction	Boolean	If this value is set to TRUE, the system manipulates the weighting of the candidates for the vendor, adding additional confidence if the vendor name, the vendor street address, the vendor zip code, the vendor VAT registration number and the vendor SIRET ID can be found physically on the document. AP Packaged Project bumps up the ASE delivered weightings by the following: • For Zip only = +10% • For Zip + Vendor Name = +20% • For Zip + VAT Code = +20% • For Zip + Street Name = +20% • For Zip + Street Name + Vendor Name = +30% • For Zip + Street Name + VAT Reg = +30% • For Zip + Street Name + VAT Reg = +30% • For Zip + Street Name + VAT Reg + Vendor Name = +40% An additional 20% is added to any of the above if the SIRET ID or a unique vendor identifier is found on the document (for French vendors).
CheckForAlternatePayees	Boolean	If this parameter is set to TRUE, this activates the alternate payee functionality for the profile ID.
CheckNameForNOPO	Boolean	If this parameter is set to TRUE, the system does not allow a vendor ID to validate automatically unless either the vendor name or the vendor VAT registration number is found in the OCR text of the document. This applies to all documents except PO invoices where a valid PO is found and IgnorePOVendor is set to No.
AlwaysUsePOVendorSiteID	Boolean	If this flag is set to TRUE, the system always sets the site ID during automatic extraction based on the site ID on the purchase order header.

10.2.21 BRWTAB

This table contains configuration settings which control the validation of the invoice line item data. You can set the parameters listed in the following table:

Parameter	Type	Description
ProfileID	Integer	This is the profile ID.
ExtractLineItems	Boolean	If set to FALSE, line items are not required for any document.
SkipForService	Boolean	If set to TRUE, line items are not mandatory for invoices relating to a service purchase order.
LineTotalOnlyFor Service	Boolean	If set to TRUE, only the line item total value is required for the invoices relating to a service purchase order.

Parameter	Type	Description
SkipForNoPO	Boolean	If set to TRUE, line items are not mandatory for no-PO invoices.
SkipForCredit	Boolean	If set to TRUE, line items are not mandatory for credit memos.
SkipForMIRA	Boolean	If set to TRUE, line items are not mandatory for invoices which are a 1-1 match with the total purchase order value or the value of all goods receipts not yet invoiced for the entire purchase order.
SkipForUnreleasedPO	Boolean	If set to TRUE, line items are not mandatory for invoices relating to a purchase order which has not yet been released.
SkipForInvalidPO	Boolean	If set to TRUE, the system does not require line items if the missing/invalid purchase order invalid reason has been selected by the user.
SkipForInvalidVendor	Boolean	If set to TRUE, and if the vendor-not-found invalid reason has been selected by the user, the system does not require line items.
SkipForUtilityVendor	Boolean	If set to TRUE, the system does not require line items if the vendor is a utility vendor. This saves considerable processing time if utility invoices are large documents with multiple backing sheets that the system does not need to evaluate as potential line items that require extraction.
NeverValidateLine Items	Boolean	If set to TRUE, line items may still be extracted, but no validation is carried out.
SkipForMultiLineLimits	Boolean	If set to TRUE, the system does not require line items for invoices relating to a purchase order that contains multiple line items, all of which are either of type limits (item category 1 or item category 9 where goods receipt based invoice verification has not been set). If set to FALSE, the system requires the user to enter line items (description and total only) for the above mentioned purchase order types.

10.2.22 BRWMSC

This table controls the identification and handling of miscellaneous charges, such as freight, customs charges, and fuel surcharges that may appear on an invoice document, both at header and at line item levels. You can set the parameters listed in the following table:

Parameter	Type	Description
ProfileID	Integer	This is the profile ID.
UnplannedThreshold	Numeric	This is the amount in the invoice currency which represents the maximum value the sum of all unplanned miscellaneous charges may reach before the system takes a course of action.
BlockIfOverThreshold	Boolean	If set to TRUE, if the total sum of all unplanned miscellaneous charges on the invoice exceeds the set threshold, the system sets a block on a created invoice document.
BlockCode	Freetext	This is the block code to be applied to an invoice for reasons of excessive unplanned miscellaneous charges.

Parameter	Type	Description
ThirdPartyFreightCode	Freetext	This identifies the miscellaneous charge category from which the rules for posting third party freight invoices must be lifted.
HandleMiscChargesForServices	Boolean	This indicates whether the miscellaneous charge processing logic must also be applied to invoices that relate to service purchase orders.
ValidateFromDB	Boolean	If set to TRUE, the system looks into a database table in order to determine the correct general ledger entry for a miscellaneous charge.
SQLConnectionGroup	Freetext	This is the SQL connection group specifying the miscellaneous charge database connection string as set in the SQL section. If no connection group is specified, the system uses group 01.
DBTableName	Freetext	This is the name of the database table containing the GL coding strings to be used for miscellaneous charges. This table must follow the layout of BWMiscAcc, which can be provided by the administrator.

10.2.23 BRWMSCCategory

This table is used to configure specific processing behavior for different types of miscellaneous charges. You can set the parameters listed in the following table:

Parameter	Type	Description
ProfileID	Integer	This is the profile ID.
IndexID	Integer	This is the miscellaneous charge index.
Type	Freetext	This is the miscellaneous charge type, such as FREIGHT or CUSTOMS.
Code	Freetext	This is the internal code for charge group.
HeaderField	Freetext	This is the name of the AP Packaged Project header field which contains miscellaneous charges belonging to the charge group.
Alias	Freetext	This is the comma-separated list of identifying strings so that the system can recognize miscellaneous charges specified at the line item level with the article description.
LineType	Freetext	This is the ERP system line type for the charge group.
AlwaysBookTo Unplanned	Boolean	If set to TRUE, the system handles all charges found on this document that are identified as belonging to the charge group as unplanned miscellaneous charges.
AlwaysBookTo Planned	Boolean	If set to TRUE, the system looks to book all miscellaneous charges identified as belonging to the charge group to a planned charge on the purchase order.
AlwaysBookToGL Account	Boolean	If set to TRUE, the system creates a general ledger account entry for all miscellaneous charges identified as belonging to the charge group.

Parameter	Type	Description
BookToUnplanned IfNoPlanned	Boolean	The system handles these charges as unplanned, if the following applies: • AlwaysBookToPlanned is set to TRUE • This setting is set to TRUE • No conditions or line types could be found on the purchase order, but charges belonging to this charge group were found on the invoice.
BookToGL AccountIfNoPlanned	Boolean	If AlwaysBookToPlanned is set to TRUE and no conditions or line types could be found on the purchase order, but charges belonging to this charge group were found on the invoice, the system creates a general ledger entry for those charges.
GLAccount	Freetext	This is the default general ledger account code to be used for miscellaneous charge general ledger entries.
GetCostObject FromPOLine	Boolean	This is the flag to denote whether the cost object information to complete the general ledger entry for the miscellaneous charge must be read from a paired purchase order line item. The cost object can either be a cost center, an internal order or a project, and the system reads this from the first paired invoice line which has one of these cost object assignments.
DefaultCostCenter	Freetext	This is the default cost center to be used for miscellaneous charge general ledger entries.
DefaultProfit Center	Freetext	This is the default profit center to be used for miscellaneous charge general ledger entries.
DefaultTaxCode	Freetext	This is the default tax code to be used for miscellaneous charge general ledger entries. For countries and ERP systems that use tax jurisdictions, the corresponding tax jurisdiction code is read from the first paired invoice line, which denotes the ship-to address for the goods.

10.2.24 BRWUOM

This table contains the database lookup parameters used for unit of measure conversion that may occur during line pairing. You can set the parameters listed in the following table:

Parameter	Type	Description
ProfileID	Integer	This is the profile ID.
SQLConnectionGroup	NN	This is the SQL connection group specifying the unit of measure conversion table database connection string as set in the SQL section. If no connection group is specified, the system uses group 01.
DBTableName	Freetext	This is the name of the unit of measure conversion database table. This setting is mandatory if a database lookup is to be performed for unit of measure conversion.
DBMaterialNo	Freetext	This is the technical name of the column in the unit of measure conversion table that represents the item material number.
DBExternalUOM	Freetext	This is the technical name of the column in the unit of measure conversion table that represents the external unit of measure read from the invoice from which conversion is to take place.
DBNumerator	Freetext	This is the technical name of the column in the unit of measure conversion table that represents the numerator component of the unit of measure conversion ratio.
DBDenominator	Freetext	This is the technical name of the column in the unit of measure conversion table that represents the denominator component of the unit of measure conversion ratio.
DBBaseUOM	Freetext	This is the technical name of the column in the unit of measure conversion table that represents the material base unit of measure, such as the destination unit of measure.

10.2.25 BRWUOMType

This table contains mapping information between a unit of measure how it might appear on an invoice to a unit of measure ISO-code as used by the downstream ERP system. You can set the parameters listed in the following table:

Parameter	Type	Description
ProfileID	Integer	This is the profile ID.
IndexID	Integer	This is the unit of measure index.
ISOCode	Freetext	This is the unit of measure ISO-Code/internal ERP system format for the UOM.
Alias	Freetext	This is the comma-separated list of aliases that the unit of measure may appear under on the invoice for this UOM.

10.2.26 BRWTOL

This table holds the tolerances used to validate the extracted amounts.

Tolerances are set in tolerance groups, which, in turn, can be assigned to specific currencies in the BRWCurrency table. Each group has a group code and a set of three tolerances controlling the permitted deviations at header level, at line item level and for the purposes of validating tax amounts. This is a global table that is independent of the client and profile ID.

You can set the parameters listed in the following table:

Parameter	Type	Description
IndexID	Integer	This is the tolerance group index.
HeaderTolerance	Number	This is the tolerance value for the invoice header amounts in the invoice currency.
TableRowTolerance	Number	This is the tolerance value for each individual line item in the invoice currency.
TaxTolerance	Number	This is the tolerance value for comparing the invoice tax amount with the system calculated tax amount based on the line item tax codes.
NoDecimalPlaces	Boolean	This indicates whether currencies assigned to this tolerance group are relevant for decimal places. This is appropriate for currencies where a single unit of that currency is the smallest currency unit in current use, for example, if there are no active cent or penny sub-units, and the Japanese Yen or the Hungarian Forint is used. If 10.400 is extracted and the currency is Hungarian Forints (HUF), and HUF is assigned to a tolerance group with this parameter set to TRUE, then the value is formatted to 10400. If the parameter is set to FALSE, it will be formatted to 10.40. However, if 10.40 is extracted, then the formatting stays at 10.40.

10.2.27 BRWIVR

This section contains the default setting associated with the invalid reason field. This is a global table that works independently of the client or profile. You can set the parameters listed in the following table:

Parameter	Type	Description
DefaultText	Freetext	This is the default invalid reason, such as NONE.
DefaultExportCode	Freetext	This is the export code associated with the default invalid reason, such as 0.

10.2.28 BRWIVRTexts

If the dynamic Verifier form is being used, this global table holds the field text that is presented to a user within the Verifier application for the Invalid Reason field. You can change the text elements in the DisplayText column to suit project requirements. You also can add new rows that correspond to an existing or custom IndexID in the BRWIVRType table to support translations for additional languages.

Parameter	Type	Description
TextID	Integer	This is the unique ID for the text element. This field is tied to the IndexID column in the BRWIVRType table, hence it is not possible to add a new TextID without a corresponding entry with the same ID in the BRWIVRType table.
LanguageID	Freetext	This is the two-character language ISO-code for the error message text. For example, EN is English, DE is German, and CN is Chinese. AP Packaged Project ships with English and Chinese languages pre-configured.
Display Text	Freetext	This is the display text for the language specified in the LanguageID column.

10.2.29 BRWIVRType

This table holds the invalid reasons that may be selected in Verifier. It is a global table that is independent of the client or profile. You can set the parameters listed in the following table:

Parameter	Type	Description
IndexID	Integer	This is the invalid reason index.
RuleID	Freetext	This is the rule ID for the invalid reason. The rule governs how Verifier behaves as a result of a particular invalid reason being selected. The following rules are available: • SETVENDORTOVALID sets the vendor field to valid. • SETPOTOVALID sets the purchase order number field to valid; no line pairing is carried out. • ALLOWNONPOVENDOR allows a vendor ID to pass even if it is unconnected to the purchase order as long as the vendor ID exists. • SETAMOUNTSTOVALID sets the amount fields and the table to valid; no line pairing is carried out. • THIRDPARTYFREIGHT sets the vendor number field to be valid as long as it exists and processes the document according to the third party freight rules during line pairing. • SETVENDORANDPOTOVALID sets the purchase order and vendor number fields to valid; no line pairing is carried out. • NONVATCOMPLIANT sets the local VAT amount, exchange rate and vendor & bill-to VAT registration number fields to valid. • STOCKINVOICE sets the purchase order number field to valid, line pairing is still carried out based on purchase orders set in UserExitLinePairingPOs. • ZEROVALUEINVOICE permits a zero total to pass in Verifier.
VerifierDisplay	Freetext	This is the invalid reason message displayed in Verifier.
ExportCode	Freetext	This is the invalid reason code exported by AP Packaged Project if the Invalid Reason field is set.

10.2.30 BRWERR

This table contains the list of error messages that may be displayed to a user in Verifier or written into the AP Packaged Project log file. It is a global table that is independent of the client or profile. You may add new error messages using error number 900 onwards.

Parameter	Type	Description
LanguageID	Freetext	This is the two-character language ISO-code for the error message text. For example, EN is English; DE is German; and CN is Chinese
Message	Freetext	This is the error message text.
ErrorNumber	Integer	This is an error message number.

10.2.31 BRWINF

This table controls setting associated with the Verifier information dialog boxes. It is a global table independent of the client or profile. You can set the parameters listed in the following table:

Parameter	Type	Description
DialogHeader	Freetext	This is the Information window title bar text.
DisableMIRAPopup	Boolean	If set to TRUE, this disables the window in Verifier informing the user that line items are not required because there is a 1-1 match on value between the invoice and the purchase order.
DisableCurrencyPopup	Boolean	If set to TRUE, and if the vendor or PO is changed, the system does not prompt the Verifier user if the currency is about to be over-written with a vendor or purchase order default currency.

10.2.32 BRWINFType

This table stores the information messages that appear in Verifier. It is a global table independent of the client or profile. You must change only the text associated with each message. You can set the parameters listed in the following table:

Parameter	Type	Description
InfNumber	Integer	This is the information message ID.
LanguageID	Freetext	This is the two-character language ISO-code for the information message text. For example, EN = is English, DE is German, and CN is Chinese. AP Packaged Project ships with English and Chinese languages pre-configured.
Message	Freetext	This is the information message for the message ID. Within the information message, text symbols [VEN] and [CUR] are used to represent the current vendor ID and currency respectively.

10.2.33 BRWEXP

This table holds configuration settings relating to the AP Packaged Project data export options. It is keyed upon the export profile ID, which can be assigned to individual clients. You can set the parameters listed in the following table:

Parameter	Type	Description
EXPProfileID	Integer	This is the export profile ID.
ProfileName	Freetext	This is the export profile name.
Description	Freetext	This is the description of export profile.
RedoAllExports	Boolean	If set to TRUE, the system carries out all export options that have been activated even if that export has been carried out before. For example, if four export options are activated, three are completed and the last one fails, then the document goes to status 750 denoting an export failure. If the flag is set to FALSE, upon retrying, only the failed export is carried out. If the flag is set to TRUE, then all four exports are performed again.
OutputStandard ResultsFile	Boolean	If set to TRUE, the system outputs a standard results file into the export directory. The file extension is set to .txt.
Filename	Freetext	This setting controls the name of the standard results file. If set to URN, this names the file according to the component of the image file name mapped in the IMP section. If left blank, or set to anything else, the file name is set to the same name as the document file name. The file extension is .txt.
DefaultExportPath	Freetext	This is the UNC path to the export directory which is used as the default must no export directory be set in RTS.
OutputTiffFile	Boolean	If set to TRUE, the system outputs a TIFF file of the document image in the export directory.
Tiffname	Freetext	This setting controls the name of the output TIFF file. If set to URN, this names the file according to the component of the image file name mapped in the IMP section. If left blank, or set to anything else, the file name is set to the same name as the document file name.
TiffDPI	Number	This specifies the DPI of the outputted tiff image, such as 300. The default TIFF resolution is 300 dpi.
TiffFormat	Freetext	This is the compression format of the outputted TIFF file. The following options are available: • G4FAX = Grade 4 compression • G3FAX = Grade 3 compression • LZWFAX = LZW Compression • HUFFAX = HUF Compression The default compression is G4FAX.
RedactInvoice Number	Boolean	If set to TRUE, the system redacts the invoice number on an outputted TIFF image.
OutputPDF	Boolean	If set to TRUE, the system outputs a searchable PDF file for each document.

Parameter	Type	Description
PDFName	Freetext	This setting controls the name of the output PDF file. If this is set to URN, it names the file according to the component of the image file name mapped in the IMP section. If left blank, or set to anything else, the file name is set to the same name as the document file name.
SendToDolphinWorkflow	Boolean	This is the flag to indicate whether the document must be sent to the Dolphin workflow.
CustomExport	Boolean	This is the flag to indicate whether a custom export must be carried out, as specified in the script user exit UserExitCustomExport.
ExportToDB	Boolean	This is the flag to denote whether the extracted data must be written to a database upon document export.
SQLConnectionGroup	NN	This is the SQL connection group specifying the export database connection string as set in the SQL section. If no connection group is specified, the system uses group 01. The header and line item tables must exist in the same database.
DBKey	Freetext	This is the key to be used for each new database record inserted into the header table. If this is set to URN, the portion of the file name mapped to the URN column in the IMP section is used. Any other value or a blank entry uses the entire document file name. Before inserting a new record, the system deletes any existing records with the same key.
DBHeaderTable	Freetext	This is the name of the table in the database where the header fields must be inserted.
DBHeaderKey	Freetext	This is the name of the column in the header export database table which represents the key field for the header record.
DBHeaderOperation	INSERT or UPDATE	This is the operation to be performed on the database record. If set to INSERT, the system inserts a new record. If set to UPDATE, the system updates an existing record with the same database key. There is no default setting.
DBLineItemsTable	Freetext	This is the name of the database table into which the invoice line item information must be written. The line item database export always occurs as an INSERT, and any existing lines for the same document record are deleted first.
DBLineItemsKey	Freetext	This is the name of the column in the line item export database table which, along with the header level key, represents the key field for each invoice line item record. This is typically set to the invoice line item number, such as 1,2,3, and so on.
DBGLItemsTable	Freetext	This is the name of the database table into which general ledger account line entries must be written.
DBGLItemsKey	Freetext	This is the name of the column in the general ledger export database table which, along with the header level key, represents the key field for each general ledger record. This is typically set to the general ledger item number, such as 1,2,3, and so on.
DBTaxTable	Freetext	This is the name of the database table into which VAT table line entries must be written.

Parameter	Type	Description
DBTaxKey	Freetext	This is the name of the column in the tax export database table which, along with the header level key, represents the key field for each tax line item record. This is typically set to tax item number, such as 1,2,3, and so on.
DBStatusExported	Freetext	This is the value or code that denotes a document successfully exported from AP Packaged Project.
OutputXMLFile	Boolean	If set to TRUE, the system outputs an XML file to the export directory configured on the RTS export instance. If no directory is configured, then the default export path parameter is used. If this is not configured, then the XML export fails and the batch is sent to status 750.
XMLFilename	URN [blank]	This setting controls the name of the XML output file. This can be set either to URN, which names the file according to the component of the image file name mapped in the IMP section. If left blank or set to anything else, the file name is set to the same name as the document file name.
XMLFileType	Freetext	This is the file extension applied to the XML file, such as XML = .XML; TXT = .txt. This is not required. If left blank, the file extension is XML by default.
XMLEncodingHeader	Freetext	This is the XML file coding header that forms the first line in the XML file. For example, setting the value as <xml version="1.0" encoding="UTF-16"?> produces an XML file that supports non-Western characters such as letters from the Russian, Greek and Chinese alphabets.
XMLFileHeader	Freetext	This denotes the value of the file header tag in the XML file, such as <MyFileHeader>. This value is Document by default if nothing else is set.
XMLInvoiceHeader	Freetext	This denotes the value of the tag marking the invoice header section in the XML file, such as <InvoiceHeader>. This value is InvHeader by default if nothing else is set.
XMLLineItemsHeader	Freetext	This denotes the value of the tag marking the line items section in the XML file, such as <LineItems>. This value is InvLines by default if nothing else is set.
XMLLineItemsTag	Freetext	This denotes the value of the tag marking each individual line item in the XML file, such as <LineItem>. This value is LINE by default if nothing else is set.
XMLGLLinesHeader	Freetext	This denotes the value of the tag marking the general ledger account line items section in the XML file, such as <GLLines>. This value is GLLines by default if nothing else is set.
XMLGLLinesTag	Freetext	This denotes the value of the tag marking each individual general ledger account line item in the XML file, such as <GLLines>. This value is GLLINE by default if nothing else is set.
XMLTaxHeader	Freetext	This denotes the value of the tag marking the tax line items section in the XML file, such as <VATLines>. This value is TaxLines by default if nothing else is set.

Parameter	Type	Description
XMLTaxTag	Freetext	This denotes the value of the tag marking each individual tax line in the XML file, such as <VATLine>. This value is TAXLINE by default if nothing else is set.
XMLStatusExported	Freetext	Value or code that denotes a document successfully exported from AP Packaged Project. This value is subsequently placed as the output against the XMLStatus field.
OutputCSVFile	Boolean	This is the master switch for all CSV file output. If set to FALSE, the system does not output any configured CSV files in the BRWCSV table, irrespective of whether the local switch for a CSV file group is set to TRUE.
DeliveryNoteSeparator	Freetext	This is the single character separator for the output of multiple delivery note numbers if the delivery notes table is activated and column ExtractDeliveryNotesIntoTable is set to TRUE in the BRWNUM table. This applies to database, XML and CSV file output.
SendToProcessIT	Boolean	Flag to indicate whether the document must be exported using XML to the ProcessIT interface table.
PITSQLConnectionGroup	NN	This is the SQL connection group specifying the ProcessIT database connection string as set in the SQL section. If no connection group is specified, the system uses group 01.
PITDBTable	Freetext	This is the name of the ProcessIT interface table where the output XML file must be written.
OutputPITXML	Boolean	Flag to indicate whether the Process IT export XML file must be written to the output export directory.
OutputPITRDYFile	Boolean	This flag indicates whether the XML file output must be accompanied by a ready file to indicate that the XML has been written out fully. The ready file has an RDY extension and the same file name as the XML file.
PITXMLFileType	Freetext	This is the file extension to be added to the ProcessIT output XML file.
PITKey	URN [Blank]	This is the key used for the Process IT export. It controls the way that output XML file names are named and the manner in which the SOURCE_ID column is populated in the ProcessIT interface table in the Oracle database. If this is set to URN, the portion of the image file name mapped to the URN column in the IMP section is used. Any other value or a blank entry uses the entire document file name.
PITDocType	Freetext	This is the value that is to be passed as the ProcessIT XML DOCTYPE property – if it is left blank, the client name is used.
PITInvoice	Freetext	This is the value that is passed to the ProcessIT equivalent of the document type field if the document is an invoice.
PITCredit	Freetext	This is the value that is passed to the ProcessIT equivalent of the document type field if the document is a credit note.
PITSourceName	Freetext	This is the value to be written to the SOURCE column in the ProcessIT interface table. If left blank, the value is PERCEPTIVE by default.

Parameter	Type	Description
PITHeaderXML1	Freetext	This is the value that is used as the first header in the PIT XML file.
PITHeaderXML2	Freetext	This is the value that is used as the second header in the PIT XML file. This header must begin with an <ARCDOC> label or export fails.
PITIncludeFieldPositions	Boolean	Flag to indicate whether the ProcessIT XML file must include positional information, candidate confidence values and field validity statuses where available for header and line item fields.
ProcessDirectorFunction	Freetext	This is the name of the RFC function module that the system calls at the time of document export to send the invoice data to Process Director.
UseGenericPDFunction	Boolean	This flag indicates whether the integration to Process Director occurs using the Process Director generic function module.
GenericPDFunction	Freetext	This is the name of the Process Director generic function module that is called.
GenericMappingID	Freetext	This is the mapping schema used by the Process Director generic function module.
LateArchiveViaPD	Boolean	This flag indicates whether late archiving must occur using the generic Process Director function module.
PDLateArchiveAsPDF	Boolean	This flag is set to TRUE if the document is converted to PDF for late archiving.
PDFTempDirectory	Text	This is the temporary export directory used by the system to convert a document to PDF for late archiving via Process Director.
OutputOCRXMLFile	Boolean	This flag is set to TRUE if the output of the OCR XML file is required.
OCRXMLFileKey	URN [Blank]	This is the key used for the OCR XML export. It controls the way that output XML file names are named and the manner in which the SOURCE_ID column is populated in the ProcessIT interface table in the Oracle database. If this is set to URN, the part of the image file name mapped to the URN column in the IMP section is used. Any other value or a blank entry uses the entire document file name.
OCRXMLFileType	Freetext	This is the file extension to be added to the OCR XML file. If no file type is specified, a default of .xml is used.
OCRXMLHeading	Freetext	This is the value that is used as the header line in the OCR XML file.
IncludeCandidateInfo	Boolean	This flag indicates whether candidate information is written into the OCR XML file for each of the extraction fields.

10.2.34 BRWEXPHeader

This table is used to map the AP Packaged Project header export fields into fields in the XML file or columns in a database. You can set the parameters listed in the following table:

Parameter	Type	Description
EXPProfileID	Integer	This is the export profile ID.
FieldName	Freetext	This is the name of the field.
XMLTag	Freetext	This column represents the tag that is used for the field in an exported XML file. If left blank, the field is not exported.
DBColumnName	Freetext	This is the technical name of the target field in the export database.
ProcessITTag	Freetext	This column represents the tag that must be used for the field in an exported XML file for ProcessIT, whether written out to a directory or sent to the ProcessIT interface tables. If left blank, the field is not exported.

10.2.35 BRWEXPLines

This table is used to map the AP Packaged Project line item fields in the XML file or columns in a database. You can set the parameters listed in the following table:

Parameter	Type	Description
EXPProfileID	Integer	This is the export profile ID.
FieldName	Freetext	This is the name of the field.
XMLTag	Freetext	This column represents the tag that is used for the field in an exported XML file. If left blank, the field is not exported.
DBColumnName	Freetext	This is the technical name of the target field in the export database.
ProcessITTag	Freetext	This column represents the tag that is used for a field in an exported XML file for ProcessIT, either written out to a directory or sent to the ProcessIT interface tables. If left blank, the field is not exported.

10.2.36 BRWEXPTax

This table is used to map the AP Packaged Project tax line item fields into fields in the XML file or columns in a database. You can set the parameters listed in the following table:

Parameter	Type	Description
EXPProfileID	Integer	This is the export profile ID.
FieldName	Freetext	This is the name of the field.
XMLTag	Freetext	This column represents the tag that is used for a field in an exported XML file. If left blank, the field is not exported.

Parameter	Type	Description
DBColumnName	Freetext	This is the technical name of the target field in the export database.
ProcessITTag	Freetext	This column represents the tag that is used for a field in an exported XML file for ProcessIT, whether written out to a directory, or sent to the ProcessIT interface tables. If left blank, the field is not exported.

10.2.37 BRWEXPGL

This table is used to map the AP Packaged Project general ledger line item fields into fields in the XML file or columns in a database. You can set the parameters listed in the following table:

Parameter	Type	Description
EXPProfileID	Integer	This is the export profile ID.
FieldName	Freetext	This is the name of the field.
XMLTag	Freetext	This column represents the tag that is used for the field in an exported XML file. If left blank, the field is not exported.
DBColumnName	Freetext	This is the technical name of the target field in the export database.

10.2.38 BRWLPR

This table contains the settings that are used by the system when carrying out the line pairing operation at the point of document export. You can set the parameters listed in the following table:

Parameter	Type	Description
ProfileID	Integer	This is the profile ID.
DoLinePairing	Boolean	This is the flag to indicate whether the line pairing operation must be carried out at the point of document export.
DoLinePairingFor Service	Boolean	This is the flag to indicate whether line pairing must be carried out for invoices that relate to a service purchase order.
CheckForMultiplePOs	Boolean	This is the flag to indicate whether the system must consider multiple purchase order numbers on the document at the time of line pairing.

Parameter	Type	Description
DescriptionThreshold	Number	This value expressed as a percentage, such as 30 = 30%, denotes how confident the system needs to be to pair a line item based on the fuzzy match on description. A setting of 100 requires an exact match between the invoice line and purchase order line descriptions. For example, if the PO line item description was BROWN HATS, then a 100% match would be achieved if the invoice line item description was BROWN HATS, brown hats, BROWN or HAT. For information purposes, BROWN HAT would be a 99.99% match and HATS BROWN would be an 85% match. A setting of zero means the closest match to the invoice line must be considered as long as there is some degree of similarity. The default value if nothing or an invalid entry is present is 0. We recommend setting this value to 30. The invoice line item description must be at least 5 characters in length for the system to consider it when selecting a purchase order line item.
DescriptionDistance	Number	This value expressed as a percentage, such as 10 = 10%, denotes the minimum confidence distance between the best and second best possibility for the fuzzy match on the description. For example, if this value is set to 10% and the system is 51% sure that line 1 is the right result, but 45% sure that line 2 is the right result, then the line does not pair as the distance between them (6%) is less than the minimum confidence distance. The default value is 0%. We recommend setting this value to 10.
DescriptionTolerance	Number	This denotes the percentage tolerance with which the invoice unit price is allowed to deviate from the purchase order unit price to allow the lines to pair if the pairing was via a fuzzy match on the description. If left blank, this additional check is not carried out.
LimitsDescThreshold	Number	This value expressed as a percentage, such as 30 = 30%, denotes how confident the system needs to be to pair a service line item based on a fuzzy match on description. A setting of 100 requires an exact match between the invoice line and purchase order line descriptions. A setting of zero means the closest match to the invoice line must be considered as long as there is some degree of similarity. The default value if nothing or an invalid entry is present is 0. We recommend setting this value to 50. The invoice line item description must be at least 5 characters in length for the system to consider it when selecting a purchase order line item.

Parameter	Type	Description
LimitsDescDistance	Number	This value expressed as a percentage, such as 10 = 10%, denotes the minimum confidence distance between the best and second best possibility for the fuzzy match on the service line item description. For example, if this value is set to 10% and the system is 51% sure that line 1 is the right result, but 45% sure that line 2 is the right result, then the line does not pair as the distance between them (6%) is less than the minimum confidence distance. The default value is 0%. We recommend setting this value to 10.
LimitsDescTolerance	Number	This denotes the percentage tolerance with which the aggregate total value of assigned invoice lines may exceed the original purchase order line item order total. For example, if the original purchase order line is raised with a total of 10,000 and the tolerance is set to 10 (i.e. 10%) then the system stops pairing invoice lines to this purchase order line as soon as the total value booked to date exceeds 11,000. If left blank, this check is not carried out.
UOMCheck	Boolean	If set to TRUE, the system takes into account any possible unit of measure differences when comparing the invoice quantity to the purchase order quantity.
PUOMCheck	Boolean	If set to TRUE, the system applies conversions to the extracted invoice quantity based on ratios read from the purchase order if the PO order unit of measure differs from the PO order price unit of measure, but the invoice unit of measure and the purchase order price unit of measure are the same.
ConvertQuantityFromDB	Boolean	If set to TRUE, the system performs a lookup of the unit of measure conversion table specified in the BRWUOM table if the unit of measure captured on the invoice line does not correspond to the order unit of measure on the PO line that it has been paired to, and the conversion ratio cannot be calculated mathematically based upon the PO line item details and the PO history. If the database lookup is not configured correctly, or the relevant entry is missing, the line item is not paired.
PairToSingleGR	Boolean	If set to TRUE, the system only pairs the invoice line to a single goods receipt line if the corresponding purchase order line is set for goods receipt based invoice verification.
FindGRWithDelivery Number	Boolean	If set to TRUE, the system considers the external delivery note number set against a goods receipt line in the ERP system as a priority when pairing an invoice line to a goods receipt line if the corresponding purchase order line is set for goods receipt based invoice verification.

Parameter	Type	Description
OnlyUseDeliveryNumberTo FindGR	Boolean	This parameter is used in conjunction with the above FindGRWithDeliveryNumber parameter. If set to TRUE, the system only pairs an invoice line against a goods receipt line if the external delivery note number against the goods receipt in the ERP system can be found on the invoice. If the goods receipt in the ERP system has no external delivery note number set against it, the line is not paired. For this feature to function, the FindGRWithDelivery parameter must also be set to TRUE.
RequirePODetails ForMultiMatrl	Boolean	This is a flag to denote whether, in the event that the pool of purchase order line items contains more than one line for the same material (as denoted by an identical material number on the purchase order), the system has to find a referenced purchase order number and (optionally) a purchase order line item number on the document against the invoice line so that the system knows which purchase order line to pair the invoice line to. If this flag is set to TRUE, and no such PO details are found on the invoice, then the line item is not paired.
IgnoreCompletedPOLines	Boolean	If set to TRUE, during the line pairing process, the system does not consider any purchase order lines that have already been fully invoiced. Fully invoiced is defined as the quantity invoiced to date being equal or greater than the quantity originally ordered.
ActivateSubDebCheck	Boolean	If set to TRUE, the system sets the subsequent debit flag to X during line pairing for an invoice line that is paired to a purchase order line. In this purchase order line the quantity that has been invoiced to date is exactly equal to the quantity that has been delivered to date. That quantity is greater than zero and the invoice unit price adjustment is less than half of the original unit price on the purchase order.
EnableIntegrityCheck	Boolean	If set to TRUE, when carrying out line pairing against purchase orders that have multiple open lines for the same material or have lines set for goods receipt-based invoice verification where multiple open goods receipts exist, the system does not book the invoice line to the first open PO line/goods receipt. This does not apply in instances where a referenced purchase order/line at the line item level, a delivery note number, or values and quantities can distinguish the correct PO line/goods receipt to book the invoice line to. This parameter must always be set to YES in implementations where a workflow component exists between AP Packaged Project and the downstream ERP system, and AP Packaged Project books directly to the ERP system, but does not create fully posted invoices. As the system dynamically reads the live purchase order history information from the ERP system during line pairing, if two invoices appear in quick succession referencing the same purchase order, both can be paired to the same PO line/goods receipt. The ERP PO history could not be updated subsequent to processing the first invoice if it was not posted immediately. This would otherwise make the PO line/goods receipt unavailable for the second invoice.

Parameter	Type	Description
ExcludeFreightFromMIRA Process	Boolean	If set to TRUE, the value of any miscellaneous charges on the invoice is not included in a calculation to see whether there's a one-to-one relationship between the invoice and purchase order.
MultiPairingToSinglePOLine	Boolean	If set to TRUE, the system is able to pair multiple lines on the invoice for the same material to a single line on the purchase order.
ActivateLog	Boolean	If set to TRUE, the system writes out a log file containing trace entries for the line pairing operation as well as for the automatic tax code determination procedure. This is written into the standard AP Packaged Project log file.
PUOMTolerance	Number	This is the percentage with which the PO unit price must be within the invoice unit price to infer the same order price unit of measure. If left blank, the system looks for an exact match. This is only considered if UOMCheck is set to TRUE. The recommended setting is 10.
UnitPriceTolerance	Number	This activates an extra check for invoice lines that pair to purchase order lines due to a high correlation on the amount fields. If set to 10, the system does not pair an invoice line to a purchase order line if the purchase order has another line item that has pricing within 10% of the purchase order line originally chosen. If set to zero, no extra check is carried out.
ReadPOLinesViaUserExit	Boolean	This flag must be set to TRUE if the purchase order line item details must be retrieved via a custom lookup in UserExitReadPODetails.
GetPOLinesFromDB	Boolean	This is the flag to denote whether the purchase order line items must be read from a database.
SQLConnectionGroup	NN	This is the SQL connection group specifying the purchase order database connection string as set in the SQL section. If no connection group is specified, the system uses group 01.
DBTableName	Freetext	This is the name of the database table containing the purchase order line item information.
UseStoredProcedure	Boolean	If set to TRUE, the system calls a stored procedure in order to retrieve the purchase order line item information from the database.
StoredProcedureName	Freetext	This is the name of the stored procedure to be used in order to retrieve the purchase order lines from the database.
StoredProcedure Parameters	Freetext	This is the comma-separated list of the parameters that must be used when calling the stored procedure above. These parameters must tie in to entries in the BRWSPC table.

Parameter	Type	Description
DBPARTITIONID	Freetext	This is the technical name of the purchase order line item database table column which holds the record partition ID. If the UsePOPartition column is set to TRUE in the BRWPON table and the system has been set up to read purchase order line items, this value must be populated or the system logs an error.
DBPO	Freetext	This is the name of the column in the purchase order line database table which holds the purchase order number.
DBLine	Freetext	This is the name of the column in the purchase order line database table which holds the purchase order line number.
DBMaterialNo	Freetext	This is the name of the column in the purchase order line database table which holds the purchase order line item material number.
DBMaterialGroup	Freetext	This is the name of the column in the purchase order line database table which holds the material group to which the purchase order line item belongs.
DBDescription	Freetext	This is the name of the column in the purchase order line database table which holds the purchase order line item description.
DBPOQuantity	Freetext	This is the name of the column in the purchase order line database table which holds the order quantity for the purchase order line item.
DBUnitPrice	Freetext	This is the name of the column in the purchase order line database table which holds the unit price for the purchase order line item.
DBPOTotal	Freetext	This is the name of the column in the purchase order line database table which holds the overall total for the purchase order line item.
DBTaxCode	Freetext	This is the name of the column in the purchase order line database table which holds the tax code for the purchase order line item.
DBTaxJurCode	Freetext	This is the name of the column in the purchase order line database table which holds the tax jurisdiction code for the purchase order line item.
DBUOM	Freetext	This is the name of the column in the purchase order line database table which holds the order unit of measure for the purchase order line item.
DBPriceUnit	Freetext	This is the name of the column in the purchase order line database table which holds the purchase order line item price unit.
DBPUOM	Freetext	This is the name of the column in the purchase order line database table which holds the order price unit of measure for the purchase order line item.

Parameter	Type	Description
DBTotalQuantityDelivered	Freetext	This is the name of the column in the purchase order line database table which holds the total quantity delivered to date for the purchase order line item.
DBTotalValueDelivered	Freetext	This is the name of the column in the purchase order line database table which holds the total value delivered to date for the purchase order line item.
DBTotalQuantityInvoiced	Freetext	This is the name of the column in the purchase order line database table which holds the total quantity invoiced to date for the purchase order line item.
DBTotalValueInvoiced	Freetext	This is the name of the column in the purchase order line database table which holds the total value invoiced to date for the purchase order line item.
DBItemCategory	Freetext	This is the name of the column in the purchase order line database table which holds the item category/line type of the purchase order line item.
DBPlant	Freetext	This is the location code for the ship-to address to which the goods were delivered or where a service was performed.
DBChargeCode	Freetext	This is the name of the column in the purchase order line database table which holds the charge code for the purchase order line item.
DBChargeCodeID	Freetext	This is the name of the column in the purchase order line database table which holds the charge code ID for the purchase order line item.
DBCompanyCode	Freetext	This is the name of the column in the purchase order line database table which holds the company code for the purchase order line item.
DBERPPOType	Freetext	This is the name of the column in the purchase order line database table which holds the JD Edward purchase order type for the purchase order line item.
DBBusinessUnit	Freetext	This is the name of the column in the purchase order line database table which holds the PeopleSoft purchasing business unit for the purchase order line item.
DBERS	Freetext	This is the name of the column in the purchase order line database table which holds a flag which indicates whether this purchase order is intended for the Evaluated Receipt Settlement (ERS) process.

10.2.39 BRWPMT

This table contains settings that relate to the payment method and its relationship to the requirement for a bank account. You can set the parameters listed in the following table:

Parameter	Type	Description
ProfileID	Integer	This is the profile ID.

Parameter	Type	Description
BankMethods	Freetext	This is the comma-separated list of payment methods that denote a bank transfer and hence the requirement for a bank account.
AccountCurrency NotRequired	Boolean	If this flag is set to TRUE', the system does not require a bank account currency in order to select a bank account number.However bank accounts in the vendor master that do contain a currency that matches the currency of the invoice take priority in the bank account selection.

10.2.40 BRWCTR

This table contains settings whereby the system can be pointed to a lookup table for countries to determine their local currencies and whether they are part of the EU or not. This is used in the automatic tax code determination procedure.

This is a global table that is independent of the client and profile. The table must only ever have one row.

You can set the parameters listed in the following table:

Parameter	Type	Description
ValidateFromDB	Boolean	If this is set to TRUE, the country is checked against a database.
SQLConnectionGroup	NN	This is the SQL connection group specifying the country database connection string as set in the SQL section. If no connection group is specified, the system uses group 01.
DBTableName	Freetext	This is the name of the country database table. This setting is mandatory if a database lookup is to be performed.
DBCountry	Freetext	This is the name of the column in the country database table that holds the country key.
DBCurrency	Freetext	This is the name of the column in the country database table that holds the currency for the country.
DBEUMember	Freetext	This is the name of the column in the country database table that denotes whether the country is an EU state member or not. X denotes an EU state member.
DBName	Freetext	This is the name of the column in the country database table that holds the full name of the country in English.
DBVATRates	Freetext	This is the name of the column in the country database table that holds the list of valid VAT rates for the country. The list is comma-separated.

10.2.41 BRWMAT

This table contains settings whereby customer material number formats are defined.

During line item extraction, the system strives to find a customer material number against the invoice line item. This custom material number can subsequently be used downstream in the line pairing operation as invoice lines are reconciled to their purchase order counterparts. If no formats are defined, this does not mean that a material number is never extracted, but that it increases the chance of successful recognition.

You can set the parameters listed in the following table:

Parameter	Type	Description
ProfileID	Integer	This is the profile ID.
IndexID	Integer	This is the material index.
Format	Freetext	This is the format string for a possible material number. # is a wildcard character representing any number and @ represents any alpha character. The format string entered is used to help the system determine the correct customer material number. For example, if 5###### was specified as a format string, then the system would look for a seven-digit material number beginning with 5.
Ignore	Freetext	This is the list of characters that may appear in a material number in any position, such as a hyphen or a period/full stop, that the system must be tolerant of in the corresponding format string specified. This list does not need to be comma separated.

10.2.42 BRWCSV

This table contains settings that control the CSV file output formats. The system can be configured to output multiple CSV files, either per document or per AP Packaged Project batch. Each row in the table represents the configuration behind a single CSV file.

You can set the parameters listed in the following table:

Parameter	Type	Description
EXPProfileID	Integer	This is the export profile ID.
IndexID	Integer	This is the CSV file index.
OutputFile	Boolean	This is the local switch for the CSV file group. If set to TRUE, the system attempts to output a CSV file according to the configuration settings specified.
CombinedFilePerBatch	Boolean	This indicates whether one CSV file-per-document or one CSV file-per-AP Packaged Project batch is required. If output is required on a per batch basis, the output file is created with a .TMP extension until the last document in the batch is exported successfully. It is not possible to output a combined file-per-batch unless the RTS instance carrying out the document import has a document grouping setting of either 1 folder-per-batch or 1 batch-per-subdirectory.

Parameter	Type	Description
Filepath	Freetext	This parameter allows the configuration of an alternate export directory for the CSV file group. If left blank, the main export directory set against the RTS instance carrying out document export is used. If no export directory is configured against the RTS instance, the default export directory configured in the BRWEXP table is used. If this is also blank, or a specified directory does not exist, the CSV file export fails.
Filename	URN [blank]	This is the naming convention for the CSV file if it is to be outputted on a document by document basis. This can be set either to URN, which names the file according to the component of the image file name mapped in the IMP section. If left blank or set to anything else, the file name is set to the same name as the document file name.
FileType	Freetext	This is the file extension for the output file. If left blank, .CSV is used as the file extension.
FilePrefix	Freetext	This is the file prefix for the output file.
Separator	Freetext	This is the separator to be used in the CSV file line. Based on user input here, the system automatically cleanses any extracted data using this separator in order to maintain a consistent number of columns in the output. Illegal separators are a period (.), a forward slash (/) and a backslash (\). The chosen separator must still be entered manually in the format configuration for each line in the CSV file.
DateFormat	DDMMYYYY MMDDYYYY YYYYMMDD	This is the desired output format for date fields. If no entry is made, the system considers the output date format configured in the BRWDAT table. If no entry is made there, the default output format of DDMMYYYY is used.
DateSeparator	Freetext	This is the separator to be used in conjunction with the date format setting above. For example, if the date format above is MMDDYYYY and a hyphen (-) is entered as the separator in this parameter, and the date is 2nd November 2009, then 11-02-2009 is the output.
InvoiceType	PO NPO	This is the filter applied to the CSV file so that output can be controlled by the Value Invoice Type field. If this value is set to PO, the CSV file is only written if the invoice type is PO. If this value is set to NPO, the CSV file is only written if the invoice type is NO-PO.
OutputImage	Boolean	Setting this to TRUE outputs the original image document (retaining the original image file name and file type) into the same directory as the CSV file.

Parameter	Type	Description
FormatLineN	Freetext	This parameter controls the format of each row to be outputted in the CSV file. Up to five rows can be outputted per CSV file. To output the TIFF name, the invoice number (1234) and the vendor number (ABC) separated by a colon, the setting must contain the following: <%TNM>:<%INO>:<%VID> Which would produce the following: myFile.tif:1234:ABC Additional text can also be included in the format. For example: File=<%TNM> | Invoice Number=<%INO> | Vendor ID=<%VID> would produce the following. File=myFile.tif | Invoice Number=1234 | Vendor ID=ABC The columns are represented by the following symbols: ' <%TNM> The name of the original imported file (no file path) ' <%TNF> The name of the original imported file (no file path or file extension) ' <%TND> The name of the original imported file with export directory file path ' <%SDT> The scan date in the format YYYY-MM-DD ' <%BNM> The batch name ' <%EID> The employee ID

Parameter	Type	Description
		' <%EFN> The employee first name ' <%ELN> The employee last name ' <%DTP> The document type, such as INVOICE or CREDIT ' <%ITP> The invoice type, such as PO or NO-PO ' <%IDT> The invoice date ' <%INO> The invoice number ' <%TAX> The invoice tax amount ' <%WTX> The invoice withholding tax amount ' <%DCT> The invoice discount amount ' <%MSC> The invoice misc charge amount ' <%CUR> The invoice currency ' <%TOT> The invoice total ' <%PST> The PST/QST amount ‘ <%HST> The HST amount ' <%ICM> The ICMS tax amount ' <%PON> The PO Number ' <%URN> The unique Reference Number ' <%VID> The vendor ID ' <%IVD> The internal Vendor ID ' <%SID> The site ID ' <%VNM> The vendor name ' <%BTO> The bill-to name ' <%CCO> The company code ' <%POR> The Payment Order Reference Number (POR) ' <%PSN> The Payment Order Subscriber Number ' <%KID> The Payment Reference ' <%ACC> The account Number ' <%BAC> The bank Account Number

Parameter	Type	Description
		' <%BCD> The bank Account Code ' <%PRI> The priority Flag ' <%EXC> The exchange Rate ' <%IVR> The invalid Reason ' <%ICD> The invalid Reason Code ' <%LNK> The ocument Link ' <%ERP> The ERP document key ' <%EPT> The ERP system PO type ' <%BSU> The business unit ' <%DEL> The delivery note ‘ <%DLD> The delivery date ‘ <%DUE> The due date ‘ <%ISR> The ISR retention amount ‘ <%IBN> The IBAN number ‘ <%BIC> The BIC / Swift code ‘ <%REF> Your ref ‘ <%CNC> Mainland China VAT invoice code ‘ <%CNP> Mainland China VAT invoice password ‘ <%VVT> Vendor VAT registration number ‘ <%BVT> Bill-to VAT registration number ‘ <%MXU> Mexican UUID number

Parameter	Type	Description
LineItem	Freetext	This parameter controls the format of line item entry to be output in the CSV file. If left blank, no line item detail is outputted. One row is added into the CSV file for each line item. To output the PO number, the PO line item number and the total, this parameter must be set as follows: <%LPO><%LPL><%LTO> The available literals are as follows: ' <%LNO> The invoice line item number ' <%LPO> The PO number ' <%LPL> The PO line item ' <%LDS> The PO line description ' <%LMN> The material number ' <%LMG> The material group ' <%LQT> The quantity ' <%LUM> The order unit of measure ' <%LUP> The unit price ' <%LPU> The order price unit of measure ' <%LQU> The quantity in order price unit of measure ' <%LTO> The line item total ' <%LTC> The tax code ' <%LTJ> The tax jurisdiction code ' <%LFV> The freight vendor ID ' <%LGN> The goods receipt document number ' <%LGY> The goods receipt document year ' <%LGI> The goods receipt document item number ' <%LSN> The service entry sheet number
		' <%LSI> The service entry sheet item number ' <%LSD> The subsequent debit/credit indicator ' <%LLT> The line type ' <%LCD> The charge code ' <%LCI> The charge code ID ' <%LDL> The AP Packaged Project line item ' <%LPT> The plant ' <%LTY> The ERP purchase order type ' <%LBU> The ERP purchasing business unit ' <%LCC> The company Code ‘ <%LTR> The VAT/tax rate

10.2.43 BRWSPC

The BRWSPC table is where parameters to be used in the stored procedure lookups are defined. A row in the table represents a single parameter. This is a global table which is independent of the client and profile ID. You can set the parameters listed in the following table:

Parameter	Type	Description
IndexID	Integer	This is the stored procedure parameter index.
ParameterName	Freetext	This is the name of the stored procedure parameter. This value is mandatory for each stored procedure parameter.
ParameterType	Freetext	This is the stored procedure parameter type. Possible options are BOOLEAN, INT, DATE, DOUBLE, and VARCHAR. For any other entries (including a blank entry), a type of UNKNOWN is used.
ParameterSize	Freetext	If the parameter type is VARCHAR, this setting specifies the maximum length allowed. If this setting is left blank or contains either zero or a non-numeric entry, a default size of 50 is used.
ParameterValue	Freetext	This is where the value of the parameter is defined. Following are two options: Specify the technical name of a AP Packaged Project field (e.g. PONumber, CompanyCode) – this is case-sensitive. Specify a hard value to be passed.
ParameterDirection	Freetext	If set to I, the parameter direction is set to input. For all other entries (including a blank entry), the parameter direction is set to output.

10.2.44 BRWClient

This table is where clients are set-up and configured.

Parameter	Type	Description
ClientID	Integer	This is the unique ID of the client, which must always be set to an integer value.
ProfileID	Integer	This is the ID of the profile assigned to the client. The profile controls what fields are extracted and how they are validated. More than one client may share the same profile ID if the extraction and validation requirements are identical.
ExportProfileID	Integer	This is the ID of the export profile assigned to the client. The export profile ID controls how data is exported for a client. More than one client may share the same export profile ID if the export requirements are identical.
ClientName	Freetext	This is a free text string containing the name of the client. This data is written into the reporting database for each document assigned to a client.
InstructionsProfileID	Integer	AP Packaged Project includes a button on the dynamic verifier form which when pressed will display specific processing instructions for a particular client. The instructions profile ID is the ID assigned to a particular set of instructions that are held in table.
ForceVerify	Boolean	This is a flag value which controls whether all documents for this client must stop in Verifier. If set to TRUE, all documents stop. If set to FALSE, only documents requiring user attention stop.

Parameter	Type	Description
ClientGroup	Integer	This is the ID of the client group to which the client belongs. It is an integer value that can be set freely by the system administrator. The client group is the means by which users are assigned to have access to documents belonging to specific clients.
RequiresReview	Boolean	This is a Boolean flag which indicates whether documents assigned to the client must always be subject to review post verification.
VendorPartition	Integer	This is the ID of the vendor master data partition to be used by the client. Entries in this column must correspond to an entry in the BRWVNDPartition table.
EmployeePartition	Integer	This is the ID of the employee master data partition to be used by the client. Entries in this column must correspond to an entry in the BRWEMPPartition table.
POPartition	Integer	This is the ID of the purchase order data partition to be used by the client. Entries in this column must correspond to an entry in the BRWPONPartition table.
TaxPartition	Integer	This is the ID of the tax partition used by the client during automatic tax code determination where the standard OFRTAX table is used. Entries in this table must correspond to an entry in the BRWTAXPartition table.
Priority	Integer	When documents are imported into AP Packaged Project, they are placed into batches and each batch is assigned a priority. This priority controls the order by which the runtime server component of AP Packaged Project will process the batches, and also the order in which the documents appear in the Verifier application. The priority scale runs from 1 to 9, with 1 having the highest level of priority. If this field is populated with 1, it means that all batches containing documents from this client are accorded a priority of 1.

10.2.45 BRWProfile

This table contains a list of profiles within the project, along with a description. A profile must be registered within this table before it can be assigned to a client. You can set the parameters listed in the following table:

Parameter	Type	Description
ProfileID	Integer	This is the unique ID of the client, which must always be set to an integer value.
ProfileName	Freetext	This is the name of the profile.
ProfileDescription	Freetext	This is the description of the profile.

10.2.46 BRWTexts

This table holds the text elements that display within the Verifier application for standard system search dialog boxes, message boxes, and class names. You are not allowed to make changes to the table. However, you can add new rows referencing an existing TextID to support translations for additional languages.

Parameter	Type	Description
TextID	Integer	This is the unique ID for the text element.
LanguageID	Freetext	This is the two-character language ISO-code for the error message text. For example, EN is English, DE is German, and CN is Chinese. AP Packaged Project ships with English and Chinese languages pre-configured.
Message	Freetext	This is the display text for the language specified in the LanguageID column.

10.2.47 BRWVNDPartition

This table contains a list of the vendor partitions active within the project. A vendor partition must be registered within this table before it can be assigned to a client. You can set the parameters listed in the following table:

Parameter	Type	Description
VendorPartition	Integer	This is the unique ID of the vendor partition.
Description	Freetext	This is the description of the partition.

10.2.48 BRWEMPPartition

This table contains a list of the employee partitions active within the project. An employee partition must be registered within this table before it can be assigned to a client. You can set the parameters listed in the following table:

Parameter	Type	Description
EmployeePartition	Integer	This is the unique ID of the employee partition.
Description	Freetext	This is the description of the partition.

10.2.49 BRWUser

The BRWUser table contains a list of active users within the system, along with their corresponding authorizations. The table is keyed on a combination of a unique user name and client group. You can set the parameters listed in the following table:

Parameter	Type	Description
UserID	Freetext	This is the AP Packaged Project user ID.
ClientGroup	Integer	This is the client group to which the user has been assigned.

Parameter	Type	Description
AuthorityLevel	@@@	This is the standard AP Packaged Project role assigned to the user. VER. The standard Verifier user. SET. The standard Verifier user with permission to change verifier settings SLV. The SET + ability to use the supervised learning function. SLM. The SLV + ability to review and promote vendor learnsets to the global project. ADM = Administrator All users have the standard filter role.
RequiresReview	Boolean	If set to TRUE, all documents verified by the user move to a review state for quality control.
Domain	Freetext	This is the user Windows domain for Windows based authentication.
EncryptedPassword	Freetext	This is an encrypted password for the user. For more information about the generation of encrypted passwords, refer to Appendix R. If you use an encrypted password, leave the Password column blank as this column always takes priority.
PrimaryGroupName	Freetext	This is the Verifier user primary group name (for example, Admin). Use this name to associate a user to a predefined group of Verifier settings. If the primary group name does not exist, the system creates it automatically.
LanguageID	Freetext	This is the Verifier language preference for the dynamic verifier. Use the two-character ISO-code for the language, for example, EN for English or CN for Chinese.
Password	Freetext	If Windows authentication is not being used, this contains a string for the user password.

10.2.50 BRWFLD

This table controls which fields are activated for a given profile along with their corresponding types and validation settings. You can set the parameters listed in the following table:

Parameter	Type	Description
ProfileID	Integer	This is the profile ID.

Parameter	Type	Description
ClassName	Freetext	This is the document class name. It forms the unique key for an entry in the BRWFLD table along with the Profile ID and FieldName.. The value for this field is INVOICES by default. Additional rows are added to the BRWFLD table on top of the entries where the class name is INVOICES in order to specify custom field rules driven by the document classification. When loading the field rules for a document, the system first looks in the BRWFLD table for specific entries which match the document class name during Document_PreExtract. If entries are found, then these settings are used – for all other fields where no class-specific entries are made, the field settings with a class name value of INVOICES are used. The INVOICES entries must not be removed unless those fields are not required at all by any invoice using that profile ID.
FieldName	Freetext	This is the name of the AP Packaged Project field. Do not change the standard field names.
VerifierLabel	Freetext	This text indicates how to label the field on the Dynamic Verifier form.
Active	Boolean	This flag indicates whether the field is activated for the profile.
RequiredInRTS	Boolean	This flag indicates whether the field is mandatory in RTS.
RequiredInVerifier	Boolean	This flag indicates whether the field is mandatory in Verifier.
CountryFilter	Freetext	This is the comma-separated list of country ISO-codes that allow fields to be mandatory for specific countries. This setting is used in conjunction with RequiredInRTS and RequiredInVerifier. If a field is set to mandatory for either of them and the vendor country of origin is not specified in the list, then the field reverts to being optional.
FieldType	AMOUNT DATE TEXT TABLE	This is the type of field.
ForceVerify	Boolean	This is the flag that indicates whether the field must always be marked as invalid and sent to a Verifier for review.
DefaultValue	Freetext	This is the default value field.
DefaultIfNothingExtr	Freetext	This is the default value if the system does not automatically capture a value from the document.
SubRule	Integer	This is the field substitution rule for text fields as registered in the BRWSUBSTITUTION table.
MinLength	Integer	This is the field minimum permitted length for text fields.
MaxLength	Integer	This is the field maximum permitted length for text fields.

Parameter	Type	Description
RightJustify	Boolean	If a value is entered in PadChar, and this parameter is set to TRUE, the system pads the field value with that character from the left until the length specified in MaxLength is reached. If a value is entered in PadChar, and this parameter is set to FALSE, the system pads the field value with that character to the right until the length specified in MaxLength is reached.
PadChar	Single character	This is the padding character for a text field.
RemoveAllSpecials	Boolean	This is the flag that indicates whether to remove all special characters from a text field.
RemoveBlanks	Boolean	This flag indicates whether to remove spaces from a text field.
KeepCertainSpecials	Freetext	This is a non-delimited list of special characters that are retained if RemoveAllSpecials is set to TRUE.
RemoveStartEnd	Boolean	If set to TRUE, AP Packaged Project removes all special characters at the beginning and at the end of an extracted text value.
SubstringStartPos	Integer	The starting character used in conjunction with SubstringLength when trimming an extracted value. Positive numbers start from the left while negative numbers start from the right. For example, the invoice number must only be five characters, but the extracted value is 123456789. SubstringStartPostion of 1 and SubstringLength of 5 equal 12345. SubstringStartPostion of -5 and SubstringLength of 5 equal 56789.
SubstringLength	Integer	This is the substring length.
RemoveLeadingZeros	Boolean	If set to TRUE, AP Packaged Project removes any leading zeros from an extracted text value.
DecimalPlaces	Integer	This is the number of decimal places for an amount field.
NegativeType	Integer	This setting controls the output during export if the extracted value for an amount field is less than zero. Choose one of the following settings: • If set to 1, the minus sign appears after the amount, such as 100.00. • If set to 2, the minus sign appears before the amount, such as 1 -100.00. • If set to 3, the value appears in parentheses, such as 1 (100.00).
OutputForZero	Freetext	If the amount value is zero, then the value of this parameter will be used as the output value during export.
SubstituteValueIfOverZero	Freetext	If the amount value is greater than zero, then the value of the this parameter will be substituted as output value during export.

Parameter	Type	Description
FutureDays	Integer	This value indicates the number of days in the future from the present date for which an extracted date may be considered valid. For example, if today's date is March 20 and a date is extracted as March 31, and the value is set to 10, then the system marks the field invalid as the extracted date is 11 days in the future. If future dates are not permitted, then set the column value to zero. To disable the check, set the column value to -1. In Verifier, the user can pass any value as long as it is a valid date.
NoDaysInPast	Integer	This numerical value indicates the number of days in the past from the present date that an extracted date is considered valid. For example, if today's date is March 20 and a date is extracted as March 9, and the value is set to 10, then the system marks the field invalid as the extracted date is 11 days in the past. If past dates are not permitted, then set the column value to zero. To disable the check entirely, set the column value to -1. In Verifier, the user can pass any value as long as it is a valid date.
DateOnlyInCurrentMonth	Boolean	If this column is set to TRUE, then an extracted date stops in Verifier if the date is not in the current month. In Verifier, the user can pass any value as long as it is a valid date.
FieldMask	Freetext	This is a comma-separated list of valid entries for the extracted or user-entered value. For example, if the content of this column is set to ABCD,WXYZ, only ABCD or WXYZ appear. Wildcard characters are also permitted where a hash symbol (#) is used to represent any number, an at symbol (@) is used to represent any letter, and a question mark (?) is used to represent either a number of a latter. For example, if an entry was restricted to being 10 followed by a letter, then a hyphen, then five digits, the value 10?-##### are entered into the field.
VerifierTextID	Integer	This field is populated with an ID that links to text in the BRWFLDTexts table. If the dynamic Verifier form is used, the system sets the field label to the text specified in BRWFLDTexts table for the Verifier user language preference. This setting takes priority over the text specified against the VerifierLabel column.
ExtractionProfileID	Integer	This column is used for custom fields which are set up to use the configurable extraction feature. The content of this column contains a valid ExtractionProfileID as configured in the BRWExtractionProfile table. A value of zero means that no configurable extraction is required for this field.

10.2.51 BRWFLDTexts

If the Dynamic Verifier form is used, this table holds the text elements that are presented to a user within the Verifier application to represent field and column label display names. The text element selected for each label is driven by the value set in the VerifierTextID column of the BRWFLD table and the language preference of the user, which is set in the LanguageID column of the BRWUser table. The text elements in the Message column can change to suit project requirements. You can add new rows to reference an existing TextID to support translations for

additional languages. For text elements relating to custom fields, create these with a TextID from 900 upwards.

Parameter	Type	Description
TextID	Integer	This is the unique ID for the text element.
LanguageID	Freetext	This is the two-character language ISO-code for the error message text. For example, EN is English, DE is German, and CN is Chinese. AP Packaged Project ships with English and Chinese languages pre-configured.
Message	Freetext	This is the display text for the language specified in the LanguageID column.

10.2.52 BRWINSTR

This table contains a list of instructions and their corresponding texts that are available to be assigned to clients. When using the dynamic verifier form, a button is available to deliver an instructional text to the user to help them with processing documents for a specific client. You can set the parameters listed in the following table:

Parameter	Type	Description
ProfileID	Integer	This is the instructions profile ID.
ProfileName	Freetext	This is the instructions profile name.
Instructions	Freetext	This is the instructions text.

10.2.53 BRWSubstitution

This table contains a list of rules for substituting values in an extracted or user-entered text field. Substitution rules are assigned to text fields in the SubRule column in the BRWFLD table. You can set the parameters listed in the following table:

Parameter	Type	Description
SubstitutionRule	Integer	This is the substitution rule ID.
Original	Freetext	This is the segment of a text string that must be replaced.
Replace	Freetext	This is the text string that must be used to replace the original value entered in Original if present in the string.

The values contained in the Original and Replace columns of the BRWSubstitution table are case-sensitive.

10.2.54 BRWPONPartition

This table contains a list of the purchase order data partitions active within the project. A purchase order partition must be registered within this table before it can be assigned to a client.

You can set the parameters listed in the following table:

Parameter	Type	Description
POPartition	Integer	This is the unique ID of the purchase order data partition.
Description	Freetext	This is the description of the partition.

10.2.55 BRWTAXPartition

This table contains a list of the tax partitions active within the project. A tax partition must be registered within this table before it can be assigned to a client.

You can set the parameters listed in the following table:

Parameter	Type	Description
TaxPartition	Integer	This is the unique ID of the tax data partition.
Description	Freetext	This is the description of the partition.

10.2.56 PICI VersionHistory

This table is a reference table that contains versioning information for the AP Packaged Project database, along with the date of install. You can set the parameters listed in the following table:

Parameter	Type	Description
ID	Integer	This is the ID of version entry.
Version	Freetext	This is the database version, such as 2.1.2100, which is version 2.1, build 2100.
InstallDate	Date	This is the date at which the database was installed.

10.2.57 BRWExtractionProfile

This is a global table which is used for defining field extraction profiles as part of the configurable field extraction for AP Project fields Custom1 to Custom5. Once defined, an extraction profile is subsequently assigned to a custom field in the BRWFLD table using the **ExtractionProfileID** column.

Parameter	Type	Description
ExtractionProfileID	Integer	This is the unique ID given to the extraction profile.
Description	Freetext	This is the description of the extraction profile.
AnalysisProfileID	Integer	This is the analysis profile ID that is used to generate candidates for the field. Entries in this column correspond to an entry in the BRWAnalysisProfileID table.

Parameter	Type	Description
EvaluationProfileID	Integer	This is the evaluation profile ID that is used to evaluate candidates for the field. Entries in this column correspond to an entry in the BRWEvaluationProfileID table.
EvaluationDistance	Freetext	This represents the fuzzy factor the system uses when searching for key words and phrases specified in the evaluation profile. It is a value between zero and one, where zero requires an exact match and one accepts values that do not match at all. The default value for this setting is 0.3.
BaseWeighting	Integer	This is the base weighting that is given to all candidates generated for the field. It is expressed as a percentage, so 50 is at 0.5 weighting to the candidate. This setting can be used if it is expected that one or very few candidates are generated for a field, and that any candidate generated is considered as the extraction result.
OverwriteWith SearchString	Boolean	This flag is set to TRUE if the field result is overwritten with the string compare or levenshtein search string used to generate the candidate.
RemoveNoNumber Candidates	Boolean	This flag, if set to TRUE, removes any candidate that do not contain at least one numeric character.
Distance	Freetext	This represents the fuzzy factor the system uses when generating candidates. It is a value between zero and one, where zero requires an exact match, and one accepts values that do not match at all. The default value for this setting is 0.3.
MaxWordCount	Integer	This value specifies the maximum number of OCR words that are permitted to form a candidate for the field.
MaxWordGap	Freetext	This value specifies the maximum gap in millimeters that is allowed to exist between OCR words for inclusion as part of a generated candidate.
MaxWordLen	Freetext	This value is used to express the maximum length of a candidate in millimeters. Any candidates generated that exceed this length are discarded.
CaseSensitive	Boolean	If this flag is set to TRUE, the system generates candidates based upon the format strings entered in the field analysis profile in a case-sensitive fashion.
KeepSpaces	Boolean	If this flag is set to TRUE, then any space between OCR words are preserved in the generated candidate text.
UseRegions	Boolean	This flag is used to indicate whether candidate generation is restricted by regions on the document.
UseFirstPage	Boolean	This flag is used to indicate that the system generates candidates on the first page of the document. UseRegions must be set to TRUE for this to have effect.
FirstTop	Integer	Expressed as a percentage, this setting is used to define the top most area on the first page for which candidates are generated. A value of zero signifies starting from the top of the page, and a value of 20 signifies starting 20% down the length of the page. A value entered that exceeds 100 is considered as 100. A value less than zero is considered as zero.

Parameter	Type	Description
FirstBottom	Integer	Expressed as a percentage, this setting is used to define the lowest area on the first page for which candidates are generated. A value of 100 signifies going to the end of the page; a value of 80 signifies stopping 80% down the length of the page. A value entered that exceeds 100 is considered as 100. A value less than zero is considered as zero.
FirstLeft	Integer	Expressed as a percentage, this setting is used to define the left most area on the first page for which candidates are generated. A value of zero signifies starting from the left page; a value of 20 signifies starting 20% across the width of the page. A value entered that exceeds 100 is considered as 100. A value less than zero is considered as zero.
FirstRight	Integer	Expressed as a percentage, this setting is used to define the right most area on the first page for which candidates are generated. A value of 100 signifies continuing to the far right of the page. A value of 80 signifies stopping 80% across the width of the page. A value entered that exceeds 100 is considered as 100. A value less than zero is considered as zero.
UseSubseqPage	Boolean	This flag is used to indicate that the system generates candidates on all pages between the first and the last pages of the document. UseRegions must be set to TRUE for this to have effect.
SubsequentTop	Integer	Expressed as a percentage, this setting is used to define the top most area on the subsequent page for which candidates are generated. A value of zero signifies staring from the top of the page. A value of 20 signifies starting 20% down the length of the page. A value entered that exceeds 100 is considered as 100. A value less than zero is considered as zero.
SubsequentBottom	Integer	Expressed as a percentage, this setting is used to define the lowest area on the subsequent page on which candidates are generated. A value of 100 signifies going to the end of the page. A value of 80 signifies stopping 80% down the length of the page. A value entered that exceeds 100 is considered as 100. A value less than zero is considered as zero.
SubsequentLeft	Integer	Expressed as a percentage, this setting is used to define the left most area on the subsequent page for which candidates are generated. A value of zero signifies starting from the left page.A value of 20 signifies starting 20% across the width of the page. A value that exceeds 100 is considered as 100. A value less than zero is considered as zero.
SubsequentRight	Integer	Expressed as a percentage, this setting is used to define the right most area on the subsequent page for which candidates are generated. A value of 100 signifies continuing to the far right of the page. A value of 80 signifies stopping 80% across the width of the page. A value entered that exceeds 100 is considered as 100. A value less than zero is considered as zero.
UseLastPage	Boolean	This flag is used to indicate that the system generates candidates on the last page of the document. UseRegions must be set to TRUE for this to have effect.
LastTop	Integer	Expressed as a percentage, this setting is used to define the top most area on the last page for which candidates are generated. A value of zero signifies starting from the top of the page.A value of 20 signifies starting 20% down the length of the page. A value entered that exceeds 100 is considered as 100. A value less than zero is considered as zero.

Parameter	Type	Description
LastBottom	Integer	Expressed as a percentage, this setting is used for defining the lowest area on the last page for which candidates are generated. A value of 100 signifies going to the end of the page.A value of 80 signifies stopping 80% down the length of the page. A value entered that exceeds 100 is considered as 100. A value less than zero is considered as zero.
LastLeft	Integer	Expressed as a percentage, this setting is used for defining the left most area on the last page for which candidates are generated. A value of zero signifies starting from the left page. A value of 20 signifies starting 20% across the width of the page. A value entered that exceeds 100 is considered as 100. A value less than zero is considered as zero.
LastRight	Integer	Expressed as a percentage, this setting is used for defining the right most area on the last page for which candidates are generated. A value of 100 signifies continuing to the far right of the page. A value of 80 signifies stopping 80% across the width of the page. A value that exceeds 100 is considered as 100. A value less than zero is considered as zero.

10.2.58 BRWAnalysisProfile

This is a global table which is used for defining formats that generate candidates as part of the configurable field extraction for AP Project fields `Custom1` to `Custom5`. Each set of formats is referred to as an analysis profile.

The table has a composite key formed by the combination of the AnalysisProfileID and the IndexID, as candidates can be generated using multiple format strings. Analysis profiles are assigned to extraction profiles via the **AnalysisProfileID** column in the **BRWExtractionProfile** table.

Parameter	Type	Description
AnalysisProfileID	Integer	This is the unique ID given to the analysis profile.
IndexID	Integer	This is the index ID given to each format string used for generating candidates for the field. The index ID must be unique per analysis profile ID.
CompareType	Freetext	This is the compare type used in the generation of candidates based upon the format string. In this column, these options are possible: • SIMPLE – To indicate that the format is a simple expression. • REGULAR – To indicate that the format is a regular expression. • TRIGRAM – To indicate the the trigram method must be used for finding candidates based upon the format string. • LEVEN – To indicate that the Levenshtein method must be used for finding candidates based upon the format string. • STRINGCOMPARE – To indicate that the String Compare method must be used for finding candidates based upon the format string.
Format	Freetext	This is the format string used for generating candidates for the field.

Parameter	Type	Description
IgnoreCharacters	Freetext	This is a list of characters that appear in a candidate in any position, such as a hyphen or a period/full stop, that the system must be tolerant with when generating candidates. This list does not need to be comma separated.

10.2.59 BRWEvaluationProfile

This is a global table which is used for defining key words and phrases for evaluating candidates as part of the configurable field extraction for AP Project fields `Custom1` to `Custom5`. Each set of key words and phrases is referred to as an evaluation profile.

The table has a composite key formed by the combination of the EvaluationProfileID and the IndexID, as candidates may need to be evaluated using multiple key words and phrases. Evaluation profiles are assigned to extraction profiles via the **EvaluationProfileID** column in the BRWExtractionProfile table.

Parameter	Type	Description
EvaluationProfileID	Integer	This is the unique ID given to the evaluation profile.
IndexID	Integer	This is the index ID given to each positive word or phrase that is used to help identify the candidate. The index ID must be unique per evaluation profile ID.
Context	Freetext	This is the word or phrase that helps identify the right candidate for the field result.
Strong	Boolean	This flag indicates whether the word or phrase is a strong indicator for the correct candidate.

11 Script Customization

Project setting customizations includes such things as changing tolerances and thresholds within the project, adding new fields and classes, configuring existing ASE fields, changing or creating Verifier forms, and setting up new users. You can also add script customizations to the project. The following script customization options are available:

- Project setting customizations
- Script customizations

For custom fields and classes, scripts must be added to the appropriate custom field event and class windows. For customizations to the existing Invoices class, this must be done on the UserExits class script window.

Existing fields or classes must not be deleted or renamed under any circumstances. Doing so causes the solution to stop responding.

11.1 AP Packaged Project Modification Restrictions

While it is expected that the AP Packaged Project will be configured and customized to meet particular business requirements, it is essential that the following restrictions are observed and adhered to when performing any kind of modifications within the project:

1. Do not modify any of the existing script code contained in the AP Packaged Project
    - All custom scripts must be written in the **UserExits** module
    - For newly created fields on the **Invoices** class, you are permitted to add custom validations
    - You are permitted to add functions/subroutines at the end of the **GlobalVariables** module
2. Do not modify the generic learnset provided with the AP Packaged Project
    - The only exception to modifying the learnset is to add additional documents for the sole purpose of training new fields
    - If a customer is experiencing a problem with a vendor's invoice, you can use Supervised Learning Workflow (SLW) to create a vendor subclass to maintain vendor-specific training
3. Do not re-train any of the currently trained fields within the AP Packaged Project
4. Visual changes on the Verifier form are permitted

11.2 Sequence of Class Dependencies

When making changes to the script, you must remember that dependencies exist between the various script layers, so it is not possible to execute a script if it dependent on a script which is not being executed.

Executing a script also performs a syntax check. For that reason, the scripts must be executed in the following sequence:

```
GlobalVariables > UserExits > Project
```

11.3 Project Script Organization

The application script is organized logically across several classes.

11.3.1 Project Script Class

The project script class contains script associated with standard AP Project system events, which are known as the ScriptModule events.

These ScriptModule events are called at specific points within the Verifier workflow. For example, pre and post-import, pre and post-OCR, pre and post-classification, and at time of document export.

Do not make any changes to the code on this class level. Doing so may cause the solution to stop responding.

11.3.2 Global Variables Script Class

The global variables script class contains all global variables that are used within the AP Packaged Project.

These data definitions are exposed so that you can elect to use them within custom code if appropriate, and also so that the developer can see the definition of the custom structures and arrays as a point of reference.

In addition to global variables, this script class also includes a series of common functions and subroutines used throughout the solution. You have the option to use these common functions within the custom code placed on the user exit script class, or within the code for any additional classes that are created.

The common functions, along with a description of their potential uses, are described in the following table:

Function/subroutine	Description
ReadPrjINI	This function reads the system configuration INI file.
DicVal	This function returns the value of any parameter contained within the system configuration <project>.ini file. The function parameters are as follows: • strKey is the name of the INI file parameter. • strDic is the name of the INI section in which the parameter is held. Neither strKey nor strDic are case sensitive. For example, if parameter ExportFormat in the BRWDAT table contains MMDDYYYY, then the following command copies MMDDYYYY into local string variable strOutputDateFormat: strOutputDateFormat = DicVal("ExportFormat", "DAT") For the <project>.ini file, the parameters that have a Boolean type of OP, the function returns a value of YES or NO.
DicValCore	This internal function is called by the DicVal function. It must not be called directly from any custom script. The interface is strKey, strInterface.
Parse_INIVal_Yes	This function receives a <project>.ini file parameter value strVal that has been entered against a parameter with Boolean type OP. This function determines whether the value must be interpreted as YES or NO.
SplitString	This subroutine performs a split on a given string based on a nominated separator, and it returns the components of the string to the calling function in an array, along with the number of values in the array. The interface parameters are as follows: • strSource is the input string to be split. • strSplitArray is the array containing the split results passed back to the calling module. • strDesignator is the delimiter to be used when performing the split. • ArrayLineCount is tthe number of array elements in the returned strSplitArray. For example: Dim myString As String Dim Words() As String Dim intWordCount As Integer myString = "MARY HAD A LITTLE LAMB" A space is set as the delimiter. Call SplitString(myString, Words(), " ", intWordCount) The returned words array would contain the following: Words(1) = "MARY" Words(2) = "HAD" Words(3) = "A" Words(4) = "LITTLE" Words(5) = "LAMB" The returned intWordCount parameter would be set to 5.

Function/subroutine	Description
fnConvertToExternal	This function converts a date in the date format used internally within AP Packaged Project, such as DD/MM/YYYY) into a specified format. The interface parameters are as follows: • strDate is the date to be formatted. • strFormat is the format of date (either MMDDYYYY, YYYYMMDD – any other entry returns DDMMYYYY). • strSeparator is the separator to be used when converting the date. For example: Dim myDate as string myDate = "12/08/2009" 12th August 2009 myDate = fnConvertToExternal(myDate, "MMDDYYYY", "-") The value of myDate is now set as "08-12-2009".
fnConvertToInternal	This function is used to convert a date with a specified format into the date format used internally within AP Packaged Project, such as DD/MM/YYYY. The interface is as follows: • strDate is the date to be formatted to DD/MM/YYYY. • strFormat is the current format of strDate (either YYYYMMDD, MMDDYYYY – any other entry returns DDMMYYYY. • strSeparator is the separator currently used in strDate. For example: Dim myDate as String myDate = "2009-08-12" 12th August 2009 myDate = fnConvertToInternal(myDate, "YYYYMMDD", "-") The value of myDate is now set to "12/08/2009".
fnFormatDateForExport	This function converts a date in the Verifier output format, as configured in the DAT section of the system configuration, into a date in the export output format, as configured in the BRWDAT table. For example, an invoice date can potentially appear in any format, but the system converts it to the export format specified in the BRWDAT table. If that format is MMDDYYYY, then 12th August 2009 is displayed in Verifier as 08/12/2009, which is also the technical content of the field object text property, for example the contents of pField.Text or pWorkdoc.Fields("MyDate").Text). fnFormatDateForExport takes the technical contents of the field and converts it into the date format as specified in the BRWDAT table. Therefore, if the export format is YYYYMMDD with a hyphen (-) as the separator, then the following command populates string variable strDate with 2009-08-12: strDate = fnFormatDateForExport(pWorkdoc.Fields("MyDate").Text) The interface of the function is as follows: strDate is the date to be converted.

Function/subroutine	Description
fnWriteXMLField	This function writes a single line into the XML file. It is intended for use within UserExitXMLOutput to provide a mechanism to add a custom field into the XML file with just a single command. The interface of the function is as follows: strAttribute is the name of the <project>.ini file parameter containing the tag for the XML field. strValue is the value of the field to be outputted. For example, in the BRWEXPHEADER table, a new row is created for the fieldname InvoiceCode with a value of INVCODE in the column XMLTag. A new field is created against the invoices class in the project called InvoiceCode that contains the extracted value of 12345. This value must be written to the invoice header section of the XML file. This can be achieved by putting the following in the UserExitXMLOutput framework: Select Case strSection Case cXMLDocHeader … Case cXMLInvHeader fnWriteXMLField("HCInvoiceCode", pWorkdoc.Fields("InvoiceCode").Text) … End Select This writes out the following line into the invoice header section of the XML file: <INVCODE>12345</INVCODE>
fnWriteXMLDateField	This function writes out a date field to the XML file where the date to be written is in the Verifier output date format specified in the BRWDAT table. As well as writing the value into the XML file, the system converts the date passed into the date export format as specified in the column ExportFormat of the BRWDAT table. The interface and function usage is identical to that of fnWriteXMLField as described above.

Function/subroutine	Description
fnWriteDBHeaderField	This function writes an additional database header field into a downstream database if database export is activated in the BRWEXP table. It is intended to provide a developer with a single command to accomplish this within UserExitDBHeaderExport. The interface of the function is as follows: • strAttribute is the name of the .ini file parameter containing the table column mapping for the destination field in the database. • strValue is the field value to be written to the database. • str.INIFileKeyName is the current <project>.ini file parameter being assessed by the database header output routine. This value is passed unaltered from the value passed into the user exit. • strFieldValue is the current value of the field to be written into the database header table. This value is passed unaltered from the value passed into the user exit. For example, in the BRWEXPHEADER table, a new row has been added for the field name InvoiceCode where the column XMLTag has a value of INVCODE, which denotes the technical column name of the downstream invoice header database table. A new field has been created against the invoices class in the project called InvoiceCode which contains the extracted value of 12345. To write the field value into the column INVCODE in the database document header table, insert the following code into UserExitDBHeaderExport: fnWriteDBHeaderField("InvoiceCode", pWorkdoc.Fields("InvoiceCode").Text, str.INIFileKeyName, strFieldValue) When the export runs, the database column INVCODE is populated with the value 12345.
fnWriteDBHeaderDateField	This function writes out a date field to the invoice header database table where the date to be written is in the Verifier output date format specified in the BRWDAT table. Apart from writing the value into the database table, the system also converts the date passed into the date export format as specified in column ExportFormat in the BRWDAT table. The interface and function usage is identical to that of fnWriteDBHeaderField as described above.
fnGetFileName	This function receives a full file name, which includes the file path and file extension, and returns the name of the file itself. For example, if c:\My Documents\12345.tif is passed to the function, the output is 12345. The interface is strFileName.
fnGetBaseClass	This simple function returns the base class associated with the class passed to the function. If a base class is passed to the function, the same base class is returned. For example, if the function receives ExpenseSheets and that class is a child class of Invoices, then the function returns Invoices. The interface is strClass.
fnIsVerifier	This function returns a Boolean true value if the current WebCenter Forms Recognition Module executing the script is the Verifier Module.

Function/subroutine	Description
fnGetBatchID	This function receives the path to a document in the batch directory (strWorkfile), parses the file path, and returns the batch ID number as a string. The interface is strWorkfile.
fnIsAlpha	This function returns a Boolean true value if the string passed with the parameter strString is made up entirely of alpha characters (upper or lower case). The interface is strString.
fnGetUserDecimalSeparator	These functions read the local Windows settings for the user logged onto the machine and returns either a full stop/period, or a comma, depending on the decimal separator preferences.
fnWriteCSVField	This function replaces a user-defined literal in the CSV file configuration with its corresponding value. It is intended for use within UserExitCSVFile to provide a developer with a mechanism to add a custom field into the CSV output file with just a single command. The interface of the function is as follows: • strRecordtext is the current text of line to be written into the CSV file. • strKey is the CSV file group number, such as 01, 02, and so on. • strSymbol is the user-defined literal to be replaced, such as <%ZIC>. • strValue is the value to replace the literal with. For example: If the CSV group in the BRWCSV table with IndexID 1 has <%ZIC> in the column FormatLine1, where <%ZIC> is intended to represent the extracted value of custom field InvoiceCode, then the function in UserExitCSVFile is called as follows: fnWriteCSVField(strRecordText, strKey, "<%ZIC>", pWorkdoc.Fields("InvoiceCode").Text)
fnWriteCSVDateField	This function replaces a user-defined literal in the CSV file configuration with its corresponding date value. ,It is intended for use within UserExitCSVFile to provide a developer with a mechanism to add a custom field into the CSV output file with just a single command. This function must only be used if the date value to be written into the CSV file is in the Verifier Output Format configured in the BRWDAT table. The date is outputted in the date format configured for the CSV file group. If no configuration has been made here, it is formatted according to the date output format configured in the BRWDAT table. The interface and function usage is identical to that of fnWriteCSVField.

Function/subroutine	Description
fnSetDBConnection	This function can be called from a user exit in order to connect to a database. The function takes in a database connection string with the strConnection input parameter. If the connection is already available, the index of the connection in global database connection array objDBConn is returned. If it is not available or not open, the function initializes the connection and return the relevant index of the objDBConn object. If the connection cannot be made, the function returns -1 and an appropriate error message is written into the standard WebCenter Forms Recognition log file. For example, the following code instantiates a database connection and executes an SQL call where the variable myDBConnection represents the connection string and mySQL represents the SQL statement (both string variables): • Dim lngConnection As Long Dim myConnection As ADODB.Connection • lngConnection = fnSetDBConnection(myDBConnection) If lngConnection = -1 Then Connection could not be made – error handling Else Execute SQL using connection Set myConnection = objDBConn(lngConnection) myConnection.Execute(mySQL) End If objDBConn is a global database object available that is used in any user exit. The interface is strDBConnection.
fnMatchDBComponents	This is a supporting function that is used by fnSetDBConnection.
fnCheckDBArray	This is a utility function that checks to see whether a passed database connection array of the type ADODB.Connection is initialized. If it is not, the function initializes it. The interface is myArray().
fnExtractDBComponents	This is a supporting function used by fnSetDBConnection.
fnGetFieldAnalysisSettings	This function returns an instantiated AnalysisSettings object for the given associative search engine field oASSA and the document class strClass. The interface is strClass, oASSA.
fnIsValueInList	This function takes a comma-separated list in the input parameter strList and a value strValuePreserve. The function returns a Boolean true value if strValuePreserve is one of the values in the list. The interface is strList, strValuePreserve.
fnConvertToDouble	This utility function takes in a string strString and converts it to a double value in a way that is consistent with the locale settings of the machine. If the string cannot be converted, the output is zero. The interface is strString.
fnIsNumeric	This utility function returns a Boolean true value if all characters passed in the input parameter strTemp are numeric, such as 0-9. The interface is strTemp.
fnGetMiscChargeTotalForCode	This function adds up all of the invoice miscellaneous charges for a given miscellaneous charge code strCode as registered in the BRWMSCCATEGORY table. It returns this total with the function name. The interface is pWorkdoc, strCode.

Function/subroutine	Description
fnCalculateExchangeRate	This function calculates the correct exchange rate to pass during document export based upon user entry in the exchange rate and local VAT amount fields in the Verifier application. The exchange rate is passed out as a string via the function module name. The example usage is as follows: Dim strExchRateExport As String strExchRateExport = fnCalculateExchangeRate(pWorkdoc) The interface is pWorkdoc.
fnCheckForNull	This function receives a field component of a database record set and returns the value as a string to the calling routine via the function name. If the field component has a null value, an empty string is returned. The interface is strString.
fnConvertBoolean	This function receives a Boolean field component from a database record set. If the value is positive (and NO if the value is negative), it returns YES to the calling routine via the function name. The interface is blBool.
fnSetFromFileName	This function takes the name of a parameter in the IMP section of the <project>.ini file, along with the document file name. It parses out the corresponding value from the file name passing it back to the calling routine using the function name. If the field is a date, it is formatted in accordance with the VerifierFormat setting in the BRWDAT table. The interface is strFieldName, strFile.
RedimClientGlobals	This function takes the global client buffer array and initializes it if it has not already been done.
RedimFSGlobals	This function takes the global field settings buffer array and initializes it if it has not already been done.
fnGetClientData	This function receives a client ID and returns its corresponding settings from the BRWClient table using the ClientData structure. The example usage (to retrieve details for client zero) is as follows: Dim Client As ClientData Client = fnGetClientData("0") Interface: strClientID
fnGetClientDataForWorkdoc	This function receives a workdoc object and returns the client settings configured in the BRWClient table that are associated with that workdoc using the ClientData structure. The example usage is as follows: Dim Client As ClientData Client = fnGetClientDataForWorkdoc(pWorkdoc) The interface is pWorkdoc.

Function/subroutine	Description
fnGetFieldSettings	This is the function to retrieve the field settings from the BRWFLD table for a given combination of field name and profile ID. The settings are passed back using the FieldSettings structure. The example usage (to retrieve settings for the invoice number associated with profile ID 1) is as follows: Dim FS as FieldSettings FS = fnGetFieldSettings("INVOICENUMBER", "1") The interface is strFieldName, strProfileID.
fnFormatAmountForExport	This function receives an amount passed as a double, along with an optional field name as registered in the BRWFLD table. It returns the amount back as a string having applied the export formatting rules configured in the BRWAMT and BRWFLD tables. If no field name is passed, then the formatting rules set against AmountTotal in BRWFLD for the same profile ID is used. The interface is dblAmount, strFieldName.
fnReadSubRule	This function receives a substitution rule ID as a string and populates the SubRule object with the rule details, which is then passed back to the calling routine using the function name. The interface is strRule.
fnGetValueForIR	This function receives the text for an invalid reason as displayed in the field in Verifier via the parameter strIR and returns a corresponding property (specified by strValue) belonging to that rule back to the calling routine using the function name. Possible values for strValue are RULE and EXPORTCODE. The interface is strValue, strIR.
fnFormatTextForExport	This function receives a string value and the name of a field registered in the BRWFLD table and applies the corresponding format for exporting. The formatted field is then passed back to the calling routine using the function name. The interface is strText, strFieldName.
RedimCountryGlobals	This function takes the global country buffer array and initializes it if it has not already been done.
fnReadCountryData	This function retrieves details for a given country. The country details are passed back using the CountryData structure. In order for the function to operate, the country lookup must be activated with the BRWCTR table. Example usage (to retrieve details for the UK): Dim Country As CountryData Country = fnReadCountryData("GB") The interface is strCountry.

Function/subroutine	Description
PasswordEncryption	This subroutine receives a database connection string using the strConnection parameter. If missing a password, it checks whether an encrypted password is included within the <project>. ini file. If it is included, the password is read from the ini file, it is decrypted, and the full string passed back to the calling routine. If not, the connection string is returned as passed. If an error occurs, the strPWError parametrer is populated with an error message. The interface is strConnection, strPWError.
fnMultipleRecords	This function receives a recordset object and returns TRUE if there is more than one record in that record set. The interface is myRecordSet.
fnGetIRNone	This function retrieves the text indicating that no invalid reason is set (default of NONE) based upon project configuration settings and returns it through the function module name. The function has no import parameters.
fnReadText	This function receives a dictionary name and text ID and returns the corresponding text based upon the current language preference, passing it back to the calling module through the function name. Interface: strTextID, strDictionary
PasswordEncryption	This subroutine receives a database connection string using the strConnection parameter.If missing a password, it checks to see whether an encrypted password has been included within the <project>. ini file. If it has, the password is read from the INI file,it is decrypted, and the full string passed back to the calling routine. If it has not, the connection string is returned as passed. If an error occurs, the strPWError parametrer is populated with an error message. The interface is strConnection, strPWError.
fnMultipleRecords	This function receives a recordset object and returns TRUE if there is more than one record in that record set. The interface is myRecordSet.
fnGetIRNone	This function retrieves the text indicating that no invalid reason has been set (default is NONE) based upon the project configuration settings and returns it via the function module name. The function has no import parameters.
fnReadText	This function receives a dictionary name and text ID and returns the corresponding text based upon the current language preference, passing it back to the calling module via the function name. Interface: strTextID, strDictionary
fnWritePITXMLField	This function writes a single line into the ProcessIT XML file. The interface of the function is as follows: • strAttribute is the name of the <project>.ini file parameter containing the tag for the XML field. • strValue is the value of the field to be outputted. • FieldPos is the field positional and confidence metadata. It is used if the parameter PITIncludeFieldPositions is set to TRUE in the BRWEXP table.

Function/subroutine	Description
fnWritePITXMLDateField	This function writes out a date field to the ProcessIT XML file where the date to be written is in the Verifier output date format specified in the BRWDAT table. It also writes the value into the XML file. The system converts the date passed into the date export format as specified in column ExportFormat of the BRWDAT table. The interface and function usage is identical to that of fnWritePITXMLField as described above.
EmptyFieldPos	This subroutine resets a given FieldPos object. Interface: FieldPos
fnPopulateFieldPosForField	This function takes in a workdoc object and the technical name of an AP Project field via the input parameter strFieldName and returns a populated FieldPos structure to the calling module via the function name. The field name is case-sensitive. Interface: pWorkdoc, strFieldName

11.3.3 User Exits Script Class

This class contains the project user exit script points. You must not remove or change the definitions of the provided user exits.

11.3.4 Invoices Script Class

The Invoices script class contains the source code for the invoice class validation events, which includes the logic that is used to validate fields and the document as a whole, as well as to control the behavior of the Dynamic Verifier form.

You can add new validation events that correspond to newly created fields on the Invoices script class. These extra events must be created at the end of the existing code in the area marked in the script.

No changes must ever be made to any of the existing code. Doing so may make the solution inoperable.

11.3.5 BW Packaged/Generic Script Classes

These scripts must not be deleted, changed, or renamed becausethe `<project>.sdp` file may not be recoverable.

11.4 User Exits

A user exit is a dedicated public subroutine or function on the `UserExits` class script level where custom code can be inserted.

Each user exit is called from a relevant point in the application layer baseline code and provides you with a window to perform a custom activity as is appropriate for your implementation.

Customizations must be implemented in a modular fashion within the user exits. If any ancillary functions are required to support these modules, then these must be created as public functions.

These ancillary functions can be placed on the `UserExits` script class if they are only to be used locally. If they need to be accessed by custom script on other classes, they can be placed at the end of the existing script on the GlobalVariables class in the marked area.

11.4.1 User Exits

The user exits that are available, along with their calling points and suggested uses can be found in the following table:

User exit	Call routine	Description
UserExitCustomExport	ScriptModule_ ExportDocument	This is the user exit for custom export modules, such as for custom flat files or custom database updates. This is the only user exit which has a corresponding activation parameter within the BRWEXP table. The interface is pWorkdoc, ExportPath, strDocLink, LineData, GLData, TaxData, blLinesRequired, Address, Flags. The global variable strExportError is populated with an appropriate error message in the event the export fails. This sets the batch to a status of 750, with the error message set against the invoice number. This exit is only called for documents that have not been voided. Special handling for voided documents is inserted in the user exit UserExitVoidDocumentExport.
UserExitPostExtract	Document_ PostEvaluate on the invoices class	This is the user exit used to set any custom field defaults or to reevaluate any extracted fields. The interface is pWorkdoc.
UserExitRouteDocument	ScriptModule_ RouteDocument	This is the user exit for performing any custom activity connected to the WebCenter Forms Recognition workflow state of each document, such as changing the state based on a property of the workdoc or document file name so that they can be filtered on a user-by-user basis. The interface is pWorkdoc, State.
UserExitPONumberPostEvaluate	PONumber_ PostEvaluate on the invoices class	This is the user exit where a custom routine can be added to reevaluate the weightings for candidates for the purchase order number field, such as to check for their existence in an external database. The interface is pField, pWorkdoc.
UserExitVoidDocumentExport	ScriptModule_ ExportDocument	This is the user exit provided for the custom export of documents belonging to the void class. The interface is pWorkdoc, ExportPath, strDocLink. The global variable strExportError is populated with an appropriate error message if the export fails. This has the effect of setting the batch to a status of 750, with the error message set against the invoice number.

User exit	Call routine	Description
UserExitPONumberValidate	PONumber_Validate on the invoices class	This user exit can be used for custom purchase order number validations. It is called subsequent to the purchase order number being validated against the database table. The interface is pField, pWorkdoc, pValid, lngIndex. If the PO number valid flag is to be changed, then the values of pValid and pWorkdoc.Fields("PONumber").Valid are changed to the new Boolean value. The PO header and line details are passed in global arrays g_POHeader and g_POLines. The lngIndex parameter contains the index number of the purchase order record in the g_POHeader array. The definition of the arrays can be found on the GlobalVariables script level. If the purchase order details are being read from a database, and the line pairing is switched off, the line item array is not populated.
UserExitPONumberValidateStart	PONumber_Validate on the invoices class	This is the user exit used for custom purchase order number validations to be undertaken at the start of the PO number validate routine. The interface is pField, pWorkdoc, pValid, blExit. If the PO number valid flag is to be changed, then the values of pValid and pWorkdoc.Fields("PONumber").Valid are changed to the new Boolean value. If the parameter blExit is set to TRUE, the PONumber_Validate routine exits immediately after returning from the user exit.
UserExitTerminate	ScriptModule_ Terminate	This user exit is called from the beginning of ScriptModule_Terminate. It can be used to unload any global script objects employed in custom script. The interface is ModuleName.
UserExitPreImport	ScriptModule_ PreImport	This user exit is called from the beginning of ScriptModule_PreImport. The interface is pWorkdoc, FilePath, FileType, pCancel.
UserExitPostClassify	ScriptModule_ PostClassify	This user exit is called from the beginning of ScriptModule_PostClassify. The interface is pWorkdoc.
UserExitDocumentTypeValidate	DocumentType_Validate on the invoices class	This user exit is called at the beginning of DocumentType_Validate on the invoices class. The exit can be used to set the valid flag of the document type depending on whether it is an invoice or credit note. The interface is pField, pWorkdoc, pValid.

User exit	Call routine	Description
UserExitAmountMisc PostEvaluate	Internal application	This user exit can be used to evaluate the weighting of candidates for a miscellaneous charge in the AmountMisc field in a manner that is appropriate for the project implementation where the desired contents of the field can change depending on client requirements. The interface is pField, pWorkdoc.
UserExitChangeReportingCountry	Internal application	This user exit permits a change in the reporting country for countries that do not use tax jurisdictions. This exit is implemented as a function. The following default line must not be changed without something appropriate being in its place. UserExitChangeReportingCountry = strCountry The parameter strCountry is initially set to the country of the company code to which the invoice is to be posted, which is assumed to be the tax reporting country. As some companies have sites abroad that are VAT registered in that country, the company code country is not appropriate as a key to retrieve the appropriate tax codes from the AP Packaged Project Tax Table. This user exit provides an opportunity to implement specific business logic to select the correct country, which, in the example above, is the country where the plant is located. For PO invoices, the plant can be retrieved from the POLines array in the Werks property. The interface is strCountry, pWorkdoc.
UserExitSetTolerance	Internal application	This user exit allows greater flexibility for setting the tolerance against which amount fields are cross-validated mathematically within WebCenter Forms Recognition. Standard configuration permits different tolerance values to be assigned to different currencies, but if a further dimension is required, such as considering the company code, then this logic is implemented within this user exit. Adjusting the tolerances requires manipulating the Tolerance array. The interface is Tolerance, pWorkdoc.
UserExitDocumentOnAction	Document_OnAction on the invoices class	This user exit provides an opportunity for a developer to add a script that relates to custom buttons that they may elect to add to the Verifier form. The ActionName parameter, which is passed into the function, is populated with the technical name of the action associated with a user pressing the button as designated in Verifier Design Mode in the WebCenter Forms Recognition Designer. The interface is pWorkdoc, ActionName.

User exit	Call routine	Description
UserExitDBHeaderExport	Internal application	This user exit permits a developer to add custom header fields into the standard database export function or to change the value of an existing field. A row must be added to the BRWEXPHeader table in order to designate the mapping between the export field and the column name of the invoice header database table into which the field is written. The infterface is strINIFileKeyName, pWorkdoc, strDocLink, strFieldValue. strINIFileKeyName denotes the name of the field in column FieldName as per the added row in the BRWEXPHeader table. For example, if column FieldName is set to MyField, then strINIFileKeyName is set to MyField. strFieldValue is the value to be exported to the database. strDocLink is the link to the image of the document and could be a file path of a URL, depending on settings in the REP section of the system configuration.

User exit	Call routine	Description
UserExitXMLOutput	Internal application	This user exit is available for you to add any custom fields into the XML output file. The XML output file is divided into the following sections: • The document attributes, such as class name, scan date. • The invoice header, such as invoice number, date. • The line item detail, such as quantity, unit price. • General Ledger Coding, such as GL account, cost center. Custom fields can be entered into any of these four sections by use of the public fnWriteXMLField and fnWriteXMLDateField functions. This user exit is called by the core application code once for the document header, once for the invoice header, once for each standard line item and once for each general ledger coding line item. The LineData array parameter contains the standard line items, and the GLData array parameter contains the general ledger coding lines. The formal parameter strSection denotes the section of the XML file that the exit is being called for, and lngLine denotes the index of the line item that the user exit is being called for. The infterface is pWorkdoc, LineData(), GLData(), lngLine, strSection.
UserExitCSVFile	Internal application	This user exit is called for each header line outputted to the CSV file. It can be used to map custom user literals to their desired value counterparts, for example, to include a custom field in the CSV file output. Functions fnWriteCSVField and fnWriteCSVDateField are provided for this purpose to condense the operation into a single command. The parameter strRecordText contains the text of the current line to be written into the file and strKey is the CSV index group number for the file being processed. The infterface is pWorkdoc, strRecordText, strKey.

User exit	Call routine	Description
UserExitCSVFileLine	Internal application	This user exit is called for each line item outputted to the CSV file.It can be used to map custom user literals to their desired value counterparts, for example, to include a custom line item component in the CSV file output. Functions fnWriteCSVField and fnWriteCSVDateField are provided for this purpose to condense the operation into a single command. The parameter strRecordText contains the text of the current line to be written into the file. The strKey is the CSV group index number for the file being processed. The LineData structure is also passed in containing details of the current line being outputted. The infterface is pWorkdoc, LineData, strRecordText, strKey.
UserExitExportSuccess	ScriptModule_ ExportDocument	This user exit is called at the point where it is known that all selected exports have been successful for the document being processed. It can be used to update additional reporting data if required. The infterface is pWorkdoc.
UserExitExportFailure	ScriptModule_ ExportDocument	This user exit is called at the point where it is known that export has failed for the document being processed. It can be used to update additional reporting data if required. The reason for the export failure can be found in the global parameter strExportError. The infterface is pWorkdoc.
UserExitValueCheck	Internal application	This user exit is called from the invoice number and invoice date post evaluate events for documents that are classified to the Invoices class. It is called once per candidate, and it is provided as a window for a developer to insert script to disqualify illegal candidates for the invoice number or invoice date. The infterface is pField, pWorkdoc, oCandidate.
UserExitLinePairingPOs	Internal application	This user exit provides functionality to edit the list of purchase orders that are to be considered during the line pairing operation. New purchase orders may also be added based on the criteria that may be coded within the user exit. The interface of the user exit is pWorkdoc. The AP Package Project workdoc object PO() array contains the details of the purchase orders to be used during line pairing. If an error occurs during the user exit code, then global variable strExportError is populated with the reason for the error. Populating this variable has the effect of failing the document export.

User exit	Call routine	Description
UserExitVerifierException	ScriptModule_Verifier Exception	This user exit is triggered when a user send a document to an exception state in Thick Verifier. The interface is pWorkdoc, Reason, CreateNewBatch, BatchName, BatchDocumentState, BatchPriority, BatchFolderName, ApplyExceptionHandling.
UserExitFilterVendorSearch	Document_Post Extract, internal application	This user exit allows a developer to filter the list of vendors shown in the vendor search facility, and also to change the information that is displayed. It can also be used to remove vendors that the system must not consider to be the correct vendor or to adjust the confidence weightings of candidates that are under consideration to be the correct vendor. The interface is as follows: • pWorkdoc is the AP Packaged Project workdoc object. • oCandidate is the vendor candidate object. • Address is the vendor address structure. • strDisplay is the current vendor search box display line for vendor. This is blank if the user exit is called prior to the initial determination of the correct vendor on server side. Checking whether this string is empty or not allows a different behavior to be defined between server side and search box functions. • blReject is t is initially set to FALSE, but you can set it to TRUE if the vendor is excluded from the search results or from the extraction results. Example usage 1: You want to exclude any vendors from being extracted automatically and from appearing for selection in the vendor search if they are not set up for the invoice company code. This can be achieved through the following line of code: If Not fnIsValueInList(Address.CompanyCodes, pWorkdoc.Fields("CompanyCode").Text) Then blReject = TRUE This assumes that a comma-separated list of valid company codes for which the vendor is set up is available in the vendor master extract, and the correct column is mapped against the CompanyCodes parameter in the SRC section of the system configuration.

User exit	Call routine	Description
		Example usage 2: You want to exclude any vendors from being extracted automatically if they are not set up for the invoice company code, but you also want to allow the Verifier user to select those vendors using a vendor search. This can be achieved with the following line of code: If strDisplay = "" And Not fnIsValueInList(Address.CompanyCodes, pWorkdoc.Fields("CompanyCode").Text) Then blReject = TRUE The same assumptions for example 1 apply. Example usage 3: You want to lower the confidence of any vendor candidates that do not belong to the invoice company code by 30 percent. This can be achieved by the following line of code: If strDisplay = "" And Not fnIsValueInList(Address.CompanyCodes, pWorkdoc.Fields("CompanyCode").Text) Then oCandidate.Weight = oCandidate.Weight – 0.3 Example usage 4: The following line of code adjusts the display in the Vendor Search box so that only the vendor name, vendor street address, and zip code are displayed: If strDisplay <> "" Then strDisplay = Address.Name & ", " & Address.Address & ", " & Address.Zip The interface is pWorkdoc, oCandidate, Address, strDisplay, blReject.
UserExitSetReportingLoginName	Internal application	This user exit allows a developer to change the name of the user as reported in the reporting database. This is used in AP Packaged Project and higher where the Web Verifier is being used. Otherwise, the system always populates the Thick Verifier user column in the reporting database with the WebCenter Forms Recognition service user. Input parameter strUserName contains the user name that the system is currently using. The interface is pWorkdoc, strUserName. It is no longer necessary to insert code into this user exit for Thick Verifier implementations. The system always uses the Thick Verifier logon ID as the user name.

User exit	Call routine	Description
UserExitPreMaterialLinePairing	Internal application	This user exit is called prior to the commencement of material line pairing. It is not called for the pairing of service line items. The LineData input parameter is always empty at this point. The dblPOTotal parameter contains the total value of all purchase orders being considered during line pairing. The dblGRTotal parameter contains the total outstanding value of all purchase order numbers being considered during line pairing, such as the value of total goods received against all purchase orders minus the value of total invoice received against all purchase orders. It can be used to skip the MIRA process of line pairing by setting dblPOTotal and dblGRTotal both to zero, although this is not recommended. The interface is pWorkdoc, LineData(), dblPOTotal, dblGRTotal.
UserExitPostLinePairing	Internal application	This user exit is called after the line pairing. The automatic tax determination functions are completed during document export, but before any data outputs is actually carried out. It gives the developer an opportunity to look at the line pairing results (held in the LineData array) and make any changes or additional customizations as required. Interface parameter blLinesRequired is set to TRUE if line items are to be exported for the document in question. The interface is pWorkdoc, LineData(), TaxData(), blLinesRequired.

User exit	Call routine	Description
UserExitAddressArray	Internal application	This user exit is called each time the details for a vendor are read from the vendor pool. It gives the developer an opportunity to amend or add new parameters to the Address array. The user exit is not called if the vendor details are already read and loaded into the local cache. For example, the client has been unable to supply the vendor country of origin as a standard two-character ISO-code, as required by the application. The client is only doing business with vendors from the US and Canada and is able to provide USA and CAN as the country codes. As a result, you can add script to this use exit to convert a 3-character code to a 2-character code. The following script in the user exit converts the client's country code to the required application country code: Select Case Address.Country Case "USA" Address.Country = "US" Case "CAN" Address.Country = "CA" End Select The interface is oASSA, strVendorID, Address.
UserExitDocumentValidate	Document_Validate on the Invoices class script level	This user exit is called from Document_Validate on the Invoices class script level. It can be used to code in additional document level validations and activities. The interface is pWorkdoc, pValid.
UserExitEditDocumentWeblink	Internal application	This user exit permits a developer to manipulate the document web link, as stored in the reporting database and exported downstream. The current web link is passed into the user exit using the strWebLink interface parameter.This may be changed to meet your business needs. The current unique document ID is passed in the strDocID interface parameter, which may not be changed. The interface is strWebLink, strDocID.
UserExitInvoiceNumberValidate	InvoiceNumber_ Validate on the Invoices class	This user exit is available to include additional validations and formatting against the invoice number field. For example, your business rule states that if an invoice number is extracted that is longer than 16 characters, the system must format it so the last 16 characters are retained. The following line of code accomplishes this: If Len(pField.Text) > 16 Then pField.Worktext.Text = Right(pField.Text, 16) The interface is pField, pWorkdoc, pValid.

User exit	Call routine	Description
UserExitPOValidateStart2	PONumber_Validate on the Invoices class	This user exit is called just after UserExitPOValidateStart routine, and it is intended for the same functional purpose, except that it has an extended interface to make it easier and work with UserExitPOValidateStart. Included in the interface is the Address structure, which contains the full details of the current vendor. Also included is the POKey structure, the contents of which may be changed if required. This was not previously possible with UserExitPOValidateStart. The interface is pField, pWorkdoc, pValid, blExit, POKey, Address.
UserExitSetDocFlags	Internal application	This user exit allows a developer to set the properties on the Flags object based on a custom set of requirements. The interface is pWorkdoc, Flags.
UserExitInvoiceDate Validate	InvoiceDate_Validate on the Invoices class	This user exit is available for custom validation logic to be added to the invoice date field validation event. The interface is pField, pWorkdoc, pValid.
UserExitDueDate Validate	DueDate_Validate on the Invoices class	This user exit is available for custom validation logic to be added to the invoice due date field validation event. The interface is pField, pWorkdoc, pValid.
UserExitDeliveryDate Validate	DeliveryDate_Validate on the Invoices class	This user exit is available for custom validation logic to be added to the delivery date field validation event. The interface is pField, pWorkdoc, pValid.
UserExitVerifierFormLoad	ScriptModule_Verifier FormLoad	This user exit is called at the end of ScriptModule_VerifierFormLoad. The interface is pWorkdoc, FormClassName, FormName.
UserExitSetVendorCountry	Internal application	This user exit allows a country to be set for the purposes of field formatting and validation. For example, if the country filter is being used, it determines a vendor from which the country would normally be derived. To set the country, the import parameter strCountry is set to the two-character ISO-code for the country in question. The interface is strCountry, pWorkdoc.
UserExitScriptModule Initialize	ScriptModule_ Initialize	This user exit is called at the end of ScriptModule_Initialize. The interface is pWorkdoc.

User exit	Call routine	Description
UserExitPostImport	ScriptModule_ PostImport	This user exit is called from the beginning of ScriptModule_PostImport. The interface is pWorkdoc.
UserExitPostImportBatch	ScriptModule_Post ImportBatch	This user exit is called at the beginning of ScriptModule_PostImportBatch. The interface is pWorkdoc.
UserExitTextField Formatting	Internal application	This user exit can be used to apply custom formatting and validations to fields of type TEXT in a way that can be used to augment that standard validations and formatting configurable using the BRWFLD table. The interface is pField, pWorkdoc, pValid, FS, Client.
UserExitPreClassify	ScriptModule_Pre Classify	This user exit is called from the beginning of ScriptModule_PreClassify. Interface: pWorkdoc
UserExitDynamicVerifier Fields	Internal application	This user exit can be used to change the order of the fields as they appear on the Dynamic Verifier form. The interface is vFields().
UserExitChangeDB ConnectionString	fnSetDBConnection on the GlobalVariables script level	This user exit is located at the bottom of the GlobalVariables script. It can be used to manipulate the database connection string read from the <project>.ini file, for example to support password encryption. The connection string is passed into the user via via strConnection, and this may be changed to an alternative connection string as per the requirements of the project. The interface is strConnection.
UserExitAlternatePayee Validate	AlternatePayee_ Validate on the Invoices class	This user exit is available for custom validation logic to be added to the alternate payee field validation event. The interface is pField, pWorkdoc, pValid.
UserExitUpdateSystem Security	ScriptModule_Update SystemSecurity	This user exit is called during the system security update event that is set to run as a periodic background job on the runtime server. In AP Packaged Project, the system security event is used to load users created in the BRWUser table into the main system user table. This user exit is triggered subsequent to that process. The interface is InstanceName.

User exit	Call routine	Description
UserExitMoveDocument	ScriptModule_Move Document	This is the user exit that is called when a document is sent to an exception batch in Thick Verifier. The internal application uses this event to apprise the reporting tables of any change in the document batch ID. The interface is pWorkdoc, OldBatchID, NewBatchID, Reason.
UserExitBatchOpen	ScriptModule_Batch Open	This is the user exit that is called upon the opening of a batch in Thick Verifier. The interface is UserName, BatchDatabaseID, ExternalGroupID, ExternalBatchID, TransactionID, WorkflowType, BatchState.
UserExitProcessBatch	ScriptModule_ ProcessBatch	This is the user exit that is called during the Custom Processing workflow step. The interface is pBatch, InputState, DesiredOutputStateSucceeded, DesiredOutputStateFailed.
UserExitBatchClose	ScriptModule_Batch Close	This is the user exit that is called when a batch is exited in Thick Verifier. The interface is Username, BatchDatabaseID, ExternalGroupID, ExternalBatchID, TransactionID, WorkflowType, BatchState.
UserExitAppendWorkdoc	ScriptModule_Append Workdoc	This is the user exit that is called when a user merges documents together in Thick Verifier. The interface is pLastWorkdoc, pCurrentWorkdoc, pAppendType.
UserExitPreClassify Analysis	ScriptModule_Pre ClassifyAnalysis	This is the user that is called during the classification event where the classification matrix may be adjusted or extended to influence the classification result. The interface is pWorkdoc.
UserExitReadPODetails	Internal application	This user exit is called during the validation of the purchase order number on both server and Verifier side, as well as during document export prior to line pairing. The user exit allows a developer to implement a custom purchase order number lookup (for example, using a web service) instead of using the standard options. More details regarding this user exit can be found in Appendix M. The interface is pWorkdoc, POHeader, POLineItems(), POKey, Client, Address, strPOReadError, blReadPOLines, blDuplicatePO, blExport, and blPONotFound.

User exit	Call routine	Description
UserExitPIFExport	Internal application	The user exit is called prior to the system sending an XML output through a web service call. The user exit provides a developer with the opportunity to change the XML document that are sent to PIF through parameter xmlDoc. Within the user exit, if it is decided that export must fail and the document must be sent to Verifier with a 750 state, this is achieved by setting the global variable strExportError to an appropriate error message. The interface is pWorkdoc, LineData, TaxData, strDocLink, xmlDoc, Client, Address, blLinesRequired, and Flags.
UserExitPOVendorValidate	PONumber_Validate on the 'Invoices' class	This user exit is called on the server side only if a PO vendor is validated automatically by the system. It provides a developer with the opportunity to reject the validated PO vendor based upon custom critiera. To reject the PO vendor, the parameter blVendorOK must be set to FALSE. The interface is pWorkdoc, POKey, Address, Client, andblVendorOK.
UserExitCompanyCodeValidate	CompanyCode_Validate on the 'Invoices' class	This user exit is available for custom validation logic to be added to the company code field validation event. The interface is pField, pWorkdoc, and pValid.
		lngMaterialLineCount is the number of material lines in the LineData structure (for example, if set to 4, then the first 4 entries in the LineData array are material lines). **blCondAllotted** is the boolean flag indicating whether the miscellaneous charge could be allocated to a condition. This is initially set to FALSE, but must be set to TRUE in the user exit code if the business logic added is able to allocate the charges to the correct conditions. **strDescription** is the description of the miscellaneous charge type as read from the configuration.

User exit	Call routine	Description
UserExitFocusChanged	Document_Focus Changed on the 'Invoices' class	This user exit is called during the standard FocusChanged event in Verifier. This event is called each time the field or table cell focus in Verifier is changed, or when the system moves to a new document. Any script inserted into this user exit must be kept to a minimum as overly time-consuming operations could make the Verifier application cumbersome to use. For example, the script that is relevant only when the system moves on to a new document would be better placed in UserExitVerifierFormLoad. The interface is pWorkdoc, Reason, OldFieldIndex, NewFieldIndex.
UserExitCheckBank Account	VendorID_Validate & PONumber_Validate on the 'Invoices' class	The user exit is called during the bank account check, where the system attempts to determine the appropriate bank account based upon the content of the document and the bank records present in the vendor master data. It allows a developer to customize alternative logic for the bank account selection. The import parameter Address contains the details for the currency vendor, which the bank account records specified against the BankDetails property. The currency invoice currency is passed into the user exit via the strCurrency' parameter. The interface is pWorkdoc, Address, strCurrency.
UserExitCellText Formatting	Internal application	This user exit is called for each table cell in the LineItems table which is activated and set to the type TEXT. It can be used as a means for a developer to create custom validation and formatting routines for the table cell. The parameters passed into the user exit are as follows: • pWorkdoc - Standard Workdoc object that provides access to all document field information (including the originally extracted line item data), the document class name, the document OCR text and the document file name. myWorktext – the table cell worktext object FS – field settings object containing the defined configuration for the table cell pTable – the LineItems table object lngRow – the row index for the table cell strColumnName – the technical name of the column in LineItems Client – client settings object pValid – cell valid/invalid flag strError – cell validation error message

User exit	Call routine	Description
UserExitEmployeeIDValidate	EmployeeID_Validate on the 'Invoices' class	This user exit is available for custom validation logic to be added to the employee ID field validation event. The interface is pField, pWorkdoc, and pValid.
UserExitAccountNumber Validate	AccountNumber_ Validate on the 'Invoices' class	This user exit is available for custom validation logic to be added to the account number field validation event. The interface is pField, pWorkdoc, and pValid.
UserExitPreLinePairing	Internal application	This user exit is called during document export immediately before the commencement of the line pairing process, after purchaser order information has been read. It provides an opportunity for a developer to apply custom logic to the purchase order data returned. Interface: pWorkdoc, Client, Address
UserExitSetPITDocType	Internal application	This user exit is called during the creation of the ProcessIT XML file. It allows a developer to use custom logic when setting the XML file <DOCTYPE> attribute by changing the value of the strDocPITType parameter. Interface: pWorkdoc, strDocPITType.
UserExitPITXMLOutput	Internal application	This user exit is designed to allow a developer to edit the body of the XML file created for ProcessIT. It is called once the XML document has been assembled by the system based on settings within the AP Project configuration tables, but before it is written out to the export directory or sent to the ProcessIT interface table. The XML document is passed into the user exit via the xmlDoc parameter. Interface: pWorkdoc, LineData(), GLData(), TaxData(), xmlDoc
UserExitSetEntrySheetNumber	Internal application	This user exit allows a developer to pass a service entry sheet number via the parameter strEntrySheetNo for use during line pairing. Interface: pWorkdoc, strEntrySheetNo
UserExitMexicanUUIDValidate	MexicanUUID_Validate on the 'Invoices' class	This user exit is available for custom validation logic to be added to the Mexican UUID number field validation event. The interface is pField, pWorkdoc, and pValid.
UserExitFilterEmployeeSearch	Internal application	This user exit offers the same functionality as UserExitFilterVendorSearch (described above), except as it relates to the employee field. The interface is pWorkdoc, oCandidate, Address, strDisplay, blReject.

User exit	Call routine	Description
UserExitRejectPOFor LinePairing	Internal application	This user exit offers the capability to reject a database purchase order for inclusion in line pairing based on custom criteria such as, information contained within the purchase order header. If the purchase order is rejected, then the import parameter blReject must be set to TRUE. oPOHeader is the purchase order HEADERDATA object retrieved from using the purchase order lookup BAPI. Hence, the order-from vendor can be retrieved via the following: oPOHeader.Value("VENDOR_ID") For the purchase order header data read from a database, the record set object DBPOHeaderRecordSet must be used, so a column that contains the vendor ID with a technical name of VENDOR_ID can be accessed via: DBPOHeaderRecordSet("VENDOR_ID")
UserExitOCRXMLOutput	Internal application	This user exit is designed to allow a developer to edit the body of the OCR XML file. It is called once the XML document is complete for output, but before it is written out to the export directory. The XML document is passed into the user exit via the xmlDoc parameter. The interface is pWorkdoc, xmlDoc.

11.4.2 Retrieve Custom Error Messages

You can include custom error messages as entries in the BRWERR table, rather than hard coding them within the script. To retrieve an error message, complete the following step.

The error message number range assigned for customer usage is 900-999, which must be adhered to in order to prevent any conflicts in the event of an upgrade.

1. Open the **BRWERR** table.
2. Add the following code:

```
Dim myError As String
myError = DicVal("900", "ERR")
```

3. Save the changes.

11.4.3 Project Data Structures

AP Packaged Project uses internal data structures to pass data between functions and subroutines. You can use some of these data structures in the user exit scripts.In some instances, these structures are defined as a parameter in the interface.

11.4.3.1 LineData Structure

The LineData structure contains the line item detail that is being exported.

It is based on the document line item extraction results, but may deviate from that depending on the results of line pairing. For example, a single line on the invoice corresponds to more than one line item on the purchase order. It is also based upon how invoice miscellaneous charges are handled.

The LineData structure is present in the interfaces to the following user exits:

- UserExitCustomExport
- UserExitXMLOutput
- UserExitCSVFileLine
- UserExitPreMaterialLinePairing
- UserExitPostLinePairing
- UserExitPIFExport
- UserExitPITXMLOutput

The elements contained within the structure are listed in the following table:

Structure Element	Type	Description
INVOICE_DOC_ITEM	Integer	This is the invoice line item number from 1-n.
PO_NUMBER	String	This is the purchase order number. It is populated only if line pairing has been successful for this item.
PO_ITEM	String	This is the purchase order line item number. It is populated only if line pairing has been successful for this item.
DE_CRE_IND	X or blank	This is the subsequent debit/credit indicator that denotes whether the line item is a subsequent debit or credit line item. If this value is set to X and the document type is INVOICE, then the line item is treated as a subsequent debit, the amount only and not quantity. If the value is set to X and the document type is CREDIT, then the line item is treated as a subsequent credit. If the value is blank, the line item is treated as a regular line item.

Structure Element	Type	Description
QUANTITY	Double	This is the invoice line item quantity.
ITEM_AMOUNT	Double	This is the invoice line item total/
PO_UNIT	String	This is the order unit of measure. It is populated with the purchase order line item order unit of measure if the line item has been paired.
PO_PR_UOM	String	This is the order price unit of measure., It is populated with the order price unit of measure from the purchase order line item if the line item has been paired. In all other cases, it is blank.
PO_PR_QNT	Double	This is the invoice line item quantity expressed in the order price unit of measure.
TAX_CODE	String	This is the invoice line item tax code.It is populated using the tax determination procedure if a line item is paired.
TAXJURCODE	String	This is the invoice line tax jurisdiction code. It represents the downstream ERP system ID for the tax office to which tax is payable for this line item, as used in countries that have tax jurisdictions for their sales tax. It is populated only if the line item is paired.
UNIT_PRICE	Double	This is the invoice line item unit price.It is only populated for unpaired line items. In all other cases, it has a value of zero.
DESCRIPTION	String	Invoice line item description. If the line item is paired, the description is set to the description on the purchase order. If the line is unpaired, this field contains the raw text description that was read from the invoice. For third-party freight invoices, service invoices, and MIRA invoices where no line items were required in the TAB section, and line pairing was either not successful for any lines, or was not carried out, the description is set to THIRD PARTY FREIGHT, SERVICE and MIRA respectively.
MATERIAL_NO	String	Invoice line item material number. If the line item is paired, this is populated with the material number from the purchase order line item. If the line is not paired, this is populated with any values read from the invoice.
TAX_RATE	String	Invoice line item tax rate. This is the percentage rate of tax applied to the invoice line item. If no percentage tax rate at line item level can be ascertained, then this value is blank.
LINETYPE	String	Invoice line item type. This is lifted from the purchase order line item to which an invoice line is paired.
CHARGECODE	String	Invoice line item charge code. This is lifted from the purchase order line item to which an invoice line is paired.

Structure Element	Type	Description
CHARGECODEID	String	Invoice line item charge code ID. This is lifted from the purchase order line item to which an invoice line is paired.
MATERIALGROUP	String	Material group. This is lifted from the purchase order line item to which an invoice line is paired.
DISTILLER_LINE	String	Original WebCenter Forms Recognition line item number. This is the original line item number in the WebCenter Forms Recognition table field (as viewed in Verifier) that the invoice line was drawn from. This is always populated for unpaired line items and is a 1-based, not a zero-based index.
PLANT	String	Plant ID. This is lifted from the purchase order line item to which an invoice line is paired.
COMPANYCODE	String	Company code to which the line item corresponds. This is lifted from the purchase order when an invoice line is paired.
POTYPE	String	This is populated only if line pairing has been successful for this item.
BUSINESSUNIT	String	This is populated only if line pairing has been successful for this item.
MISCCHARGE	'X' or blank	X indicates that this is a miscellaneous charge line item. The flag is blank if the system has identified the line as a non misc-charge item.
ITEM_TEXT	String	This is the item text for the line item.

11.4.3.2 POHeader Structure

The POHeader structure is used within `UserExitReadPODetails`. Within that user exit, it is populated with the details of a purchase order header retrieved by a custom lookup to an external data source.

The components of the structure are listed in the following table:

Structure Element	Type	Description
DOCTYPE	String	This is the purchase order document type.
COMPANYCODE	String	This is the purchase order company code.
VENDROID	String	This is the purchase order from vendor.
SITEID	String	This is the site ID for the purchase order vendor.
CURR	String	This is the purchase order currency.
RELEASEFLAG	String	This is the purchase order release flag.
DIFFINV	String	This is the remit-to vendor ID.
STATUS	String	This is the purchase order document status.

11.4.3.3 POLineItems Structure

The POLineItems structure is used within `UserExitReadPODetails`. Within that user exit, it must be populated with the details of a purchase orders lines retrieved by a custom lookup to an external datasource.

The components of the structure are provided in the following table:

Structure Element	Type	Description
LINENO	String	This is the purchase order line item number.
MATERIALNO	String	This is the purchase order line item material number.
MATERIALGROUP	String	This is the purchase order line item material group.
DESCRIPTION	String	This is the purchase order line item description.
POQUANTITY	Double	This is the line item order quantity.
UOM	String	This is the purchase order line item quantity unit of measure.
UNITPRICE	Double	This is the line item unit price.
PUOM	String	This is the purchase order line item price unit of measure.
PRICEUNIT	String	This is the purchase order line item price unit.
TOTAL	Double	This is the line item order total.
TAXCODE	String	This is the purchase order line item tax code.
TAXJUSOURCE	String	This is the purchase order line item tax jurisdiction code.

Structure Element	Type	Description
TOTALQUANTITYDELIVERED	Double	This is the total quantity already delivered for the purchase order line item.
TOTALVALUEDELIVERED	Double	This is the total value of the goods already delivered for the purchase order line item.
TOTALQUQNTITYINVOICED	Double	This is the total quantity that has been invoiced for the purchase order line item.
TOTALVVALUEINVOICED	Double	This is the total value of the goods invoiced for the purchase order line item.
ITEMCATEGORY	String	This is the purchase order line item category.
PLANT	String	This is the ID of the purchase order line item plant (for example, where the goods are to be delivered).
CHARGECODE	String	This is the purchase order line item charge code.
CHARGECODEID	String	This is the purchase order line item charge code ID.
ERS	Boolean	This flag must be set to TRUE if the purchase order line item is marked for Evaluated Receipt Settlement (ERS).
MULTIPLEACCOUNTASSIGNMENT	String	This is an indicator as to whether the purchase order line item is set up with a multiple account assignments. Leave this field blank if multiple account assignment do not exist.
ACCOUNTASSIGNMENTCATEGORY	String	This is the purchase order line item account assignment category.

11.4.3.4 POKey Structure

The POKey structure contains the elements that comprise the unique key used to identify a single purchase order. The structure is used in `UserExitLinePairingPOs` and is defined using the parameters in the following table.

The POKey structure is present in the interfaces to the following user exits:

- UserExitLinePairingPOs
- UserExitPOValidateStart2
- UserExitReadPODetails
- UserExitPOVendorValidate
- UserExitRejectPOForLinePairing

The structure is defined using the parameters in the following table:

Structure Element	Type	Description
PONUMBER	String	This is the purchase order number.
COMPANYCODE	String	This is the company code. This value is populated only if the company code forms part of the key to identify a unique purchase order.
POEXTENSION	String	This is the purchase order number extension. This field contains the additional key required to identify a purchase order uniquely.

11.4.3.5 TaxData Structure

The TaxData structure holds the tax amounts that are relevant for each tax code. It is populated if automatic tax determination has been activated successfully, all lines have been paired, and tax is not required to be calculated by the downstream ERP system.

The TaxData structure is used in the interfaces of the following user exits:

- UserExitCustomExport
- UserExitXMLOutput
- UserExitPostLinePairing
- UserExitPIFExport
- UserExitPITXMLOutput

The structure consists of the components listed in the following table:

Structure Element	Type	Description
TAX_CODE	String	This is the ERP system tax code.
TAX_AMOUNT	Double	This is the tax amount.
TAX_RATE	Double	This is the tax rate as a percentage. For example, 20 would be 20%.
TAX_BASE	Double	This is the invoice amount to which the tax applies.
EXTRACT_LINE	String	This is the original line number (1-based index) of the VATTable where the value is captured. This value is only set if the TaxData structure is populated based on the content of the VAT table.

11.4.3.6 Address Structure

The Address structure contains data elements associated with a particular vendor, such as the vendor ID, the vendor name, and address details, along with additional information.

The extent to which the data is populated depends on the availability of the data in the vendor extract and mapping in the BRWSRC table.

The structure is used in the interfaces of the following user exits:

- UserExitCustomExport
- UserExitFilterVendorSearch
- UserExitAddressArray
- UserExitPOValidateStart2
- UserExitReadPODetails
- UserExitPIFExport
- UserExitPOVendorValidate
- UserExitCheckBankAccount
- UserExitPreLinePairing
- UserExitFilterEmployeeSearch

The structure consists of the parameters listed in the following table:

Structure Element	Type	Description
NAME	String	This is the name of the vendor.
ADDRESS	String	This is the vendor street address line 1.
ADDRESS2	String	This is the vendor street address line 2.
ZIP	String	This is the vendor zip code or postal code.
ID	String	This is the unique vendor ID. This is the unique ID from the point of view of the data extract, where each row must have a unique reference. This is not the unique vendor ID from the point of view of the ERP system if a site ID is also used.
SITEID	String	This is the vendor site ID.
TELNO	String	This is the vendor telephone number.
CITY	String	This is the vendor city.
STATE	String	This is the vendor state. For U.S. addresses, the state code is expected here, such as CA = California, or VA = Virginia.
COUNTRY	String	This is the vendor country. This must be the two-character ISO-code for the country, such as US = United States of America, DE = Germany, GB = United Kingdom.
POBOX	String	This is the vendor Post Office (PO) box number.
POBOXZIP	String	This is the zip code or postal code associated with the vendor Post Office (PO) box.
EUMEMBER	Boolean	This is the boolean indicator that denotes whether the vendor belongs to an EU member state country.
VATREGNO	String	This is the vendor VAT registration number. If the vendor is registered for VAT in more than one country, then multiple VAT registration numbers are provided in the form of a comma-separated list.
TAXID1	String	This is the vendor tax ID 1.
TAXID2	String	This is the vendor tax ID 2.
TAXJURCODE	String	This is the ID of the tax office where the vendor is based.
CURR	String	This is the vendor currency.
INVOICETYPE	String	This is the vendor invoice type. This is set to a value that denotes either a Purchase Order-supplying vendor or a vendor who submits invoices that legitimately do not reference a purchase order. If this column is used to determine the invoice type field, the meaning of the values contained in the column must be mapped against the POValue and NPOValue parameters in the BRWITY table.
PAYMENTMETHODS	String	This is the comma-separated list of payment method codes appropriate for the vendor.

Structure Element	Type	Description
BANKDETAILS	String	This is the vendor bank account details. This must be a colon-separated list in the following format: BankAccount,SortCode,ERPBankAccountCode A sortcode is the U.S. equivalent of a routing number.
IBAN	String	This is the vendor international bank account number.
UTILITYFLAG	String	This is the indicator as to whether the vendor is a utility vendor.
PORSUBNO	String	This is the vendor POR subscriber number used only for Switzerland.
EXTERNALID	String	This is the ERP system vendor ID if a site ID is used.
ACCOUNTGROUP	String	This is the ERP system vendor account group.
COMPANYCODES	String	This is the comma-separated list of company codes that are valid for the vendor.
SIRETID	String	This is the vendor SIRET ID. This is an ID code used in France that uniquely identifies a single vendor at a single address. It is often found on French invoices.
VENDORIDENTIFIER	String	This is the unique vendor identifier code, such as a Chinese tax number.
PARTITIONID	String	This is the vendor partition ID.
CUSTOM1 through CUSTOM5	String	Use these fields to store additional information about the vendor that is specific to your needs.

11.4.3.7 Tolerance Structure

The tolerance structure holds a series of limits the system uses when performing mathematical validations on the amounts read from the invoice.

The tolerance limits are set by the tolerance group assigned to the invoice currency. The tolerance structure is used in the interface for the `UserExitSetTolerance` user exit.

The structure consists of the components listed in the following table:

Structure Element	Type	Description
HEADERTOLERANCE	Double	This is the maximum amount by which the document header amounts, such as total = tax + freight + sum of line items/subtotal are allowed to deviate from one another before the system marks them as being invalid.
TABLEROWTOLERANCE	Double	This is the maximum amount by which the line item level calculation, such as quantity * unit price = total is allowed to deviate before the system marks a line item as being invalid.
TAXTOLERANCE	Double	If automatic tax determination is activated, then this is the maximum amount by which a system-calculated tax amount or tax rate is allowed to deviate from a tax amount read from the invoice or a tax rate contained within the tax table.

Structure Element	Type	Description
NODECIMALPLACES	Boolean	This value is set to TRUE if the invoice currency does not have a subunit, such as pennies, cents. Common world currencies that do not have subunits are the Hungarian Forint and the Japanese Yen.

11.4.3.8 Flags Structure

The flags structure contains a range of Boolean indicators for document validation that you can use to determine which field items are relevant for export.

The Flags structure is present in the interfaces of the following user exits:

- UserExitCustomExport
- UserExitPIFExport

The structure consists of the components in the following table:

Structure Element	Type	Description
MIRA	Boolean	This flag is set to TRUE if the invoice is a MIRA and line item extraction is not required for MIRA invoices. The settings in the BRWTAB table determine if line items are required under what circumstances.
INVALID	Boolean	This flag is set to TRUE if the user has selected the INVOICE AMOUNTS DO NOT ADD UP invalid reason in Verifier.
PONOTRELEASED	Boolean	This flag is set to TRUE if the PO has not been released and line item extraction is not required for invoices under that circumstance.
NOVENDOR	Boolean	This flag is set to TRUE if the user has selected either the VENDOR NOT FOUND or MISSING/INVALID VENDOR & PO invalid reasons in Verifier.
NOPO	Boolean	This flag is set to TRUE if the user has selected either the MISSING/INVALID PO or the MISSING/INVALID VENDOR & PO invalid reason in Verifier.
CREDIT	Boolean	This flag is set to TRUE if the document type is CREDIT and line items are not required for credit memos.
SERVICE	Boolean	This flag is set to TRUE if the PO type is SERVICE and line items are not required for invoices that relate to service purchase orders.
FI	Boolean	This flag is set to TRUE if the invoice type is NO-PO and line items are not required for NO-PO invoices.
THIRDPARTYFREIGHT	Boolean	This flag is set to TRUE if the vendor is third party freight.

Structure Element	Type	Description
NOLINEITEMS	Boolean	This flag is set to TRUE if any of the following options are true: • Line item extraction is switched off for the project. • The line items table field is not activated in the BRWFLD table for the profile ID. • An invalid reason of MISSING/INVALID VENDOR & PO is set. • An invalid reason of MISSING/INVALID PO is set and line items are not required under such circumstances. • An invalid reason of VENDOR NOT FOUND is set and line items are not required under such circumstances. • The vendor has been identified as a utility vendor and line items are not required for utility vendors.

11.4.3.9 AccountingData Structure

The AccountingData structure is used to hold general ledger coding strings relevant to the invoice.

It is populated at time of data export if miscellaneous charges are present on the document, and the output is required as a general ledger account entry. It is used in the interfaces of the following user exits:

- UserExitCustomExport
- UserExitXMLOutput

The GLData global project array, which is based on this structure, may be populated for either XML or database output in UserExitPostLinePairing.

The structure consists of the following components:

Structure Element	Type	Description
INVOICE_DOC_ITEM	Integer	This is the general ledger coding string line item number from 1-n.
PO_NUMBER	String	This is the purchase order number.
PO_ITEM	String	This is the purchase order line item number.
GL_ACCOUNT	String	This is the general ledger account number.
COMP_CODE	String	This is the coding string company code.
DB_CR_IND	String	This is the debit/credit indicator.
COSTCENTER	String	This is the cost center.
SERIAL_NO	String	This is the serial number.
PROFIT_CTR	String	This is the profit center.
WBS_ELEM	String	This is the work breakdown structure element.

Structure Element	Type	Description
PROFIT_SEGM_NO	String	This is the profit segment number.
CO_AREA	String	This is the controlling area.
CMMT_ITEM	String	This is the commitment item.
FUNDS_CTR	String	This is the funds center.
BUS_AREA	String	This is the business area.
COST_OBJECT	String	This is the cost object.
FUNC_AREA	String	This is the functional area.
FUND	String	This is the fund.
REF_DATE	String	This is the reference date.
ORDERID	String	This is the internal order number.
SUB_NUMBER	String	This is the sub number.
NETWORK	String	This is the project network.
ACTIVITY	String	This is the project activity.
RL_EST_KEY	String	This is the real estate key.
ASSET_NO	String	This is the asset number.
SD_DOC	String	This is the sales order document number.
SDOC_ITEM	String	This is the sales order document item number.
TAX_CODE	String	This is the tax code.
TAXJURCODE	String	This is the tax jurisdiction code.
ITEM_AMOUNT	Double	This is the coding string amount.
QUANTITY	Double	This is the quantity.
PO_UNIT	Double	This is the order unit of measure relating to the Quantity element.
PO_PR_UOM	Double	This is the order price unit of measure.
PERCENT	Double	This is the distribution percentage.
SHEET_NO	String	This is the service entry sheet number.
CUSTOM1 to CUSTOM5	String	These are custom account assignment fields that may be used by a developer.

11.4.3.10 ClientData Structure

The ClientData data structure holds element details for the current client.

It is passed explicitly as an interface parameter in the `UserExitTextFieldFormatting` user exit, but can be read from any routine where pWorkdoc is available using the the `fnGetClientDataForWorkdoc` global function.

The components of the structure are listed in the following table:

Structure Element	Type	Description
CLIENTID	String	This is the client ID.
PROFILEID	String	This is the profile ID assigned to client.
EXPORTPROFILEID	String	This is the export profile ID assigned to client.
FORCEVERIFY	Boolean	If set to TRUE, all documents assigned to this client are sent to Verifier for manual review.
CLIENTGROUP	String	This is the group to which the client is assigned.
CLIENTNAME	String	This is the client name.
INSTRUCTIONS PROFILEID	String	This is the instructions profile ID assigned to client.
REQUIRESREVIEW	Boolean	This is the Requires review flag.
VENDORPARTITION	String	This is the vendor partition ID assigned to client.
EMPLOYEEPARTITION	String	This is the employee partition ID assigned to client.
POPARTITION	String	This is the PO partition ID assigned to the client.
TAXPARTITION	String	This is the tax partition ID assigned to the client.
PRIORITY	String	This is the batch priority level for client.

11.4.3.11 CountryData Structure

The country data structure holds the basic settings for a country.

It is used for defining the output structure of the fnReadCountryData function. The components of the structure are listed in the following table:

Structure Element	Type	Description
COUNTRY	String	This is the country ISO-code ID.
EUMEMBER	Boolean	This is the flag to indicate whether the country is a member of the European Union.
CURR	String	This is the local currency of the country.
NAME	Boolean	This is the name of the country.

11.4.3.12 FieldSettings Structure

The FieldSettings structure holds details associated with a given field as read from the BRWFLD table.

The structure is used as a parameter in the `UserExitTextFieldFormatting` user exit. The components of the structure are listed in the following table:

Structure Element	Type	Description
FIELDNAME	String	This is the field name.
PROFILEID	String	This is the profile ID.
VERIFIERLABEL	String	This is the field verifier label.
ACTIVE	Boolean	This is the field active flag.
REQUIREDINRTS	Boolean	This denotes whether the field is required in RTS.
REQUIREDINVERIFIER	Boolean	This denotes whether field entry is mandatory in Verifier.
COUNTRYFILTER	String	This is the comma-separated list of countries that control whether the field is mandatory or not.
FIELDTYPE	String	This is the field type.
FORCEVERIFY	Boolean	This is the force verify indicator.
DEFAULTVALUE	String	Field's default value.
DEFAULTIFNOTHINGEXTR	String	This is the field's default value if no value is extracted automatically.
SUBRULE	String	This is the field's substitution rule.
MINLENGTH	Integer	This is the field minimum length.
MAXLENGTH	Integer	This is the field maximum length.
RIGHTJUSTIFY	Boolean	This is the indicator as to whether the field must be right-justified if a pad character is used.
PADCHAR	String	This is the padding character.
REMOVEALLSPECIALS	Boolean	This is the flag to indicate whether special characters must be removed.
REMOVEBLANKS	Boolean	This is the flag to indicate whether blank spaces must be removed.
KEEPCERTAINSPECIALS	String	This is the list of special characters that must be retained.
REMOVESTARTEND	Boolean	This is the flag to indicate whether special characters must be removed from the start and end of the string.
SUBSTRINGSTARTPOS	Integer	This is the substring start position.
SUBSTRINGLENGTH	Integer	This is the substring length.

Structure Element	Type	Description
REMOVELEADINGZEROS	Boolean	This is the flag to indicate whether leading zeros must be removed from a string.
DECIMALPLACES	Integer	This is the number of decimal places for an exported amount.
NEGATIVETYPE	Integer	This is the negative type code.
OUTPUTFORZERO	String	This is the export value if an amount field is zero.
SUBSTITUTEVALUEIFOVER0	String	This is the export value if an amount is greater than zero.
FUTUREDAYS	Long	This is the number of days that an extracted date is permitted to be in the future.
NODAYSINPAST	Long	This is the number of days that an extracted date is permitted to be in the past.
DATEONLYINCURRENT MONTH	Boolean	This is the flag to indicate whether the date must only be in the current month.
FIELDMASK	String	This is the list of valid field masks for text fields.
EXTRACTIONPROFILEID	String	This is the extraction profile ID (for custom fields 1-5).
ANALYSISPROFILEID	String	This is the field analysis profile ID.
EVALUATIONPROFILEID	String	This is the field evaluation profile ID.
BASEWEIGHTING	Double	This is the base weighting to be accorded to all candidates generated for the field.
REMOVENONUMBERCANDIDATES	Boolean	This flag indicates whether candidates are removed if they do not contain at least one numeric character.
OVERWRITEWITHSEARCHSTRING	Boolean	This flag indicates whether the extracted value is overwritten by the string compare or levenshtein search used for generating the candidate.

11.4.3.13 FieldPos Data Structure

The FieldPos data structure holds positional and confidence information for a field.

It is used in `UserExitPITXMLOutput` by custom script so that custom field output to the PIT XML file may include the field's corresponding positional and confidence information.

The components of the structure are in the following table:

Structure Element	Type	Description
TOP	Long	This is the top co-ordinate position of the field measured in pixels.
LEFT	Long	This is the left co-ordinate position of the field measured in pixels.

Structure Element	Type	Description
HEIGHT	Long	This is the height of the field measured in pixels.
WIDTH	Long	This is the width of the field measured in pixels.
PAGENR	Long	This is the page number of the document on which the field appears.
CONFIDENCE1	Double	This is the confidence value of the best candidate for the field.
CONFIDENCE2	Double	This is the confidence value of the second best candidate for the field.
VALID	Boolean	Flag to indicate whether the field is valid or invalid. If set to TRUE, the field is valid.

11.4.4 Configure Triggering of User Exits in Verifier

User exits are triggered when a user is working a problem document in Verifier.

11.4.4.1 General Actions

The following are a list of general user exit calls that apply to all fields in Verifier:

Verifier Action	User Exits
User presses Enter on any field.	UserExitSetVendorCountry
User presses Enter on any field set to the type of TEXT in the BRWFLD table.	UserExitTextFieldFormatting

11.4.4.2 Specific Actions

The following table lists the user exits that are fired when a user performs a certain task in the order in which they are fired:

Verifier Action	User Exits
User presses Enter on the document type field.	UserExitDocumentTypeValidate
User presses Enter on the invoice type field.	The following user exits are always called: • UserExitPOValidateStart • UserExitPOValidateStart2 • The following user exits may be called: • UserExitReadPODetails (if a custom PO look-up is used) • UserExitCheckBankAccount (only if the PO is validated successfully against a database) • UserExitPOValidate (only if the PO is validated successfully against a database) • UserExitAddressArray (if a vendor has not been loaded into the buffer)

Verifier Action	User Exits
User presses Enter on the invalid reason field.	The following user exits are always called: • UserExitPOValidateStart • UserExitPOValidateStart2 The following user exits may be called: • UserExitReadPODetails (if a custom PO lookup is being used) • UserExitCheckBankAccount (only if the PO is validated successfully against a database) UserExitPOValidate (only if the PO is validated successfully against a database) • UserExitAddressArray (if a vendor has not been loaded into the buffer)
User presses Enter on the invoice number field.	UserExitInvoiceNumberValidate
User presses Enter on the invoice date field.	UserExitInvoiceDateValidate
User presses Enter on the invoice due date field.	UserExitDueDateValidate
User presses Enter on the delivery date field.	UserExitDeliveryDateValidate
User presses Enter on the vendor ID field.	The following user exits are always called: • UserExitPOValidateStart • UserExitPOValidateStart2 The following user exits may be called: • UserExitReadPODetails (if a custom PO lookup is being used) • UserExitCheckBankAccount (if the invoice type is NO-PO, or only if the PO is validated successfully against a database if the invoice type is PO) • UserExitPOValidate (only if the PO is validated successfully against a database) • UserExitAddressArray (if a vendor has not been loaded into the buffer)
User opens the vendor search box or performs a vendor search.	The following user exits are always called: • UserExitFilterVendorSearch The following user exits may be called: • UserExitAddressArray (if a vendor has not been loaded into the buffer)
User clicks a button on the Verifier form.	UserExitDocumentOnAction
User verifies the last invalid field on the Verifier form.	The following user exits are always called: • UserExitDocumentValidate The following user exits may be called: • UserExitSetReportingLoginName (if reporting is activated)
User presses Enter on the company code field.	UserExitCompanyCodeValidate
Cursor/field focus is changed or the system moves to a new document.	UserExitFocusChanged

Verifier Action	User Exits
User presses Enter on the purchase order number or purchase order number extension fields.	The following user exits are always called: • UserExitPOValidateStart • UserExitPOValidateStart2 The following user exits may be called: • UserExitReadPODetails (if a custom PO lookup is being used) • UserExitCheckBankAccount (only if the PO is validated successfully against a database) • UserExitPOValidate (only if the PO is validated successfully against a database) • UserExitAddressArray (if a vendor has not been loaded into the buffer)
User presses enter on the employee ID field.	UserExitEmployeeIDValidate
User presses enter on the account number field.	UserExitAccountNumberValidate
User presses enter on the Mexican UUID number field.	UserExitMexicanUUIDValidate

11.4.5 Configure Reporting and Custom Base Classes

If the project involves adding a new base class, then the standard reporting trail as written into the reporting tables is incomplete if the following steps are not performed:

1. On the custom base class, add a script to the `Document_PreExtract` and `Document_Validate` events.
2. On the base class, create a custom `tmpCLSRES` field.

11.4.5.1 Add a Custom Script to the Document PreExtract and Document Validate Events

To insert a custom script to the `Document_PreExtract` and `Document_Validate` events, complete the following steps:

1. Open **WebCenter Forms Recognition Designer**.
2. In the **Class view in definition** mode, right-click the custom base class, and select **Show script**.
3. Copy the following script into the script window:

```
Private Sub Document_PreExtract(pWorkdoc As SCBCdrPROJLib.ISCBCdrWorkdoc)
  If InStr(UCase(ScriptModule.ModuleName), cVerifier) Then
    gblVerifierAsServer = TRUE
  Else
    gblVerifierAsServer = FALSE
  End If
  fnGetClassResultsMatrix(pWorkdoc)
  fnReporting(pWorkdoc, "DOCUMENTPREEXTRACT")
End Sub
Private Sub Document_Validate(pWorkdoc As SCBCdrPROJLib.ISCBCdrWorkdoc, pValid
As Boolean)
  gblVerifierAsServer = FALSE
```

```
  If UCase(ScriptModule.ModuleName) <> cVerifier Then
    fnReporting(pWorkdoc, "DOCUMENTVALIDATESERVER")
  Else
    fnReporting(pWorkdoc, "DOCUMENTVALIDATEVERIFIER")
  End If
End Sub
```

4. Save the script and close **WebCenter Forms Recognition Designer**.

11.4.5.2 Add the tmpCLSRES Field

The **tmpCLSRES** field stores the full classification results and weightings before they are written to the reporting database. It is an internal field and does not require any action beyond its creation. If the field is not created, the full classification results and weightings are not written into the reporting database, only the final class in which the document was placed is reported.

To create the tmpCLSRES field, complete the following steps:

1. In the custom base class, go to the **Fields** view mode.
2. Right-click in the grey space and select **Insert Field Definition**.
3. Enter the new field name as **tmpCLSRES** and click **OK**.

> **Note:** The field name is case-sensitive and must be entered exactly as shown above.

4. Right-click the new field and select **Show Properties**.
5. In the pane on the right side, select the **Validation** tab, and then select the **Always Valid** flag.
6. Save the project.

11.5 Add Additional Custom Fields

AP Packaged Project contains five custom fields that are available for use as required.

If the five fields are not sufficient for your business needs, it is possible to add additional custom fields.

11.5.1 Create an Additional Field in the WebCenter Forms Recognition Designer Module

To create a new field in the WebCenter Forms Recognition Designer module, complete the following steps:

1. Using the WebCenter Forms Recognition Designer module, open the `<project>.sdp` file.
2. Click the **Spanner** icon.
3. To reveal the field view list for the class, in the class list on the right side, expand the **Invoices** class level.
4. Scroll to the bottom of the field list, then right-click, and then select **Insert field definition**.
5. Enter a name for the new field and click **OK**.
6. Save the project.

11.5.2 Add a Validation Script to the Invoices Class

To add a validation script to the Invoices class, complete the following steps:

1. Using the WebCenter Forms Recognition Designer module, open the `<project>.sdp` file.
2. Click the **Spanner** icon.
3. To reveal the field view list for the class, in the class list on the right side of the screen, expand the **Invoices** class level.
4. Scroll to the bottom of the field list, then right-click, and then select **Show Script**.
5. Scroll to the bottom of the script and paste the following code, substituting the technical field name:

```
 Private Sub Custom6_Validate(pField As SCBCdrPROJLib.ISCBCdrField, pWorkdoc As
SCBCdrPROJLib.ISCBCdrWorkdoc, pValid As Boolean)

 Call FieldSettings(pField, pWorkdoc, pValid)

 End Sub
```

6. Save the changes.

11.5.3 Add a Database Entry in the BRWFLD Table

After you add a validation script to the Invoices class, you must create a new entry in the BRWFLD database table for the profile ID. To add a database entry in the BRWFLD database table, complete the following steps:

1. Open the **BRWFLD** database table.
2. In the **ProfileID** column, enter a profile ID.
3. In the **FieldName** column, enter the name of the field you created in the **Create an additional field** in the **Designer** module section.
4. In the **VerifierLabel** column, enter the name of the Verifier label.
5. Set the **Active** column to `TRUE`.
6. To make a field mandatory, set the **RequiredInRTS** and **RequiredInVerifier** columns to `TRUE`. Or, to make a field optional, set the **RequiredInRTS** and **RequiredInVerifier** columns to `FALSE`.
7. In the **FieldType** column, set the field type, such as `TEXT` or `DATE`.
8. To force an invoice to be verified every time, set the **ForceVerify** column to `TRUE`.
9. Save the changes.

11.5.4 Add a Field to Dynamic Verifier Form

The Dynamic Verifier form comes prepopulated with several fields. You can add additional fields to the Dynamic Verifier form to meet your business requirements.

The following sections contain the instructions for adding a field to the Dynamic Verifier form.

11.5.4.1 Edit the Verifier Form in Designer

1. Ensure that **Form_Invoices_1** is set as the default form.

2. Open the project in **Designer Mode**.
3. Navigate to **Verifier Test Mode**.
4. On the left side of the screen, in the class view, click the **Invoices** node, then right-click **Form_Invoices_2 and s**elect **Delete**.

5. Click the icon on the toolbar, and select **Insert a Default Form**.
6. On the right side of the window, add three buttons to the new Verifier form. It does not matter where these buttons are placed as long as they are in the blank space.
7. Label the first button `Instructions`, the second `Vendor search`, and the third `Employee search`.
8. On the toolbar on the right side of the screen, click the **actions** icon.
9. Click **Insert**.
10. In the **Define Actions** dialog, in the **Descriptions** column, enter a description for each of the buttons.
11. In the **Button** column, enter the following for each button:
    a. For **Instructions**, enter `Button 1 [Instructions].`
    b. For Vendorsearch, enter `Button 2 [Vendor search].`
    c. For Employee search, enter `Button 3 [Employee search].`
12. In the **Accelerator** column, in the **Instructions** row, enter `F2`.
13. Click **Close** and save the file.

11.5.4.2 Add the Custom Field to UserExitDynamicVerifierFields

1. Open the project in the **Designer**.
2. Navigate to **Designer Mode**.
3. Right-click on **UserExits** and select **Show script**.
4. Navigate to **UserExitDynamicVerifierFields** and insert the following script:

```
Public Sub UserExitDynamicVerifierFields(vFields() As String)
' User exit for changing order of verifier fields
  ReDim Preserve vFields(UBound(vFields)+1)
  vFields(UBound(vFields)) = "Custom6"
End Sub
```

5. In the script, replace `Custom6` with the technical name of the field.
6. Click **Close** and save the file.

12 Improve Data Extraction

AP Packaged Project comes preconfigured with a generic learnset for the extraction of data from invoices.

The system already delivers high-extraction results from invoice documents of any format without any special configuration. However, for those documents which stop in the Verifier application due to missing or incorrectly read fields, you can take additional measures to build upon the system's existing extraction capability, thus ensuring that these problematic documents pass through the system untouched.

12.1 Improve Extraction Process Flow

You can improve the extraction process flow by configuring the system to pass documents that might stop in the Verifier application based on the rules you create.

12.1.1 Check for OCR Problems

Problems with the OCR read of a document are the chief cause as to why AP Packaged Project may not be able to extract the invoice information.

When considering the viability of improving extraction for a given invoice, the first thing to check is the document OCR results.

The following are two types of OCR problems that affect the generic extraction results:

- OCR problems that affect the field value itself that requires extraction.
- OCR problems that affect the context surrounding the field value.

If during the document OCR, the information that required extracting is compromised, then the document is not a good candidate for further improvement.

12.1.1.1 OCR Problems on the Field Value

The following may cause OCR problems with the field value:

- Handwriting or stamps obscuring the required data
- Poor invoice print quality (e.g. a dot matrix print ribbon running out of ink)
- Shading on the invoice (e.g. black lettering on a dark grey background)
- Misalignment of print text against a template background
- Handwritten information

Checking the invoice OCR read can be done in both the Designer and Verifier applications by hovering the cursor over the area of the document where the correct field value is located.

In Designer, when viewing a document, clicking **Highlight All Words** on the top toolbar shows the OCR results for the entire document.

Each green block represents a single OCR word. Positioning the cursor over each green block shows how the system has interpreted the word.

When the actual field value is compromised by poor OCR, you can try out the following:

- Adjust scanner settings using technologies such as Virtual Rescan or PerfectPage, which can prove beneficial for problems caused by shading and text misalignment
- Ask the vendor for a better quality of invoice

12.1.1.2 OCR Challenges in the Field Contextual Information

Generic field extraction can also be compromised by poor OCR on the contextual information surrounding the field or no contextual information at all. The AP Project Extraction engine uses this context to assign confidence values to field candidates. So, if the context is not available, or it cannot be read correctly, the correct candidate may not be accorded to requisite level of confidence to be extracted.

In the example below, the context surrounding the invoice number cannot be read properly, hence the confidence value of the field is below the threshold.

The OCR on the context does not need to be perfect as the system uses fuzzy pattern-matching technologies to interpret the information surrounding the field.

The absence of a meaningful context does not mean that extraction failure is guaranteed, as the system also considers the value in relation to other fields, along with its general position on the document. However, missing contextual information leads to the correct candidate being accorded a lower confidence value in comparison to documents where a context was provided.

Extraction issues which fall into this category can be resolved by creating a class using the supervised learning workflow, which is described in the next section.

12.1.2 Build a Class using Supervised Learning Workflow

The WebCenter Forms Recognition Supervised Learning Workflow (SLW) is a standard core product feature that allows you to train the system to improve extraction for problematic invoices in a production environment.

It is appropriate for use in instances where field data is not extracted due to low system confidence or instances where the system delivers an incorrect extraction result. It cannot resolve problems caused by poor OCR, except in instances where the OCR issue concerns the surrounding field contextual information.

Before using the Supervised Learning Workflow, the learnset manager user must consider whether this is a worthwhile step for the invoice in question. For example, if very few invoices are received from the vendor in question, then it may not be worth creating a class, instead letting the documents stop in the Verifier application. However, if the invoice is from a high-volume vendor, then creating a class makes much more sense.

It is recommended that you keep a limit of 500 SLW classes per project file.

In a production environment, learnset additions are proposed by AP users through the standard Verifier application. The additions are subsequently reviewed by a learnset manager, who subsequently decides which of those additions are suitable to be promoted to the global project.

In the background, this learnset addition creates a separate class which contains the learnset improvement. The format of the name of the new class is derived from the settings against the VendorASSA field on the Invoices class level.

Typically, this is set to the name of the vendor, then an underscore, then the vendor ID, although the class is actually used for any invoice of a similar layout which exhibits a similar extraction problem.

12.1.3 Configure a Supervised Learning Workflow in Designer

WebCenter Forms Recognition Supervised Learning Workflow (SLW) is a standard core product feature that allows users to train the system to improve extraction for problematic invoices in a production environment.

You can use the SLW feature within the Designer application, which is done in Verifier Train Mode. This process is recommended for a system administrator, as it shortcuts the process of having to approve the invoice using the Learnset Manager application.

12.1.3.1 Prepare to configure Supervised Learning Workflow in Designer

Before configuring Verifier Train Mode for learning, complete the following prerequisite steps:

1. In the project file, in **Settings** dialog, select the **Verifier Train Mode** tab.
2. On the **Verifier Train Mode** tab, select **Enable automatic supervised learning workflow for this project**.
3. Select **Apply local classification and extraction automatically**.
4. In the **Put document to local learn-set** area, select **Only if adding activated by a user**.
5. In the **Learn new document area**, select **Only by user request**.
6. Select **Save all path information relative to the project file**.
7. Click **OK**.
8. In the **Classification field** list, select **VendorASSA**.
9. On the **Train Mode** tab, in the **Train Mode** area, select **Add trained documents to the learn set**.
10. In the **Learn Set Manager** area, enter the location for the Base directory.
11. In the **Backup Settings** area, select **Always automatically create a learn set backup before applying learning**.
12. Ensure that the pool against the VendorASSA field is configured and displaying a green light.

12.1.3.2 Train a Document

When completing header data, be sure to click the correct text on the image of the document to copy it into the field, so that the system knows where the data was selected on the document. The new class is not trained properly if the data is keyed in manually. If the document requires line items to be extracted, these must be trained using the AP Project Table Extraction tool, even if they were extracted using the generic line item extraction.

To train a document, complete the following steps:

1. Navigate to the problem document in the batch.
2. In the **Definition Mode** fields view, analyze the document.
3. On the toolbar, click **Verifier Train Mode**.
4. On the toolbar, click **Add to learnet**.
5. Verify the document.
6. If a field contains an incorrect extraction result and the value is not in the document, navigate to the portion in the document where the incorrect extraction result is located, right-click it, and in the list, select **Clear Candidate Assignmen**t from the context menu.
7. To move to the next document in the batch, press **Enter**.
8. To train the class, on the toolbar, click the **light bulb**.

9. Save the project file, and to display the new class, navigate to the class tree in **Definition Mode**.
10. Test the new class with the document you used for creating the class, and then with other documents from the same vendor or of the same format.

12.1.4 Adjust Field Settings

Adjusting the field settings can solve many extraction problems encountered after building a class using SFW. Adjusting the field settings invloves adjusting candidate format strings and definitions and adjusting field confidence and distance values. The following sections contain the instructions for adjusting field settings.

12.1.4.1 Adjust Candidate Format Strings and Definitions

The generic format string is `?[2-16]`, which means that a valid candidate for the invoice number is permitted to be any sequence of alpha or numeric characters between 2 and 16 characters in length. In the Ignore Characters field, the characters of `.-/\_,` can also appear in an invoice number candidate in any position.

To adjust the format strings and definitions, complete the following steps:

1. On the **Class View** page, double-click the new class to display the fields.
2. Navigate to the **Invoice Number** field and display the field properties.
3. From the **Available Templates** drop-down list, select the **InvoiceNumber** template.
4. Click **Copy Template**.
5. Adjust the settings that are appropriate for you business needs. These setting only apply to the new document class and do not have a wider effect on the project.

   Important: Do not change the format settings fields on the Invoices class. Doing so may have a adversly affect the generic extraction results.
6. On the **General** tab, make any necessary changes.
7. Save the project.

12.1.4.2 Adjust Field Confidence and Distance Values

In instances where the word count is already sufficient, the distance between words can be too small and cause the information in fields to be truncated or added to the end of text in other fields. The default distance inherited from the `Invoices` class level is `2.5`. Increasing this distance can help solve this problem. The following instructions use an invoice field that places a date at the end of the invoice number.

To adjust field confidence and distance values, complete the following steps:

1. Navigate to the field settings for the number for the new class, then select the **InvoiceNumber** template, and copy it.
2. Set the default generic format string to `#[7-9]`. This denotes any number between 7 and 9 characters in length.
3. Save the project.

With a more exact format string applied, the invoice number is now extracted correctly for the class without jeopardizing the extraction results for any other documents.

12.2 Add a Custom Script

Adding a custom script must always be the last resort for the correction of extraction problems.

Usages of the custom scripts include the following:

- Correcting OCR issue where the correct result is always known for a given vendor.
- Defaulting mandatory field values which the vendor does not actually state on the invoice.
- Improving line item extraction for vendors who present the information in a way that is not supported by the Table Extraction engine.

12.2.1 Correct OCR Issues

To correct OCR issues, complete the following steps:

1. Navigate to Designer Definition Mode (classes view), and then right-click the custom class name and select **Show Script**.
2. Insert the `InvoiceNumber_Format` script.
3. On the **Object** menu, select **InvoiceNumber**.
4. On the **Procedure** menu, select **Format**.
5. Set the `pField.Worktext.Text` parameter to **My Value**.
6. Save the project.

12.3 Configure Automatic Extraction for Custom Fields

AP Project includes five custom fields (`Custom1` to `Custom5`) which can be used for the extraction of data in addition to what is offered by the standard fields within the project.

It is possible to configure a level of automatic extraction for these fields using options within the AP Project database at the client profile level.

12.3.1 Prerequisites and Conditions for Automatic Extraction

The following prerequisites and conditions are required for automatic field extraction to function:

- Automatic field extraction works only for fields `Custom1` to `Custom5`.
- Automatic field extraction can only be carried out on documents that are initially classified either to the `Generic` class, or a related sub-class.
- Analysis templates `Custom1` to `Custom5` must exist within the project – these are delivered as standard and must not be deleted, in case an error is raised during document processing.
- Analysis templates `Custom1` to `Custom5` must be assigned to their respective custom fields for automatic extraction to take place. The project file is delivered with this in place.

12.3.2 Set Up a Field Analysis Profile

The field analysis profile is where formats for field extraction candidates are defined. Each field analysis profile has a unique ID – the `AnalysisProfileID`.

Field analysis profiles are set up in AP Project in the BRWAnalysisProfile table.

Each row in the table represents one format string, with the unique key formed by the `AnalysisProfileID` and the `IndexID`, so that many format strings can be entered for one

analysis profile. The table comes pre-populated with a sample entry for analysis profile 0. This is meant for illustrative purposes and must not be used.

New field analysis profiles must be created using an analysis profile ID of `1` and upwards. IndexIDs must be sequential.

Format strings need to be entered in the Format column. The `CompareType` column specifies the type of comparison that the string is used for. The options available are the following:

Compare type	Meaning
SIMPLE	Simple expression
REGULAR	Regular expression
TRIGRAM	Trigram compare method
LEVEN	Levenshtein compare method
STRINGCOMPARE	String compare method

If any other entry is made into the `CompareType` column, the system uses an internal default of SIMPLE.

For each format string, it is also possible to configure the `IgnoreCharacters` which is a non-comma-separated list of characters that may appear in the candidate in any position. However, these are not explicit in the format string definition.

12.3.3 Set Up a Field Evaluation Profile

Field evaluation profiles are defined in the BRWEvaluationProfile table.

An evaluation profile is where key words and key phrases used for evaluating candidates generated by the field analysis profile are defined.

Each row in the table represents one key word or phrase with the unique key formed by the `EvaluationProfileID` and the `IndexID`, so that many key words can be defined per evaluation profile. The table comes pre-populated with a sample entry for evaluation profile `0`. This is meant for illustrative purposes and must not be used.

New field evaluation profiles need to be created using an evaluation profile ID of `1` and upwards. IndexIDs must be sequential.

Each key word or phrase must be entered in the **Context** column. The system uses what is entered here to evaluate candidates by looking for matching context above or to the left of each candidate.

The `Strong` flag in the table can be used to indicate that the keyword is of high-relevance to the identification of the correct candidate. Hence, the system accords a higher weighting to those candidates that are nearer to this keyword.

12.3.4 Setting Up a Field Extraction Profile

Field extraction profiles are set up in the BRWExtractionProfile table. They control the way in which field candidates are generated and evaluated.

Each row in the table represents a unique extraction profile, which can subsequently be assigned to custom fields in the BRWFLD table. The key for each extraction profile is the extraction profile ID. The table comes pre-populated with a sample entry for profile zero. This is meant for

illustrative and database integrity reasons. It must not be used or changed for a production system.

New extraction profiles need to be created using an extraction profile ID of `1` and upwards.

12.3.4.1 Assign an Analysis Profile to a Field Extraction Profile

In order to generate candidates for a custom field, the field extraction profile must have an analysis profile assigned, which controls the generation of candidates. This is accomplished by populating the **AnalysisProfileID** column in BRWExtractionProfile with the relevant analysis profile ID. This column is mandatory, and it defaults to `0`. If an invalid analysis profile ID is entered, extraction fails for the document.

12.3.4.2 Additional Analysis Settings in BRWExtractionProfile

The BRWExtractionProfile table includes additional configuration settings to refine the generation of candidates over and above the entries against the analysis profile. These additional settings apply globally to all defined format strings:

- The **RemoveNoNumber** column is set to `TRUE` if the system must automatically remove all candidates generated by the format string(s) that do not contain a numeric character.
- The **OverwriteWithSearchString** column must be set to `TRUE` if an extracted field value is overwritten with the original Levenshtein or String Compare search string that is used to generate the field candidate. This mitigates any OCR problems if the field content is always a fixed value.
- The **Distance** column value represents the degree of *fuzziness* that the system needs to apply when generating candidates based upon the format strings entered against the analysis profile. A value of 0 denotes an exact match to the format string.A value of 1 means that no match is needed at all. The recommended value here is 0.3, which denotes that up to a 30% variance between format string and candidate is permitted. If a value less than 0 or greater than 1 is entered, the distance is set as 0.3.
- `MaxWordCount`, `MaxWordGap`, and `MaxWordLen` denote the maxiumum number of OCR words that are permitted to form a candidate., The maximum gap between OCR words (measured in millimetres) and the maximum length of the candidate (also measured in millimetres).
- The `CaseSensitive` flag controls whether candidates are generated in a case-sensitive fashion using the format strings on the analysis profile.`KeepSpace` controls whether spaces between individual OCR words need to be retained within the text of the generated candidate. It is usually recommended that both these options are set to `FALSE`.

It is also possible to restrict candidate generation to specific regions on the document.

In order to do this, the **UseRegions** column must be set to `TRUE`. From there, the columns **UseFirstPage**, **UseLastPage** and **UseSubseqPage** must be set to `TRUE` if candidates are generated on the first page, last page, and all the pages in between respectively. For example, if candidates are generated only on the first page, then **UseFirstPage** must be set to `TRUE` and **UseLastPage** and **UseSubseqPage** are both set to `FALSE`.

For each of the three groupings, it is also possible to restrict candidate generation to a specific region using the **Top**, **Bottom**, **Left**, and **Right** options. These values are expressed as percentages and are used for specifying the boundaries of a rectangular area where candidates are generated.

For example, a value of `0` in **FirstTop** means that the top most boundary of the rectangle is the very top of the first page of the document; a value of `20` means that the top most boundary is 20% down the length of the page. A value of `100` in **FirstBottom** means that the lower most boundary

of the rectangle is right at the bottom of the first page (100% of the way down), a value of `80` means that the rectangular area stops 80% of the way down the page.

Similarly, a value of `0` in **FirstLeft** means that the left most boundary of the rectangular area is the far left of the first page of the document. A value of 20 means that the search area starts 20% across the width of the page. A value of `100` in **FirstRight** means that the right-most boundary of the rectangular search area is at the far right of the page (100% of the way across).A value of `80` means that the rectangular area stops 80% of the way across the page.

If a value entered into these columns is less than `0`, then the system considers the value as `0`. Conversely, if a value entered is greater than `100`, the system considers the value as `100`.

12.3.4.3 Assign an Evaluation Profile to a Field Extraction Profile

The assignment of an evaluation profile, which gives weighting to candidates based on key words and phrases, is optional. If no evaluation profile is assigned, a value of `0` needs to be entered into the **EvaluationProfileID** column in the BRWExtractionProfile table.

Instead, candidates can be given a base weighting. This means that all candidates generated for a field are at least accorded the value specified in the **BaseWeighting** column of BRWExtractionProfile. This approach can be useful if the field value extracted is uniquely distinct on the document based upon its format.Any candidate that is generated can only be the correct value for the field. The base weighting is entered as a percentage. So a value of `50` means that the weighting of all generated candidates is increased by `0.5`.

If the keyword approach is required, an evaluation profile is assigned to the extraction profile by populating the **EvaluationProfileID** column so the system knows how to evaluate each candidate. An invalid entry in this field – that is, an evaluation profile that does not exist in the BRWEvaluationProfile table –causes the extraction to fail for the document.

When the system searches the document for key words and phrases set up in the evaluation profile, it searches for them in a *fuzzy* manner. The extent of the *fuzziness* is controlled by the **EvaluationDistance** column. A value of `0` denotes an exact match to the key word or phrase.A value of `1` means that no match is needed at all. The recommended value here is `0.3`, which denotes that up to a 30% variance is permitted for the key word or phrase. If a value less than `0` or greater than `1` is entered, the evaluation distance is set as `0.3`.

12.3.4.4 Assign a Field Extraction Profile to a Custom Field

Field extraction profiles are assigned to fields per profile ID in the BRWFLD table via the **ExtractionProfileID** column. A value of zero denotes that no automatic field extraction is carried out for the field. Any other values entered here must be valid within the BRWExtractionProfile table. Otherwise, extraction fails for the document.

13 Golden Tax

Golden Tax is the name given to a government project in mainline China aimed at providing an integrated, nationwide VAT monitoring system. As such, when a VAT invoice is generated by a vendor, it is done so via government-certified Golden Tax software so that the VAT aspect of the invoice is recorded and monitored centrally. All companies and individuals are required by law to raise domestic VAT invoices in this manner.

When an invoice is received by a buyer, key details stated on the invoice are validated against the central Golden Tax database to affirm that the invoice is genuine. Only invoices that validate against this central database may be included in a company VAT return for which a VAT refund from the government may be due. Hence, it is in the buyer's interest to ensure that only Golden Tax-certified invoices are being processed.

13.1 Use AP Project with Golden Tax

AP Project has the ability to extract the relevant details from standard, domestic VAT invoices that are required for the purposes of the Golden Tax validation process. These details can subsequently be included in a standard export, such as XML, database export or CSV file export, and passed to the relevant systems downstream.

The relevant fields are as follows:

- Invoice code (BRWFLD name = InvoiceCodeCN)
- Invoice password (BRWFLD name = InvoicePasswordCN)
- Invoice number
- Invoice date
- Invoice total amount
- Invoice tax amount
- Bill-to VAT registration number
- Vendor VAT registration number

Invoices relevant for the Golden Tax process are identified using classification. That is, they are classified initially to the `Invoice_CN` class, before being returned to the parent `Invoices` class. It is also possible to classify documents manually into the `Invoice_CN` class, or to re-classify invoices back to the `Generic` class in the event of a misclassification.

In the BRWFLD table, it is now possible to specify whether a field is needed or not needed based upon the initial document class.Hence the Golden Tax specific fields and corresponding validation rules which are particular to domestic VAT invoices can be configured as separate entries in BRWFLD relating exclusively to the `Invoice_CN` class.

Class-specific rules are configured via specifying the initial classname in the **ClassName** column in BRWFLD. Hence, for any given field, if a class-specific rule exists, there are two rows in BRWFLD for the profile ID, once for the regular `Invoices` class, and one for the specific class. When reading the BRWFLD table, the system looks for a class-specific entry. If one exists, it is used.If a class-specific entry does not exist, the entry for the `Invoices` class is used. If none exist, the field is considered inactive and not relevant for the profile ID.

Typically, the invoice number, date, total amount, and tax are already mandatory fields for any AP Project project, but, via configuration, the additional fields, namely invoice code, invoice password, the bill-to VAT registration number and the vendor VAT registration number can all be made mandatory with the relevant formatting and validation rules added. The system

attempts to extract all of these field automatically for documents classified to the `Invoice_CN` class.

The default AP Project database install, or an upgrade from a previous database version, add suggested standard formatting and validation settings for these additional four fields. It also includes entries in the BRWEXPHeader table so that the additional four values can be included in XML or DB output. New literals are also available for CSV file output.

13.2 Special Configuration for the Invoice Password

The invoice password, which appears as a four-line block in the top right hand corner of a standard Chinese VAT invoice, is subject to additional formatting rules. The invoice password is a string of either 84 or 108 characters, consisting of numbers and special characters.

In the BRWTAXCONFIG table, it is possible to configure the permitted special characters (default `+<>-*/`) using the parameter `CNPasswordSpecials`, and also the permitted length as a comma-separated list of possibilities using the parameter `CNPasswordValidLengths`. The `CNPasswordValidLengths` defaults to `84,108`, indicating that the field value must be either 84 or 108 characters in length.

Appendix A Tax Configuration for Countries Without Tax Jurisdictions

AP Packaged Project uses a lookup table for the purpose of determining and validating tax codes for purchase order-based invoices prior to posting them in the downstream ERP system. It is appropriate for use with countries where the tax rate is determined by the actual tax code itself, rather than an associated tax jurisdiction code as used in the US, Canada and Brazil.

The tax table is installed into either an Oracle or SQL Server database using an install SQL script that we provide. Usage of the tax table is activated for a profile ID within the BRWTAXCONFIG table.

Tax Table Structure

The columns in the tax table are as follows:

Column name	Type	Description
Country	String	Country code for the country to which the tax code applies. It represents the country in which the company code is legally registered. For example, GB, FR, CH, DE, and so on. This column must be populated for all records in the table.
TaxCode	String	Tax code
ShipTo	String	Ship-to country code This is the country code for the country to which the goods were shipped. This is compared by the system either to the country of the company which received the goods or to the country of the plant set on the relevant purchase order line item, depending on the configuration specified in the TAX section. If the ship-to country is the same as the company code country, then it must be entered as it is. If it is different, but is an EU member state, it must be set to EU. If it is a non EU-member state, it must be set to XX. This column must be populated for all records in the table.
ShipFrom	String	Ship-from country code. This is the country code for the country from where the goods were shipped. This is compared by the system to the country of the order-from vendor. If the ship-from country is the same as the company code country, then it must be entered as it is. If it is different, but is an EU member state, it must be set to EU; if it is a non EU-member state, it must be set to XX. This column must be populated for all records in the table.
Service	"X" or blank	This denotes that the record in the tax table is relevant for a service, as opposed to a material line item.
MaterialGroup	String	Purchase order material category/type/group or class.
MaterialNo	String	Purchase order material number.
VendorID	String	Order-from vendor ID.

Column name	Type	Description
Percentage	Number (two decimal places)	Percentage rate associated with the tax code (for example, 17.5, 19.6, 10, 18, and so on). This value is compared with the tax rate captured at the line-item level to support mixed tax rates on a single invoice, or, if no rate is captured, then it is compared to the overall tax rate for the invoice as a whole. When populating the tax table, 999 denotes a blank or unknown percentage. If the percentage is zero, then 0 must be entered.

Tax Table Access Sequence

At runtime, AP Packaged Project uses an access sequence to interrogate the tax table in order to find the correct tax code for each invoice line.

This step is not carried out if the system is configured to use the purchase order line tax code in all cases and that tax code is populated or if the tax table is not populated with any entries relating to the country of the invoice company code.

The following steps indicate the access sequence:

1. Check for combination of vendor, material group, ship-to, and ship-from.
2. Check for vendor and combination of ship-to and ship-from.
3. Check for material number and combination of ship-to and ship-from.
4. Check for material group and combination of ship-to and ship-from.
5. Check for combination of percentage, ship-to, ship-from, and service indicator.
6. Check for combination of ship-to, ship-from, and service indicator.
7. Check for combination of ship-to, ship-from, and tax percentage.
8. Check for ship-to and ship-from combination (default tax code for the country).

The system begins at step 1. If a tax code is found for that step in the access sequence, the corresponding tax code is used; if not, the system moves to step 2, and so on.

If, at the end of step 8, no tax code can be determined, then the invoice is considered to be an exception for reasons of a missing tax code.

Tax Code Validation

Once tax codes for all invoice lines have been determined, the system uses the percentage rate against each tax code (except in cases of foreign tax being charged) to compare what the tax must be using those codes against what has actually been billed on the invoice.

If the amounts are outside of the configured tolerance, the invoice is parked.

If any single invoice line has been allotted a foreign tax code, which is determined by the ship-from country differing from the company code country, then the entire tax amount on the invoice is booked to that tax code.

If the tax table contains no entries for the country of the invoice company code, no validation is carried out.

Populating Tax Table

The following table shows an example of how the tax code table might typically be populated for the UK (Country Code = GB):

Country	TaxCode	VendorID	ShipTo	Ship From	Service	Material Group	Material No	Percentage
GB	V0		'GB	GB				0
GB	V1		GB	GB				17.5
GB	V3		GB	EU				999
GB	V4		GB	EU	X			999
GB	V6		GB	GB				8
GB	V9		GB	GB				15
GB	VF		GB	XX				999
GB	V9		GB	GB				999

Using this configuration, the system behaves as follows as a result of each record in turn.

If it is a domestic transaction, and no tax is charged, then tax code V0 is always used.

1. If it is a domestic transaction, and tax at the rate of 17.5% is charged on the invoice, then tax code V1 is always used.
2. If the goods are shipped from an EU member state country to the UK, then tax code V3 is always used.
3. If the invoice is from an EU member state vendor and the service is performed in the UK, then tax code V4 is always used.
4. If it is a domestic transaction, and tax at the rate of 8% is charged on the invoice, then tax code V6 is always used.
5. If it is a domestic transaction, and tax at the rate of 15% is charged on the invoice, then tax code V9 is always used.
6. If the invoice is from a vendor outside of the EU, and the goods are shipped to the UK, then tax code VF is always used.
7. If no specific item tax percentage is determined from the invoice, but it is a domestic transaction, then tax code V9 is always used.

Step 8 represents the default tax code for a combination of ship-to and ship-from countries. The most common non-zero tax code must be set against this record.

If an additional business requirement involved a new tax code (VS) which must be used in instances where a non-EU member state vendor submitted an invoice for a service carried out outside the EU, and where tax was charged at 0%, then the following entry must be added to the tax table:

Country	TaxCode	VendorID	ShipTo	Ship From	Service	Material Group	Material No	Percentage
GB	VS		XX	XX	X			0

Equally, if a requirement arose whereby a different tax code (DS) must be used in all instances involving a service carried out domestically irrespective of whether tax is charged or not, the following entry must be added to the tax table:

Country	TaxCode	VendorID	ShipTo	Ship From	Service	Material Group	Material No	Percentage
GB	DS		GB	GB	X			999

Handle Special Cases for Tax Codes

The tax table provides functionality for handling special cases where a specific tax code is required.

The table permits the user to perform the following actions:

- Allocate a specific tax code to a specific vendor irrespective of the type of transaction and the rate of tax charged.
- Allocate a specific tax code for a specific material irrespective of the rate of tax charged.
- Allocate a specific tax code for a specific material group irrespective of the rate of tax charged. This can be used in instances where the tax against an individual invoice is not VAT reclaimable, or only partially reclaimable. For example, when all goods and services provided by vendor 12345 are only 50% VAT reclaimable, then a 50% VAT reclaimable tax code (VR) must always be used.

The corresponding tax table entry must appear as indicated in the following table:

Country	TaxCode	VendorID	ShipTo	Ship From	Service	Material Group	Material No	Percentage
GB	VR	0000012345	GB	GB	X			999

By creating entries that include a combination of the vendor, material number, and the material group, these instances can be handled as the access sequence checks for these entries before selecting a tax code based solely on the type of transaction (for example, service) and the tax rate.

It is also possible to leave the tax code column blank for such special cases which must always force those invoices to exception in a downstream process for someone to review.

For all entries in the tax table relating to special cases, the ship-to and ship-from countries must continue to be populated.

Appendix B Tax Configuration for Countries Using Tax Jurisdictions

This appendix contains the steps for completing the tax configuration section for ERP systems and countries that use tax jurisdiction codes. If tax jurisdictions codes are used, the system requires the following items per invoice line in order to create a invoice:

- The invoice line item tax code
- The invoice line item tax jurisdiction code

Under this ERP system configuration, which is applied at the country level, such as the US, Canada, India, and Brazil, where the tax rate is calculated based on the part of the country where the item was bought or consumed, the tax jurisdiction code represents the rate of tax applied to the purchase. The tax code represents whether the item is taxable or nontaxable as well as serving as an instruction on how the ERP system must process the tax.

Often the rates connected to tax jurisdiction codes are held within a specialist tax application external to the ERP system such as Vertex or Taxware. Tax jurisdictions are configured by profile ID in the BRWTXJCodes table.

Basic Tax Configuration Example
The following is a sample configuration for a US-based company. In this example, each line on the purchase order read from the ERP system has been allotted a tax code. Code I0 denotes a tax-exempt item and code I4 denotes that the item is taxable.

When the invoice arrives, it is for a single-line item. The line pairing routine successfully identifies that line item and the system proceeds to acquire the tax code. In this example, the following are the basic scenarios for acquiring the tax code.

- If the purchase order line has a tax code of I0, and no tax is charged on the invoice, the system must use tax code I0 for the invoice line.
- If the purchase order line has a tax code of I0, and the vendor is charging tax, the tax code must remain at I0 for the invoice line, and the vendor must be short-paid the tax they are billing, such as the total amount of the invoice must be the total amount minus the total tax amount.
- If the purchase order line has a tax code of I4, and no tax is charged on the invoice when tax is expected, the system must use a tax code for the invoice line that self-assesses the use tax for payment to local tax authorities (code U1).
- If the purchase order line has a tax code of I4, and tax is charged on the invoice, the invoice line tax code must remain at I4, and the tax must be paid to the vendor as billed.

For the profile ID zero, the following three tax group entries would be created and the columns configured in the BRWTXJCodes database table:

For entry 1 (I0)

Column in the BRWTXJCodes table	Value
ProfileID	0
Index	1
TaxCode	I0
Country	US
PayTaxAsBilled	FALSE
ShortPayIfTax	TRUE

For entry 2 (I4)

Column in BRWTXJCodes table	Value
ProfileID	0
Index	2
TaxCode	I4
Country	US
IfNoTax	U1
PayTaxAsBilled	TRUE
ShortPayIfTax	FALSE

For entry 3 (U1)

Column in BRWTXJCodes table	Value
ProfileID	0
Index	3
TaxCode	U1
Country	US
IfNoTax	FALSE
PayTaxAsBilled	TRUE
ShortPayIfTax	0

The tax jurisdiction code passed to the invoice line item must always be set to the tax jurisdiction code set against the purchase order line item. If the invoice has multiple line items, and if tax is being billed and all tax codes selected for the invoice lines carry a short-pay instruction, the vendor is only short paid the tax.

If the invoice has multiple line items, the vendor is charging tax. Any single invoice line item tax code carries an instruction to pay the tax as billed, the system books the entire invoice tax amount to the ERP system using the pay-tax-as-billed tax code in question.

Default Purchase Order Tax Code
If a tax code is not present on the purchase order line item, the system is able to default a tax code based on the purchase order line account assignment category.

To enable the system to default a tax code based on the purchase order line account assignment category in the **BRWTXJCodes** table, in the **AccountAssignmentCategories** column, add assignment categories as a comma-separated list for the desired tax code.

Configure State-dependent Tax Code
You can configure a specific tax code for a specific ship-to state. For example, if the purchase order line tax code is I4, but it needs to be changed to U1 if the tax is not chargeable except for a vendor who is based in a state that collect taxes for these transactions.

To configure state-dependent tax codes, complete the following steps:

1. Open the **BRWTXJCodes** table.
2. Set the **ProfileID** column to `0`.
3. Set the **Index** column to `2`.
4. Set the **TaxCode** column to `I4`.
5. Set the **Country** to `UD`.
6. Set the **IfNoTax** column to `IU`.
7. Set the **PayTaxAsBilled** column to `TRUE`.
8. Set the **ShortPayIfTax** column to `FALSE`.
9. Save the changes.
10. Open the **BRWTAXCONFIG** table.
11. Select the **Plant validation** option, and point it to a populated data source in the table.
12. Save the changes.

Note If both `VendorState` and `ShipToState` are populated, then only that combination triggers use of either the `IfTaxState` and `IfNoTaxState` codes.

Appendix C Configure Invoice Type Field

The invoice type field represents whether the invoice is related to purchase order or not. This controls whether the purchase order number and line item fields are mandatory, and how the document is handled at the point of export.

The system configuration determines the initial setting for this field, as well as how the circumstances under which this default setting must be changed.

The following sections contain three typical example business requirements and the corresponding settings in the BRWITY table, BRWSRC table, and the IMP section of the `<project>.ini` file.

Business Requirement 1
The following example is related to an invoice type that must be `NO-PO` unless a purchase order is found on the document.

Set the parameters listed in the following table in the BRWITY database table:

Column name	Value
ProfileID	0
DefaultValue	NPO
SetByVendor	FALSE
SetByVendorCCExceptions	
POValue	MM
NPOValue	FI
SetToPOIfPOFound	TRUE
SetToPOIfValidPOFound	FALSE

1. To switch the invoice type to PO based on whether it exists in the PO validation database system, set the parameters listed in the following table in the BRWITY database table:

Column name	Value
SetToPOIfPOFound	FALSE
SetToPOIfValidPOFound	TRUE
SetToPOIfPOPopulated	FALSE

2. If the invoice type must always be set to `PO` if the purchase order number field has content, whether captured automatically or entered by a user in Verifier, then set the `SetToPOIfPOPopulated` column to `TRUE`.

Business Requirement 2
The following example is related to an invoice type that defaults to `PO`, but must switch to `NO-PO` if no purchase order number is found and the vendor is permitted to supply invoices without a purchase order.

Set the parameters listed in the following table in the **BRWITY** database table:

Column name	Value
ProfileID	0
DefaultValue	PO
SetByVendor	TRUE
SetByVendorCCExceptions	
POValue	MM
NPOValue	FI
SetToPOIfPOFound	TRUE
SetToPOIfValidPOFound	FALSE

Settings in the BRWSRC table are the following:

Column name	Value
InvoiceType	XXXXXX

1. In the `InvoiceType SRC` setting above, `XXXXXX` represents the name of the **vendor ASSA** field column to which the invoice type value is passed from the vendor extract file. Map the value of this component for both `PO` and `NO-PO` invoices to the **POValue** and **NPOValue** columns in the BRWITY table. In the example above, the system expects to find either MM or FI in the ASE column.

In the vendor master extract, the field denoting the invoice type is determined based on business rules and can typically be set by the following.

- The vendor industry key, `NO-PO` invoices typically come from utility, car hire, and hotel vendors.
- Whether the vendor has a purchasing view created.
- The vendor account group.
- The vendor classification.

2. To restrict the vendor invoice type determination on a company-code by company-code basis in the **BRWITY** table, set the **SetByVendorCCExceptions** column to `1000,2000`.

Note If the `SetByVendor` parameter is set to `YES` and the invoice is meant for company codes 1000 or 2000, then the system does not apply the vendor invoice type preference. If the `SetByVendor` parameter is set to `NO`, then the system applies the vendor invoice type preference only if the invoice is meant for company codes 1000 or 2000.

Business Requirement 3

The following example is related to an invoice type that is determined at the point of scan and passed through the document file name.

1. Set the parameters listed in the following table in the BRWITY database table:

Column name	Value
ProfileID	0
DefaultValue	PO
SetByVendor	FALSE
SetByVendorCCExceptions	
POValue	MM

Column name	Value
NPOValue	FI
SetToPOIfPOFound	FALSE
SetToPOIfValidPOFound	FALSE

2. Open the **<project>.ini** file.
3. Navigate to the **IMP** section.
4. Set the following parameter:

   ```
   IMP_VL_InvoiceType=COMPONENTX
   ```

5. The InvoiceType SRC setting above represents the underscore-separated component of the document fil ename where the invoice type is passed from the scanning step. Map the value of this component for both `PO` and `NO-PO` invoices to the **POValue** and **NPOValue** columns in the BRWITY table. In the example above, the system expects to find either `MM` or `FI` in the **ASE** column.

Appendix D Configure the Vendor ID Field without Using Partition

This section describes how the vendor ID field can be configured within AP Project. The system uses the Associative Search engine to determine the vendor ID. This mandates that each vendor at a single address must have a unique identifier that is either numeric or alphanumeric.

Configuration options are available to ensure compatibility with downstream Enterprise Resource Planner (ERP) systems for the following scenarios.

- A standard vendor ID is used.
- A single vendor at a single address is denoted by both a vendor ID and site ID in tandem.
- An external and an internal vendor ID field is used.

Configure a Standard Vendor ID Field
This is the most basic configuration option to implement and must be used when the ERP system provides a single field identifier for a single vendor at a single address.

In this scenario, each record in the vendor extract supplied by the client, whether it is provided as a CSV file or within a database table, must represent a single vendor at a single address. Additionally, one column in the record needs to be a unique identifier. The generation of the vendor pool based upon the vendor extract fails if more than one record shares the same unique identifier.

To configure a standard vendor ID field, complete the following steps:

Note In this example, the vendor extract is provided as a CSV file.

---

1. Navigate to the directory that contains the Global project file.

Note If you are using UNC paths, the relevant directories must have the appropriate shares, which is usually full control, so that the system can perform the required read-write operations.

---

2. Create a new directory and name it `Pool`.
3. Open the `<project>.ini` file, locate the **ASA** section, and set the following parameters. If the unique identifier for each row in the vendor extract is numeric, then the `AlphaNum` parameter must be set to `No`. In all other cases, it should be set to `YES`.

```
ASA_VL_01_Class=Invoices
ASA_VL_01_Fieldname=VendorASSA
ASA_OP_01_AlphaNum=NO
ASA_OP_01_PoolRelative=NO
ASA_VL_01_PoolPath=\\MyComputer\Projects\Global\Pool
ASA_VL_01_PoolDirectory=Pool
ASA_VL_01_PoolName=Vendor
ASA_OP_01_FileRelative=NO
ASA_VL_01_ImportPathFilename=\\MyComputer\Projects\Global\vendor.csv
ASA_VL_01_ImportFilename=vendor.csv
```

4. If you are using supervised learning for the client deployment, populate the path to the Pool directory with a UNC path, and set the `PoolRelative` parameter to `No`. If you are not using supervised learning, and the pool directory is located in the same directory as the project file,

then the PoolPath can be left empty. But set the PoolRelativ parameter to Yes, and set the PoolDirectory parameter to the name of the pool directory.

5. If you are using UNC paths, ensure that the relevant directories have the appropriate shares (usually full control) so that the system can perform the required read-write operations.
6. If the vendor extract file is not in the same directory as the project file, then set the FileRelative parameter to No, and populate the ImportPathFileName with the UNC path to the vendor extract file. If the vendor extract file is located in the same directory as the project file, set the FileRelative parameter to Yes, and populate the ImportFileName with the name of the vendor extract file.
7. In the invoices class, navigate to the **VendorASSA** field, display the field settings, and complete the following substeps. The names of the columns shown in the field display are system-assigned. These names may not be indicative of the contents of the fields in the vendor extract file. SupplierID is always the first column, SupplierIndex is the second column, SupplierName is the third coulmn, Field4 the fourth column, and so on.
   - To import the pool, on the **File Import** tab, click **Import**.
   - To configure the search fields to identify the vendor, on the **Analysis** tab, in the **Search** column, select the **Vendor name**, **Street address**, **City**, **Zip/postal code**, **Vendor telephone numbers**, **Tax and VAT identifiers** boxes.
   - To select the column in the vendor extract that denotes the unique identifier for the vendor record, in the **ID** column, select the radio button.
8. To reimport the pool, on the **File Import** tab, click **Import**.
9. Set the **Class** settings to [*vendor name*] + underscore + [*vendor ID*] However, if the Supplier_ID column contains the unique vendor ID, and the SupplierName column contains the vendor name, which varies depending on the column order in the actual vendor extract file, then set the class to [SupplierName]_[SupplierID].

Note The entry in the class settings box denotes how the system names classes that are built using the supervised learning workflow. But, this field must be populated, irrespective of whether the supervised learning is being deployed for the client or not.

---

10. Configure the field settings. The field settings control how the vendor's address is displayed on the Verifier form. It is a multi-line field, and the first line must be set to the unique identifier for the record in the vendor extract. The file must have the following structure, but the structure may vary depending on your business needs. However, the first line must be set to the unique identifier:

```
[*Unique ID*]
[*Vendor Name*]
[*Street Address*]
[*City*]
[*State / Region*] [*Postal / Zip Code*]
```

11. If the SupplierID represent the unique record identifier, SupplierName represents the vendor name, Field4 represents the vendor street address, Field6 represents the vendor city, Field8 represents the vendor state, and Field7 represents the vendor zip / postal code, then configure the following parameters:

```
[SupplierID]
[SupplierName]
[Field4]
[Field6]
```

```
[Field8] [Field7]
```

12. The configuration of the vendor field is complete. A green light with the message Engine Is Ready must appear in the field status box. Save and close the project file.

Map Vendor Master Extract Fields

You must map the identification of each field in the vendor master extract to the relevant fields in the table BRWSRC. The system uses this mapping internally for the purpose of displaying the vendor search box and also for applying vendor specific business rules.

To map the vendor master extract fields, complete the following steps:

1. In the BRWSRC table, provide corresponding values for any of the following columns:

Column	Mapped Value
ID	SupplierID
SiteID	
Name	SupplierName
Address1	Field5
Address2	
City	Field6
Zip	Field7
State	Field8
Country	Field9

Note It is recommended that you map as many fields in the vendor extract as possible. The `IDparameter`, along with all the fields shown in the vendor address display box and the vendor's country of origin are required. The names of the mapped fields are case-sensitive.

Configure Vendor and Site ID Field

Use the instructions in this section if you have a single vendor at a single address that is represented in the downstream ERP system by a combination of a vendor ID and a site ID. AP Project permits the use of a composite key within the vendor file extract, so the two values need to be within a single column.

The steps for placing the values in a single column depend on whether the vendor ID and Site ID fields are numeric or alphanumeric. If the both fields are numeric, the vendor extract file needs to contain an additional column representing the combined vendor ID and site ID. The formula must be as follows:

```
Unique Identifier = (Vendor ID * 1000000 ) + Site ID
```

For example, if the vendor ID is 1234 and the site ID is 5678, then the unique identifier is as follows:

```
(1234 * 1000000) + 5678 = 1234005678
```

If either the vendor ID or the site ID contains alpha characters, then the formula for combining the two is as follows:

```
Unique Identifier = VendorID + [Separator] + Site ID
```

For example, if the vendor ID is A12345, the Site ID is 1000, and the designated separator is a hyphen (-), then the Vendor Extract file must have the following in the unique identifier column:

```
A12345-1000
```

The nominated separator is specified within the `AlphNumSiteSeparator` column in the BRVND table for a vendor. This must be populated and adhered to if you are using alphanumeric Vendor ID fields and Site ID fields. An error is displayed if this configuration is not followed, or more than one separator is found as part of a single unique identifier.

Therefore, in the Vendor Extract file, the following three columns are required:

- A column representing the combined unique identifier
- A column representing the vendor ID
- A column representing the site ID

Configure Vendor Field

To configure the vendor fields, complete the steps in the **Configure a standard Vendor ID field** section with the following variations:

1. For step 8, set the ID in the class name to the field that represents the standalone vendor ID. It must not be set to the field that represents the unique identifier for the vendor record.
2. For step 11, in the BRWSRC table, complete the following mapping:

   - Map the **Unique Record Identifier** field to the **ID** column.
   - Map the **SiteID** field to the **SiteID** column.
   - Map the **Vendor ID** field to the **ExternalID** column.

   For example, if the CSV file has allocated technical names of Supplier ID to the unique ID column, `SupplierIndex` to the site ID, and `Field 11` to the vendor ID component, then the mappings must be as follows:

Column in BRWSRC	Value
ID	Supplier ID
SiteID	SupplierIndex
ExternalVendorID	Field11

Configure an external vendor ID

Use the configuration steps in this section if the downstream ERP system differentiates between an internal and an external vendor ID by using an internal vendor ID at the database table level, but the user is presented with an external ID in the application itself.
If the client requires that the Verifier application follows this pattern and displays the external vendor ID to the user, then the vendor extract requires that the external vendor ID is included as a column, but the ERP system internal vendor ID is the unique identifier.

To configure an external vendor ID, complete the steps in the **Configure a standard Vendor ID field** section with the following variations:

1. For step 8, set the ID in class name to the field that represents the external vendor ID.
2. For step 11, in the **BRWSRC** table, complete the following mapping”

   - Map the **Unique Record Identifier** field to the **ID** column.
   - Map the **external vendor ID** field to the **ExternalVendorID** column.

Column in BRWSRC	Value
ID	Supplier

Column in BRWSRC	Value
ExternalVendorID	Field11

3. If the downstream uses an external vendor ID and a site ID, in the **BRWSRC** table, complete the following mapping:

   - Map the **Supplier** field to the **ID** column.
   - Map the **SupplierIndex** field to the **SiteID** column.
   - Map **Field11** to the **ExternalVendorID** column.

Column in BRWSRC	Value
ID	SupplierID
SiteID	SupplierIndex
ExternalVendorID	Field11

Note In the above example, **Field 11** represents the external vendor ID field, rather than the internal vendor ID displayed in the **Configure a standard Vendor ID field** section.

Work with Addresses in Non-Western Languages

If the vendor extract contains non-Western characters. For example, if vendor addresses are present in Russian, Bulgarian or Greek, then version 3 of the associative search engine must be selected. In the project OCR settings for Finereader, **Russian** must be added for Russian and Bulgarian; **Greek** must be added for Greek documents.

In version 5.2, if the thick client Verifier is being used, you must set the desktop locale to use the non-Western language as the language for non-unicode applications. This limits the display to one non-Western alphabet in thick client Verifier at a time.

Multiple non-Western language display within the Verifier application is only supported in case of thin client deployments.

CSV File Format Guidelines

Flat CSV files must be compliant with the following guidelines:

General Guidelines

- Each row in the file must represent a single vendor at a single address.
- Each row must include, as a minimum, columns that represent the vendor name, street address, city, postcode/zip code, and country.
- Each row in the file must have one column that is a unique identifier for that record and is common only to that row.
- Each row in the file must have an equal number of columns.
- The column separator must be a semicolon. Each column must have all semicolons removed.
- Each column must have all double quotes (“) removed.

Vendor Country

You must have a column which represents the vendor country included in the vendor extract file. This must always be the two-character International Organization for Standardization (ISO) code for that country. The following table contains a list of some common ISO codes:

Country	ISO code
United States	US

Country	ISO code
United Kingdom	GB
Germany	DE
Switzerland	CH

Intercompany Vendors
Any intercompany vendors included in the Enterprise Resource Planner (ERP) vendor master data must be excluded from the file, along with the employees who are set up as vendors. Intercompany vendors are handled in a different way within the solution.

Non-Western characters
If the CSV file includes non-Western characters, the file must have an encoding type of UNICODE. The system does not support ANSI or UTF-8 encoding.

Appendix E Configuring Miscellaneous Charges

Miscellaneous charges refers to the additional items a vendor may include on an invoice document that have to be booked with the primary invoice items.

Examples include a freight charge, a customs charge, an energy surcharge or an administration code.

Miscellaneous charges can appear at both the header and item level on an invoice.

At the header level, AP Packaged Project provides two fields for this purpose: `AmountFreightPrepaidAndAdded` and `AmountMisc`.

At the line item level, the **category** column identifies and classifies any miscellaneous charges captured that the vendor specifies as a line item on the invoice. This category is set automatically by AP Packaged Project drawing on the system configuration settings applied in the BRWMSCCategory table.

The handling of miscellaneous charges is dependent on the business rules of the client. The configuration options within the BRWMSCCategory permit table are as follows:

1. Miscellaneous charges to be booked as `unplanned delivery costs`. (i.e. a single, header-level amount field within the downstream ERP system)
2. Miscellaneous charges to be booked as a separate invoice line item with a specific line type. (Oracle Financials)
3. Miscellaneous charges to be booked as a direct general ledger account entry

These events occur during the line pairing operation carried out during document export. Line pairing must be activated in the BRWLPR table for miscellaneous charge processing to occur. If line pairing is deactivated, then any miscellaneous charges that appear on the invoice is outputted in the form in which they were captured.

Note The system always checks for the existence of a regular purchase order line item representing the miscellaneous charge before applying the configured processing option. If one is found, then this line item is used. Miscellaneous charges set up as regular purchase order line items are identified by the purchase order line item description and the extent to which it fits with the aliases configured as described in the *Assigning Line Items to a Miscellaneous Charge Category* section below.

Miscellaneous Charge Categories

Miscellaneous charge categories are configured within the BRWMSCCategory table.

Each row within the table for a profile ID represents a miscellaneous charge category. The category denotes the type of miscellaneous charge, such as freight, a customs charge, and so on.

The number of categories that must be configured is dependent on how you want to process these charges, and the number of miscellaneous charge categories is driven by the number of different processes you want to use.

For example, if all miscellaneous charges, irrespective of what they are, must be summed and booked as a single general ledger account entry in the downstream ERP system, then only one miscellaneous charge category needs to be configured.

However, if you want to book specific miscellaneous charges to a specific general ledger account depending on what type of charge it was, such as account 1000 for freight; account 2000 for customs charges; account 3000 for pallet charges, and so on, then there must be as many miscellaneous charge categories as the number of possible general ledger codes to book them against.

Within each group, a code and name is assigned to each miscellaneous charge category. In the Dynamic Verifier application, if a line item is identified as belonging to a particular miscellaneous charge category, the code set for that group in the system configuration is copied into the **category** column for that particular line item.

Assign header fields to a miscellaneous charge category
Each miscellaneous charge category can be assigned fields either at the header or line item level.

In the BRWMSCCategory table, the `HeaderField` parameter controls the mapping between a header field and the miscellaneous charge category.

The two standard header fields available for mapping are `AmountFreightPrepaidAndAdded` and `AmountMisc`. A sample configuration for mapping the freight field to the freight category for profile ID zero is seen as follows:

Column in the BRWMSCCategory database table	Value
ProfileID	0
Index	1
Type	FREIGHT
Code	F
HeaderField	AmountFreightPrepaidAndAdded

More than one header field can be mapped to a single miscellaneous charge category using comma separation as seen in the following table:

Column in the BRWMSCCategory database table	Value
HeaderField	AmountFreightPrepaidAndAdded, AmountMisc

The name of the WebCenter Forms Recognition header field is not case-sensitive.

Assign line items to a miscellaneous charge category
The assignment of line items to a miscellaneous charge occurs using the extracted line item description.

In the **Alias** column, a comma-separated list of keywords are entered to indicate that the line item belongs to the miscellaneous charge group, as seen in the following table:

Column in BRWMSCCategory table	Value
ProfileID	0
Index	1
Type	FREIGHT
Code	F
HeaderField	AmountFreightPrepaidAndAdded
Alias	FREIGHT,DELIVERY,CARRIAGE,UPS,TRANSPORT

The list of miscellaneous charge aliases is not case-sensitive.

Post miscellaneous charges as a specific line type
If the business rule for processing charges belonging to the miscellaneous charge category involves booking them as a specific line type in the downstream ERP, then the required line type must be entered in the **LineType** column.

For example, if the ERP line type for freight is FRT, the following must be populated:

Column in BRWMSCCategory table	Value
ProfileID	0
Index	1
Type	FREIGHT
Code	F
HeaderField	AmountFreightPrepaidAndAdded
Alias	FREIGHT, DELIVERY, CARRIAGE, UPS, TRANSPORT

If the line type value is populated, this overrides any further handling settings applied to the miscellaneous charge group. At the point of export, the miscellaneous charge is outputted as a line item.

Post miscellaneous charges as unplanned costs
If the business rule for processing charges belonging to the miscellaneous charge category involves always booking them as an unplanned cost in the downstream ERP system, then this is configured using the `AlwaysBookedToUnplanned` parameter.

For example, the following table contains the values for the columns in the BRWMSCCategory table:

Column in BRWMSCCategory table	Value
ProfileID	0
Index	1
Type	FREIGHT
Code	F
HeaderField	AmountFreightPrepaidAndAdded
Alias	FREIGHT, DELIVERY, CARRIAGE, UPS, TRANSPORT

At the time of export, the total value of all miscellaneous charges in every category that require handling as unplanned delivery costs are passed to an export database field using the export field name `UnplannedFreight in the BRWEXPHeader` table.

Post miscellaneous charges to a general ledger account
If you post miscellaneous charges to a general ledger account, the system creates a separate general ledger entry for the miscellaneous charge.

A general ledger entry consists of the following components:

- A general ledger account code representing the type of expense
- Cost object, such as a cost center, an internal order, a profit center, a project
- A tax code

To determine the general ledger account code, the system looks into a custom database table. If an entry exists for the invoice company code and the miscellaneous charge code, then this is the

general ledger account code that the system uses. If no entry exists, or the custom table lookup has not been set to `TRUE` in the **ValidateFromDB** column in the BRWMSC table, then the default general ledger code for the miscellaneous charge group is used.

The sequential derivation of the cost object is as follows:

- If the `GetCostObjectFromPO` parameter against the miscellaneous charge group is set to `TRUE`, the cost object is lifted from the first invoice line paired that uses either a cost center/profit center, an internal order or a project.
- If a profit center or cost center exists in the custom table, this is used as the cost object.
- The default cost center/profit center in the **DefaultCostCenter / DefaultProfitCenter** columns in the BRWMSCCategory table is used.

The sequential derivation of the tax code is as follows:

- If an entry exists in the custom table for the invoice company code and miscellaneous charge category, this tax code is used.
- The tax code is lifted from the first paired invoice line item.
- The default tax code set in the `DefaultTaxCode` parameter in the BRWMSCCategory table is used.
- For countries and ERP systems that use tax jurisdictions, the tax jurisdiction code is derived from the first paired invoice line.

If the general ledger account is not relevant for tax postings in the ERP system, a double asterisk (**) must be entered into the tax code column in the table. This has the effect of passing a blank tax code and a blank tax jurisdiction code downstream.

If the custom table lookup is activated, but communication does not succeed either due to missing or incorrect configuration or database unavailability, the export event for the document fails.

A sample configuration for posting miscellaneous charges as general ledger entries is shown in the following table. This must be entered as a row into the BRWMSCCategory table for the profile ID.

Column in the BRWMSCCategory table	Value
ProfileID	0
Index	1
Type	FREIGHT
Code	F
HeaderField	AmountFreightPrepaidAndAdded
Alias	FREIGHT,DELIVERY,CARRIAGE,UPS,TRANSPORT
LineType	
AlwaysBookToUnplanned	FALSE
AlwaysBookToPlanned	FALSE
ValidConditions	
AlwaysBookToGLAccount	TRUE
BookToUnplannedIfNoPlanned	FALSE
BookToGLAccountIfNoPlanned	FALSE
GLAccount	1000
GetCostObjectFromPOLine	TRUE

Column in the BRWMSCCategory table	Value
DefaultCostCenter	1000
DefaultProfitCenter	2000
DefaultTaxCode	I0

With the configuration above, if the custom table is not used, the system books the miscellaneous charge as a general ledger entry using general ledger account 1000 with the cost object and tax code from the first paired invoice line. If no cost object exists against the first paired invoice line, then cost center 1000 and profit center 2000 is used. If no tax code exist against the first paired invoice line, tax code I0 is used.

General Ledger Account Code table

To facilitate greater versatility when posting freight as general ledger account entries, a lookup table is available so that specific general ledger account codes, cost objects, and tax codes can be assigned on a company code-by-company code basis and a plant-by-plant basis incorporating the purchase order line type, if required.

The following table contains an example for populating the table for company code GB01, plant 1000, line type A and miscellaneous charge category F:

Company Code	Category	Line Type	Plant	GLAccount	CostCenter	Profit Center	TaxCode
GB01	F	A	1000	1000	2000	3000	I0

The **company code**, **plant**, **line type**, and **category** columns are mandatory and form the unique key for each record in the table.

If the general ledger account, cost object, and tax codes are to be set at a company code level rather than a plant level, a space must be entered into the **plant** column. If no line type is to be used, a space must also be entered into the **line type** column.

The system accesses the table according to the following sequence:

1. By company code, plant and line type
2. By company code and line type
3. By company code and plant
4. By company code alone

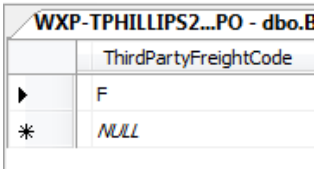
As soon as a matching record is found, the access sequence breaks and that record is used. The above must be considered when populating the table.

Third-party Freight

Within AP Packaged Project, a third-party freight invoice refers to a specific business scenario whereby an invoice is received from a vendor billing for freight. Yet that vendor legitimately quotes the purchase order number of another vendor (the material vendor), and it's against this material vendor's purchase order that the freight charge needs to be booked.

Freight invoices that do not fall into this category are handled as regular invoices.

During line pairing, the system books the net value of the invoice against the material vendor's purchase order according to the rules set against the miscellaneous charge group assigned to third-party freight vendors, which is set in the BRWMSC database table. This denotes that the rules must be derived from whichever miscellaneous charge group has been assigned a code of F.



If the rule is set to `post as unplanned`, the system also creates a zero-value debit invoice line against the first line item of the purchase order belonging to the material vendor.

If the rule is set to the `post against planned` condition types on the purchase order, these condition types must be goods receipted.

Add New Miscellaneous Charge Fields

The AP Packaged Project project file comes bundled with two fields for capturing miscellaneous charges at the header level, specifically the **AmountFreightPrepaidAndAdded** and **AmountMisc** fields.

The **AmountFreightPrepaidAndAdded** field is pre-trained to extract freight and transportation costs from invoices. The **AmountMisc** field is not trained, so there is flexibility to designate that field for a specific type of miscellaneous charge, such as a pallet charge or an energy surcharge. `UserExitAmountMiscPostEvaluate` is available for any custom script deemed appropriate to assist with the extraction of this field.

If a third field is required, it must be created at the Invoices level, and it must be called `AmountMiscXXX`, where `XXX` is a meaningful field descriptor. This ensures that the new miscellaneous charge field is included in the mathematical validation applied to the invoice header level amounts.

Additionally, a standard validation event for the new field must be added to the invoices class level containing a script to call the system header level validation function `fnValidateAmount`. This function also takes care of the field formatting.

Appendix F Options for Export Files

The following options are available in the BRWCSV table:

- The destination output directory.
- Whether one file must be written per document or per batch.
- Whether the file must be written out for PO or NON-PO invoices or both.
- The file extension used.
- The name of the file (either the entire document image file name, or just the URN component), along with an optional file prefix.
- Whether a copy of the original image must also be written to the output directory.
- The format of any date fields within the file with an optional separator.
- The fields outputted and the order of those fields in the file to a maximum of five lines.
- Whether line items must be outputted and what line item detail must be included.
- The delimiter used to separate the fields.

The following options are available in table the BRWEXP table:

- The name of the file (either the entire document image file name, or just the URN component).
- The file extension to be used.
- All XML tags used within the file.
- The format of any date fields within the file with an optional separator.
- Those fields that are to be outputted, and those that are not.

Appendix G Deactivating the ASE Fields

This section describes the steps required to deactivate Associative Search Engine fields within the project file.

G1. Delete the ASA Entries from the <project>.ini File

To delete the ASA entries from the `<project>.ini` file, complete the following steps:

1. Open the **<project>.ini** file.
2. Navigate to the **ASA** section.
3. Find the field that you want to deactivate and its associated class.
4. Either delete the group for the field, or comment each line for the group out with a single quote (') or hashs (#) symbol as the first character of the line.
5. Save the changes and close the file.

G2. Switch Off the ASE

To switch of the ASE, complete the following steps:

1. Open the **<project>.sdp** file using the WebCenter Forms Recognition Designer module.
2. Navigate to the class that holds the definition of the field and then to the field.
3. Display the field properties.
4. In the **Available analysis engine** list, select **No Analysis Engine**.
5. Save the changes and close the file.

Appendix H Configuring Intercompany Vendors

Intercompany invoices arise in instances where different subsidiaries of the same group of companies are invoicing one another. This is prevalent in large multinational corporations where a substantial proportion of the daily invoice volume can represent this type of transaction.

Intercompany invoices need to be handled separately from regular third-party invoices. The reason is that the presence of bill-to address information on a third-party invoice, coupled with the presence of intercompany vendor addresses in the vendor extract, can skew the recognition rates of third-party vendors. Therefore, the system could potentially propose an intercompany vendor as the best vendor for a third-party invoice because the intercompany vendor details represent a good fit to the bill-to address details provided on the invoice.

To avoid this issue, the solution provides a dedicated Intercompany class.

H1. Vendor Extracts

If intercompany invoices are processed, then two vendor extract files need to be provided.One file contains only third-party vendors and another contains only intercompany vendors.

The third-party vendor extract file is set up in the normal manner. The intercompany vendor extract file is set up in the same way, except that it is mapped to the VendorASSA field on the intercompany class level.

H2. Configure the Intercompany Classification

The instructions in this section are optional. If they are not completed, all intercompany invoices are classified to the Invoices class, and they stop in Verifier because the system is unable to find the correct vendor. If this occurs, you must reclassify each intercompany invoice manually.

The following examples provide guidelines for classification depending on your client circumstances. Select the appropriate classification strategy based on the following information:

- The volume of intercompany invoices.
- The variety of intercompany invoice formats.
- The extraction success on intercompany invoices through the generic learnset.
- The degree of document separation at time of scanning.

Example 1

In this example, intercompany invoices are separated from third-party vendor invoices at the point of scan. If intercompany invoices are separated from third-party vendor invoices and are scanned using a different scan job, use this method to classify them using the correct class.

If this applies to your business environment, you can use a script in the `UserExitPostClassify` user exit script to force the document to the intercompany node, thus guaranteeing 100 percent correct classification to the extent that the document was scanned correctly. In the event of a mis-scan, the document is classified to the `Invoices` class and a user has to reclassify it to the intercompany class manually.

The `fnGetFileName` function provides a simple method for extracting the file name without the file path and file extension. The name of the intercompany class is case-sensitive and is always referred to as Intercompany.

Example 2

You process a significant volume of intercompany invoices but there are only a handful of different formats.

If the number of intercompany invoice formats is total five or less, then an example of each format can be added directly to the classification learnset for the Intercompany class using the layout classification engine. It is recommended that you add at least three examples of each format.

It is not necessary to train a corresponding extraction learnset for the intercompany class as the class automatically inherits the extraction training from the prepackaged learnset held at the invoices class level.

However, if the generic extraction is not delivering acceptable extraction results, you can create an extraction learnset exclusively for the intercompany invoices at the intercompany level using the **Normal Train Mode** in the solution Designer module.

It is not possible to use the supervised learning workflow feature in conjunction with the intercompany classes. You can create and amend the intercompany classes only through the Designer module.

Example 3

You process a significant volume of intercompany invoices and there are a wide variety of different formats.

Large, multinational organizations with different subsidiaries or suborganizations located around the globe may experience high volumes of intercompany invoices in a multitude of different invoice formats.

If this is the case, then you must create new intercompany subclasses under the existing intercompany class to accommodate the different format examples required to build a classification learnset.

Depending on volumes and the success of extraction using the generic learnset, it may be advantageous to create a dedicated extraction learnset for each format. There is no restriction on the number of subclasses you can create though it is recommended that each has a title indicative of the formats of documents it is designed to handle.

Appendix I Configure Late Archiving

This section provides instructions for configuring late archiving.

I1. Configure Late Archiving With Mobius

AP Packaged Project can generate a Mobius-compliant list file that is imported into the Mobius archiving solution during document export. To configure late archiving with Mobius, complete the following steps:

1. Open the **BRWCSV** table.
2. Set the parameters listed in the following table:

Column in the BRWCSV **table**	**Value**
OutputCSVFile	TRUE
Index	1
OutputFile	TRUE
CombinedFilePerBatch	TRUE
Filepath	\\myMachine\MobiusImportFI
FileName	
FileType	TXT
FilePrefix	F1
Separator	
DateFormat	
DateSeparator	
InvoiceType	NPO
OutputImage	TRUE
FormatLine1	REPORT-ID=FIDOC
OutputCSVFile	TRUE
Index	1
OutputFile	TRUE
CombinedFilePerBatch	TRUE
OutputImage	TRUE
FormatLine1	REPORT-ID=FIDOC
FormatLine2	FILE=<%TND>,TYPE=TIF,"SECTION=<%ERP>"
FormatLine3	TOPIC-ID=SCANDOC,"TOPIC-ITEM=<%TNF>"
FormatLine4	TOPIC-ID=COMPCODE,"TOPIC-ITEM=<%CCO>"
FormatLine5	TOPIC-ID=VDRLINK,"TOPIC-ITEM=<%ERP>"
LineItem	

3. Optional. Create a separate list file for `PO` and `NO-PO` invoices, each in different directories, if required.

4. The document export fails if the Mobius list file cannot be written to the specified directory. To generate this list file for the export profile ID, in the *BRWEXP* table, set the **OutputCSVFile** column to `TRUE`.
5. Save the changes.

I2. Configure Late Archiving with WebCenter Forms Recognition

The reporting feature lets you store image data within its underlying database, which allows as an image archive.

To archive the image into the reporting database, complete the following steps in <project>.ini configuration file:

1. Set the **REP_OP_StoreImageInReportingTables** parameter to `Yes`.
2. To specify the table in which the image is to be stored, set the **REP_VL_ReportingDBImageTable** parameter to `BRWDOCIMAGE`.
3. To configure the archive URL, which is the template of a web address that brings up the image, set the **REP_VL_ArchiveURL=http://myarchivesystem.com?ID** parameter to `XXXXX`.

Note: The system automatically substitutes the `XXXXX` component of the URL with the unique image ID in the archive.

Appendix J Configure the VAT Number Compliance Check

AP Packaged Project provides a VAT registration number compliance check to satisfy an EU legal requirement that applies to invoices which charge value added tax.

This requirement states that, if VAT is to be charged on the document, it is incumbent upon the vendor to quote not only their own VAT registration number (the vendor VAT registration number), but also the VAT registration number of the party being invoiced (the bill-to VAT registration number).

Both VAT registration numbers must be valid for the same country, which is why VAT is generally most prevalent on domestic invoices.

If the VAT compliance check is activated, AP Packaged Project sends a document to Verifier if all of the following hold true:

- Tax is being charged on the document.
- Both vendor and the company code being invoiced are based in an EU member state country.
- One or both of the VAT registration numbers are not present on the document, or the registration numbers have not been captured automatically, or they do not share the same country prefix – the system may be configured only to require the VAT registration number of the vendor.

A further requirement is that, if VAT is to be charged, the currency of the invoice must be the local currency of the country whose VAT rates are being applied. In the example above, if the German vendor is to issue an invoice charging UK VAT, the invoice currency must be UK pounds sterling (GBP). If the invoice is issued in the local currency of Germany, such as Euros/EUR, then the invoice must quote either of the following:

- The UK pounds sterling equivalent of the VAT amount, such as the local VAT amount.
- The exchange rate between UK pounds sterling and Euros at time of issuing the invoice.

AP Packaged Project detects these and requires the user to enter either a local VAT amount or an exchange rate if no value is captured automatically. This check is carried out when a pair of valid VAT registration numbers is found. The currency of the company code is used as the currency in which either the invoice as a whole or the local VAT is quoted.

If the VAT table is used for capturing the invoice tax amounts, it is also possible to extend the VAT compliance check to verify that the VAT rates captured within the table are valid for the country where VAT is applied if the total invoice tax amount is greater than zero. The country where VAT is applied is derived firstly from the country code represented by the first two characters of an extracted vendor VAT registration number. If not available, the vendor country of origin is used.

J1. Configure VAT Compliance

To configure VAT compliance, complete the following steps:

1. Open the **BRWTAXCONFIG** database table.
2. Set the **ActivateVATCompliance** column to `TRUE`.
3. Save the changes.

J2. Configure Extraction

To activate the extraction of both the vendor and bill-to VAT registration numbers, complete the following steps:

1. Open the **BRWTAXCONFIG** table.
2. Set the **ActivateVATComplianceCheck** column to `TRUE`.
3. In the **VATCheckCompanyCodeExceptions** column, enter one or more company codes. Separate each country code with a comma.
4. Save the changes and close the file.
5. Open the **BRWSRC** table.
   To enable the system to extract the VAT registration number of the vendor, the VAT registration number must be populated within the vendor extract file/database used by the project.
6. In the **VATRegNo** column, enter the database column that contains the VAT registration number.
7. Save the changes and close the file.
8. Open the **BRWCCO** table.
9. If the company code lookup is to be against a database then the value of the database columnn **ValidateFromDB** should be set to `TRUE`.
10. To configure the connection group representing the connection string that points to the database containing the company code lookup table, set the **SQLConnectionGroup** column to `00`.
11. Configure the columns in the BRWCCO table with the following parameters:

Column in the BRWCCO **table**	**Value**
DBTableName	OFRCOMPANY
DBColumnName	COMPANYCODE
DBCountry	COUNTRY
DBCurrency	CURRENCY
DBVATRegNos	VATREGNO

12. Save the changes and close the file.
13. In the **OFRCOMPANY** table, map the technical names of the table and the associated columns to the remaining database lookup parameters.
    The values are not extracted if any of the following are true:
    - A vendor cannot be determined from the invoice.
    - A company code cannot be determined from the invoice.
    - The company code is listed as an exception company code.
    - The user has selected an invalid reason other than `NONE, THIRD PARTY FREIGHT, PO VENDOR <> INVOICE VENDOR or STOCK INVOICE`.
    - The registration numbers are not present in the document, or the document was of a bad quality due to which the numbers could not be read.

    Note: Multiple VAT registration numbers either per vendor or per company code are supported. These must be included in the relevant columns in the vendor extract or the company code table as a comma-separated list.

14. Save the changes.

J3. Configure VAT Compliance Validation

To configure VAT Compliance Validation, complete the following steps:

1. Open the **BRWSRC** database table.
2. Set the **EUMember** column to the location of the EU member state flag in the CSV file.
3. Set the **EUMemberAlias** column to `X`.
4. Save the changes.
5. Open the **BRWCCO** database table.
6. In the **DBCountry** column, enter the country of origin.
7. Open the **BRWCTR** database table.
8. Set the following parameters:

Column in the BRWCTR **table**	**Value**
ValidateFromDB	TRUE
SQLConnectionGroup	SQL connection group
DBTableName	OFRCOUNTRY
DBCountry	COUNTRY
DBCurrency	CURRENCY
DBEUMember	EUMEMBER
DBName	COUNTRYNAME
DBVATRates	VATRATES *

> **Note:** The **WFR_AP_Tables_Create_<*database*>.sql** script provided with the AP Packaged Project creates and populates a table called **OFRCOUNTRY** that provides data for all world currencies and their associated countries.

9. Save the changes.
10. Optional. To limit the VAT requirement to only check the VAT registration number of the vendor, open the **BRWTAXCONFIG** database table.
11. Set the **VendorVATCheckOnly** column to `TRUE`.
12. Save the changes.
13. To configure vendor VAT registration number-only check so that it only applies to certain company codes, in the **VendorVATCheckCoCodeExceptions** column, enter the company codes as a comma-separated list.

J4. Configure the VAT Rate Check

To configure the VAT rate check for the VAT table, complete the following steps:

1. Open the **BRWTAXCONFIG** database table.
2. Set the **CheckVATRates** column to `TRUE` and save the changes.
3. Open the **BRWCTR** database table.
4. Ensure that the company lookup is configured and the `DBVATRates` parameter points to the database column that contains the country VAT rates.
5. Save any changes.

J5. Configure Cross-Border EU VAT Registration Number Checks

For cross-border EU transactions, a vendor may zero-rate the value-added tax (VAT) charged on their invoice, but they must specify both their VAT registration number and the VAT registration

number of their customer on the document. Their customer is obliged to calculate and declare the VAT at their local rate to the local tax authorities as if the purchase had been made from a domestic vendor.

To configure the cross-border EU VAT registration number checks, complete the following steps:

1. Open the **BRWTAXCONFIG** table.
2. Set the **CheckVATCrossBorder** column to `TRUE`.
3. If there are company codes that do not require the cross-border VAT check, in the **CrossBorderCoCodeExceptions** column, enter the company codes as a comma-separated list.

The system sends a document to Verifier if all of the following conditions are true:

- Zero tax is being charged.
- Both the vendor and company code are domiciled in different EU countries.
- The vendor has a VAT registration number populated in the vendor extract.
- Either the vendor or VAT registration number or the bill-to VAT registration number is missing or has not been captured from the document.

Appendix K Activate the Payment, Delivery and Due Date Fields

In the standard version of the AP Packaged Project project file, the delivery note number, the payment reference, the due date, and the delivery date fields are not activated to maximize system performance for projects in which these fields are not needed.

To activate these fields, use the steps in the following sections.

K1. Activate the Payment Reference Field

To activate the **Payment Reference** field, complete the following steps:

1. In WebCenter Forms Recognition Designer, open the **<project>.sdp** file.
2. Navigate to **Definition Mode**.
3. To display the fields, click **Invoices**.
4. Navigate to the **PaymentReference** field, and display the field properties.
5. From the **Available Templates** list, select the predefined **PaymentReference** template.
6. Save the project.

K2. Activate the Delivery Note Number Field

To activate the **Delivery Note Number** field, complete the following steps:

1. In WebCenter Forms Recognition Designer, open the **<project>. sdp** file
2. Navigate to **Definition Mode**.
3. To display the fields, click **Invoices**.
4. Navigate to the **DeliveryNote** field, and display the field properties.
5. From the **Available Templates** list, select the predefined **DeliveryNote** template.
6. Save the project.

K3. Activate the Due Date Field

To activate the **Due Date Number** field, complete the following steps:

1. In WebCenter Forms Recognition Designer, open the **<project>. sdp** file,
2. Navigate to **Definition Mode**.
3. To display the fields, click **Invoices**.
4. Navigate to the **DueDate** field, and display the field properties.
5. From the **Available Templates** list, select the predefined **DueDate** template.
6. Save the project.

K4. Activate the Delivery Date Field

To activate the **Delivery Date Number** field, complete the following steps:

1. In WebCenter Forms Recognition Designer, open the **<project>. sdp** file.
2. Navigate to **Definition Mode**.
3. To display the fields, click **Invoices**.
4. Navigate to the **DeliveryDate** field, and display the field properties.
5. From the **Available Templates** list, select the predefined **DeliveryDate** template.
6. Save the project.

Appendix L Unit of Measure Conversions

Units of measure appear at line item level on an invoice. They describe the magnitude of the physical quantity to which the line item quantity relates. Common units of measure found on invoices include *each*, *kilograms*, *lbs*, *boxes*, *cases*, and for service invoices that are billing time expended, *hours* and *days*.

AP Packaged Project looks for a unit of measure when extracting the invoice line items.A designated column is included in the line items table for that reason. This is not a mandatory entry.

When exporting the line items, the unit of measure is passed downstream because a paired line item is always the one that was lifted from the purchase order line. This is typically mandated by the downstream ERP system. Hence, if the invoice and purchase order units differ, the system needs to adjust the quantity, so that it is expressed in the purchase order unit of measure to prevent the wrong quantity being booked.

L1. Configure the Measurement Conversion Adjustments

When exporting line items, the unit of measure passed for a paired line item is always the one that is lifted from the purchase order line. This is typically mandated by the downstream ERP system. Therefore, if the invoice and the purchase order units differ, the system needs to adjust the quantity so that it is expressed in the purchase order unit of measure to prevent the wrong quantity being booked.

To configure measurement conversion adjustments, complete the following steps:

1. Open the **BRWLPR** table.
2. Set the **UOMCheck** column to `TRUE`.
3. For a system with a purchase order line that has an order unit of measure as well as a price order unit of measure, set the **PUOMCheck** column to `TRUE`.
4. Save the changes.

L2. Configure the Percentage Tolerance for Measurements

In the majority of cases, the system is able to perform quantity conversion because the units of measure are universal constants, such as converting kilograms to tonnes, or because the relationship is clear from a comparison of the invoice and purchase order quantities and/or pricing data. You can configure the percentage tolerance by which the invoice unit price is allowed to be less or greater than the purchase order unit price in order to infer the same order price unit of measure. If the system is unable to establish the conversion ratio, or if it is unable to confirm whether the quantity actually requires converting, the line is not paired. This is most likely in instances where the invoice and purchase order unit of measure differences are custom to a particular material and/or there are significant differences in the pricing.

To configure the percentage tolerance, complete the following steps:

1. Open the **BRWLPR** database table.
2. In the **PUOMTCheck** column, accept the default of `10` percent, or enter another percentage value.
3. Save the changes.

Percentage Tolerance for Measurements Example

A construction company orders 100 bags of concrete, expecting to pay $69.90 US dollars per bag. The company material number for concrete is 1234A. The purchase order would appear as follows:

Material No	Description	Quantity	Unit of Measure	Unit Price	Total
1234A	Concrete	100	BAG	$69.90	$6,990.00

The vendor then issues an invoice for a partial shipment of the concrete, but the vendor is invoicing in LBS.

, In the above example, the invoice quantity is converted from LBS to bags, and therefore, the price differences does not occur with 100 percent certainty. For 100% certainty, it should be known how many LBS are in each bag of that particular type of concrete. As such, the system would fail the line pairing for this example under regular circumstances.

To mitigate this, there is a unit of measure conversion table, which if populated correctly, allows the example above to succeed.

L3. Configure the AP Packaged Project Unit of Measure Conversion Table

The AP Packaged Project Unit of Measure Conversion table, `OFRUOMConversions`, contains the conversion ratio between the base unit of measure and a custom unit of measure for a given material.

Note If the database connection cannot be established, or incorrect configuration/table entries are present, then the system does not fail document export, but the line item is not paired. If line pairing logging is activated, then any error and tracking messages are recorded in the AP Packaged Project log file.

The structure of the OFRUOMConversions table is in the following table:

ColumnName	Type	Description
Material	String	Client material number
BaseUOM	String	Material base unit of measure in the ERP system which accounts for all stock of that material in a common unit. External unit of measure.
Numerator	Integer	Conversion ratio numerator
Denominator	Integer	Conversion ratio denominator
UOM	String	External unit of measure

During the conversion, the system performs a lookup on the table using the external unit of measure read from the invoice (in this case, LBS) and the material number read from the purchase order (in this case, 1234A).

If a record is found, the system checks whether the order unit of measure on the purchase order (in this case, BAG) is equal to the base unit of measure for the material. If it is, the invoice quantity is converted as per the following calculation:

```
 Converted Quantity = Invoice Quantity * (Numerator / Denominator) = 30 * ( 1 / 15 ) = 2
```

Therefore, 30 pounds is equal to 2 BAGS, so 2 is the corresponding quantity expressed in the purchase order unit of measure.

Line pairing now succeeds for the invoice line. The system passes a quantity of 2, a unit of measure of BAG, and a total line item value of $141.60 to the downstream ERP system.

To configure the AP Packaged Project Unit of Measure Conversion table, complete the following steps:

1. Open the **BRWLPR** table.
2. Set the **ConvertQuantityFrom** column to `TRUE`.
3. Save the change.

L4. BRWUOM Table

The database details are configured per profile ID in the BRWUOM table. The following table contains the default configuration settings, but the database connection string and table and column names can be changed as needed.

Column	Value
ProfileID	0
SQLConnectionGroup	1
DBTableName	OFRUOMConversions
DBMaterialNo	MATERIAL
DBBaseUOM	BASEUOM
DBNumerator	NUMERATOR
DBDenominator	DENOMINATOR
DBExternalUOM	UOM

L5. Unit of Measure Triangulation

A Unit of Measure Triangulation occurs when the order unit of measure for the purchase order line item is not the same as the base unit of measure for the material. Hence, three different units of measure are in play:

- The invoice line item unit of measure.
- The purchase order line item unit of measure.
- The base unit of measure for the material.

For example, the construction company in example 2 decides to order more concrete, though this time they wish to order 10 pallets. The purchase order would be raised as follows:

Material No	Description	Quantity	Unit of Measure	Unit Price	Total
1234A	Concrete	10	PAL	$1,398.00	$1,3980.00

The vendor continues to bill in LBS, and submits the invoice as follows:

Description	Quantity	Unit of Measure	Unit Price	Total
Concrete	900	LBS	$4.72	$4,248.00

Within the construction company's ERP system, the base unit of measure for the material remains as BAG.

Dealing with this situation requires an additional entry in the unit of measure conversion table to represent the conversion ratio between the material base unit of measure (in this case, BAG) and the purchase order unit of measure (in this case, PAL).

Each single pallet holds 20 bags; hence the additional entry must be as follows:

Material	BaseUOM	Numerator	Denominator	UOM
Concrete	900	LBS	$4.72	$4,248.00
1234A	BAG	1	15	LBS
1234A	BAG	20	1	PAL

When converting the invoice quantity, the system first converts it to the quantity in the base unit of measure, then converts it again into the same quantity, but expresses it in the purchase order unit of measure.

The calculations are as follows.

When converting to base unit of measure, the following equation is used:

```
 Base UOM Quantity = Invoice Quantity * (Numerator / Denominator) = 900 * (1 /
15) = 60
```

When converting to the purchase order unit of measure, the following equation is used:

```
 Converted Quantity = Base UOM Quantity * (Denominator / Numerator) = 60 * (1 /
20) = 3
```

Line pairing now succeeds for the invoice line item, and a quantity of 3, a unit of measure of PAL, and a total of $4248.00 is passed to the downstream ERP system.

Under no circumstances must an entry in the table be made as follows:

Material	BaseUOM	Numerator	Denominator	UOM
1234A	PAL	1	300	LBS

Whereas this may be correct mathematically, it is INCORRECT from a table population point of view. It may lead to incorrect quantity conversions as the base unit of measure for the material is not pallets. The developer must always check with the client so that the base unit of measure for the material can be established correctly. The content of that column must be consistent for all entries in the table relating to a given material.

L6. Configure the Unit of Measure Aliases

You can configure groups per profile ID to convert the alias as it appears on the invoice to your standard ERP system ISO-code. To configure the unit of measure aliases, complete the following steps:

1. Open the **BRWUOMType** table.
2. In the **Alias** column, enter aliases that represent the ISO code for a particular measurement. For example, for LBS, enter `LB`, for TON, enter `Tonne`, and so on.
3. Save the changes.

Appendix M Purchase Order Number Validations

AP Packaged Project is able to extract and validate purchase order numbers that relate to data held in ERP system database tables.

Different ERP systems have different ways of organizing data in their underlying databases, which necessitates an alternative approach to data validation depending on what type of ERP system is being used.

As standard, the following types of purchase order numbers are supported for database validation:

- Standard purchase order numbers (as used by Oracle e-Business Suite)
- JD Edwards purchase order numbers
- Peoplesoft purchase order numbers
- ERP systems where the database record is keyed from the purchase order number and company code

Only one purchase order number validation approach can be used per AP Packaged Project project file.

Purchase order number validations are configured per profile ID in the BRWPON table. Purchase order line level configuration that is to be used as part of line pairing is configured per profile ID in the BRWLPR table. If line pairing is to be used, both the header level validation and the line level validation must be configured in tables BRWPON and BRWLPR.

M1. Work with Standard Purchase Order Numbers

Standard purchase orders are those that can be identified uniquely in the downstream ERP system using the purchase order number alone. In other words, the ERP system purchase order header table has a single key field called Purchase Order Number.

An example purchase order header table structure (with sample data) reflecting this arrangement would be as follows:

PONUMBER	VENDORID	COMPANYCODE	POTYPE	CURRENCY
1928370	100010	GB01	GD	GBP

When a purchase order number is extracted in the purchase order number field, the system can be configured to validate the order against a database, and also to bring back other items of data, such as the vendor ID, the company code, the purchase order type and the currency.

M2. Configure Purchase Order Number Validations

To configure purchase order number validation, complete the following steps:

1. Open the **BRWPON** table.
2. Set the **ValidateFromDB** column to `TRUE`.
3. To configure the database connection, in the **SQLConnectionGroup** column, enter the SQL connection group number.
4. To complete the configuration, you must specify the name of the PO header database table supplied by the client, along with two mandatory column mappings, the PO number and the vendor ID. You can add additional columns, too.

The following sample configuration reflects the purchase order header table structure above:

Column in the BRWPON table	Value
DBTableName	PO_HEADER
DBPO	PONUMBER
DBVendorID	VENDORID
DBSiteID	
DBCurrency	CURRENCY
DBCompanyCode	COMPANYCODE
DBStatus	
DBDocType	POTYPE
DBBusinessUnit	PO_HEADER

5. To set the document company code as default to the one on the purchase order, set the **SetCompanyCodeFromPO** column to `TRUE`.
6. Save your changes.

Result: The system now validates any extracted purchase order against the content of this table. If the purchase order is not present, or if a connection to the table cannot be established, or if the column mapping is incorrect, the system marks the purchase order number field as invalid, and the document is sent to Verifier.

M3. Validate a Purchase Order Vendor against a Document

To validate a purchase order vendor against a document, complete the following steps. This must be configured in an environment where the invoice vendor must always be the same as the purchase order vendor.

1. Open the **BRWVND** table.
2. Set the **IgnorePOVendor** column to `FALSE`.
3. Set the **UseASSAIfPOVendorInvalid** column to `FALSE`.
4. If the invoice vendor is different than the purchase order vendor, set the **UseASSAIfPOVendorInvalid** column to `TRUE`.
5. Save the changes.

M4. Configure the Invoice Currency to Match the Purchase Order

In Verifier, if the PO vendor differs from the invoice vendor for both of the previous configurations, the `PO VENDOR <> INVOICE VENDOR` invalid reason must be configured. To default the invoice currency as that of the purchase order if no other value has been extracted, complete the following steps:

1. Open the **BRWCurrency** table.
2. Set the **DefaultPOCurrency** column to `TRUE`.
3. To configure the system to flip the AP Packaged Project PO type based on whether SV is found in the **POTYPE** column of the Purchase Order Header table, set the **ServicePOTypes** column to `SV`.

4. If your service purchase-order numbers always begin with 42 or 43, and if the purchase order number is successfully validated against the database, then to force the purchase order type to change to `SERVICE`, in the **ServicePOPrefixes** column, enter `42,43`.
5. Save the changes.

M5. Configure Padding for Extracted PO Numbers

You can configure padding an extracted purchase order number with leading zeros so that it matched the data held in the purchase order header table. To configure padding for extracted PO numbers, complete the following steps:

1. Open the **BRWFLD** table.
2. Set the **FieldName** column to `PONumber`.
3. Set the **MaxLength** column to `10`.
4. Set the **RightJustify** column to `TRUE`.
5. Set the **PadChar** column to `0`.
6. Save the changes.

Result The purchase order validation can also be extended to include data held at the line item level.

M6. Configure Alternative PO Number Validation

Alternative PO numbers refer to those where a unique purchase order is identified in the purchase order header table by a combination of the PO number itself and the company code. To configure the system to use alternative PO number validation, complete the following steps:

1. Open the **BRWPON** table.
2. Set the **POKeyIncludesCompanyCode** to `TRUE`.
3. In the **DBTableName** column, enter `PO_HEADER`.
4. In the **DBPO** column, enter `PONUMBER`.
5. In the **DBVendorID** column, enter `VENDOR_ID`.
6. In the **DBCompanyCode** column, enter `COMPANYCODE`.
7. Save the changes.

M7. Configure PO Line Item Validation

If the information to determine is held at line item level, the system uses purchase order line item information to assess whether the purchase order type is `MATERIAL` or `SERVICE`. The system also assesses whether the invoice is a MIRA, such as where there is a 1-1 relationship between net invoice value and total purchase order value and to perform line pairing. To configure PO line item validation, complete the following steps:

Note: The structure of the purchase order line item database table varies depending on the ERP system.

---

1. Open the **BRWLPR** table.
2. Set the **GETPOLinesFromDB** column to `TRUE`.
3. Set the **SQLConnectionGroup** to the database connection string.
4. Set the **DBTableName** column to `PO_LINES`.
5. Map the **DBPO** and **DBLINE** columns.

6. For alternative purchase orders, map the **DBPO, DBLINE**, and **DBCOMPANYCODE** columns to the key fields.
7. Map the **Material Number** column to the column in the PO line item table that represents the unique material number for that item.
8. If available, map the **Price Unit** column. Otherwise, it is assumed that it is always numeric **1**.
9. If the value in the **Item Category** column lets the system determine whether an item is a material or a service, which in turn can influence the setting of the AP Packaged Project purchase order type field, map the **Item Category** column.
10. Save the changes.
11. Open the **BRWPON** table.
12. If `SRV` or `DSRV` denotes that the purchase order line item is a service, then set the **ServicePOTypes** column to `SERVICE`.
13. To have the PO type determine the order unit of measure, in the **ServicePOUOMs** column, enter a comma-separated list of the units of measure in the order you want them to appear.
14. Save the changes.

Additional LPR Table Mappings

The following table contains sections that may be mapped, and it describes the circumstances under which mapping those fields are beneficial:

Column	Description
DBMATERIALGROUP	This column represents the material group to which the purchase order line item material belongs. If available, it can be mapped simply to pass this data to a downstream system for a paired line item. It is also used in the automatic tax determination procedure if the selection of the correct tax code is driven by the material group of the item.
DBTAXCODE DBTAXJURCODE	These two columns represent the tax information that was set when the purchase order was originally raised. For countries that do not use tax jurisdictions (i.e. where tax rates are set by government at the national level such as within the European Union), the tax code tells the ERP system how to handle tax for the line item in terms of the percentage rate of tax to be charged. It also describes whether an item is a service, tax-exempt, or zero-rated because it is, for example, an EU cross-border transaction. The tax jurisdiction code is not used under this circumstance. For countries that do use tax jurisdictions (i.e. where tax rates are set at a local level such as in the US, Canada, India, and Brazil), the tax code tells the ERP system whether the item is fundamentally subject to tax or not. The accompanying tax jurisdiction code, which is the identification number of a specific tax office, represents the actual percentage rate information. These columns need only be mapped if the automatic tax determination feature is being used.

<table>
<tr><th>Column</th><th>Description</th></tr>
<tr><td>DBPUOM</td><td>This column represents the order price unit of measure, which is the unit of measure that is associated with the unit price for the purchase order line item.

It may differ from the regular unit of measure (column DBUOM) which is associated with the order quantity on the purchase order line.

Example:

The client may raise a purchase order for 1000 each (EA) of product A, but the unit price of $10.00 is set per case (CASE).

If there are 10 EA per CASE, then the PO line would appear as follows:

<table>
<tr><th>Description</th><th>Quantity</th><th>UOM</th><th>UnitPrice</th><th>Total</th></tr>
<tr><td>Product A</td><td>1000</td><td>EA</td><td>$10.00</td><td>CASE</td></tr>
</table>

The PUOM column MUST always be mapped in implementations where instances such as the above are possible within the client's ERP system. Not doing so may cause incorrect invoice quantities to be passed downstream for paired line items.</td></tr>
</table>

DBTOTALQUANTITYDELIVERED
DBTOTALVALUEDELIVERED
DBTOTALQUANTITYINVOICED
DBTOTALVALUEINVOICED

These four columns provide AP Packaged Project with purchase order line item history data, so that the system knows exactly what has been invoiced and goods-receipted to date, both in terms of quantities and overall values.

If this information is available in the purchase order line item table, it can improve the success rate of the line pairing operation. This may involve creating a view based on the standard line item table and a separate history table.

Example 1:

An invoice is received for product A with a quantity of 2, a unit price of $10.00, and an overall line total of $20.00.

The corresponding purchase order has two line items, both for product A with the same quantities and pricing:

Line	Description	Order Qty	Unit Price	Total
1	Product A	2	$10.00	$20.00
2	Product A	2	$10.00	$20.00

During line pairing, the system is not able to make a decision between these identical line items, so line pairing fails (assuming that column EnableIntegrityCheck is set to TRUE in the BRWLPR table, else the invoice line is booked to PO line 2).

However, line item 1 may have an open goods receipt against it, whereas there is no goods receipt against line 2. Hence, line 1 is the preferable line to book against.

By mapping the additional history columns, it now becomes clear to the system which line to book against as, with this extra information, the lines are no longer identical.

Line	Description	Order Qty	Unit Price	Total	TQD	TVD	TQI	TVI
1	Product A	2	$10	$20	2	$20	0	0
2	Product A	2	$10	$20	0	0	0	0

In this case, the system books the invoice to line 1.

Example 2:

A second invoice comes in with identical line item detail as the invoice in example 1. Since the first invoice has already been booked on the ERP system, the status of the history would have changed, as line 1 is not only goods receipted, but fully invoiced as well.

Hence, the purchase order line detail appears as follows:

Line	Description	Order Qty	Unit Price	Total	TQD	TVD	TQI	TVI
1	Product A	2	$10	$20	2	$20	2	$20
2	Product A	2	$10	$20	0	0	0	0

In this instance, the system is able to see that the first purchase order line is already fully invoiced.Hence the invoice line is paired to available PO line item 2.

In example 1 above, the MIRA condition would hold as the net value of the invoice would be equal to the total value of goods receipts not yet invoiced. Hence, by mapping these extra columns, not only would line pairing succeed, but also the user would not be required to validate line item extraction needlessly in Verifier (if the system is set to skip line item validation for MIRA invoices). This is controlled in the BRWTAB table.

Column	Description
	Example 2 would also meet the MIRA condition if there was a $20 goods receipt value against line 2, but no invoices booked against it. If the IgnoreCompletedPOLines column is set to TRUE in the BRWLPR table, then any fully booked purchase order lines (i.e. where the total quantity invoiced is greater than or equal to the quantity ordered) are not considered for line pairing under any circumstances. The first PO line in example 2 illustrates this scenario.
DBPLANT	The plant is a code set in the client ERP system that represents the physical location where the purchase order line goods are to be delivered, or where a service is to be performed. For example, it could represent a warehouse or an office building. The plant column can be mapped simply to pass that data to a downstream ERP system for each paired line item, but the system requires the information if either: 1. The invoice includes a miscellaneous charge, and miscellaneous charges are set to be outputted as general ledger account entries, where the corresponding coding string in the BRWMisc table is driven by the plant on the purchase order; 2. The automatic tax code determination feature is being used and the country of the invoice company code cannot be used to determine where the goods were delivered. The specific address details related to each plant code (i.e. the country and the state) can be read either from the database table OFRPlant. Settings in the BRWTAXCONFIG table control the appropriate data source.
DBCHARGECODE DBCHARGECODEID	These two columns are made available for implementations involving Oracle e-Business Suite where the charge code and the charge code ID information needs to be brought into AP Packaged Project and then passed back for each line item where line pairing has succeeded.

M8. Configure Database Validations Using a Stored Procedure

By default, the system accesses purchase order header and line item tables using an SQL select call. However, it is possible to use a custom stored procedure instead, which the client may wish to do for data security purposes. The stored procedure must be written to return a record set in the same way that would have been achieved had a regular SQL select statement been executed. The system can be configured to use a custom stored procedure for both purchase order header and line item validations.

To configure database validations using a stored procedure, complete the following step:

1. Open the **BRWPON** table.
2. Set the **UseStoredProcedure** column to `TRUE`.
3. In the **StoredProcedureName** column, enter the custom stored procedure name.
4. In the **SQLConnectionGroup**, enter the SQL connection group.
5. Open the **BRWLPR** table.
6. Set the **UseStoredProcedure** column to `TRUE`.
7. In the **StoredProcedureName** column, enter the custom stored procedure name.
8. In the **SQLConnectionGroup**, enter the SQL connection group.
9. Save the changes.
10. Open the **BRWSPC** table.
11. Set the formal interface parameter name in the stored procedure to represent an AP Packaged Project field. Each of the following parameters is assigned a type, and if that type is `VARCHAR`, a length is assigned as well:

- BOOLEAN
- INT
- DATE
- DOUBLE
- VARCHAR

Note: The parameter value, if set to represent afield, is case-sensitive and must be the technical name of the field, such as `PONumber`, `CompanyCode`, `DocumentType`, `AmountTotal`, and so on. If the value entered is not the technical name of a field, the system assumes it is a hard-coded value. Input parameters passed into the stored procedure, such as the values passed from AP Packaged Project, must have a direction of I. Output parameters coming from the stored procedure have a direction of O. If the direction is missing or invalid, the parameter is considered as an output parameter.

12. Save the changes.

Once the parameters have been defined, assign them to the corresponding purchase order header stored procedure. Refer to the *Assign parameters to a purchase order header stored procedure* section for more information.

M9. Assign Parameters to a Purchase Order Header Stored Procedure

To assign parameters to a purchase order header stored procedure, complete the following steps:

1. Open the **BRWPON** table.
2. Set the **StoredProcedureParameters** column to the assigned purchase order line item such as `1,2,3`.
3. Save the changes.
4. Open the **BRWLPR** database table.
5. Set the **StoredProcedureParameters** column to the stored procedure parameters you entered in the **StoredProcedureParameters** column in step 2.

M10. UserExitReadPODetails

UserExitReadPODetails is available so that a developer may code an alternative approach to retrieve purchase order details (for example, using web services), rather than utilizing the standard functionality. Once retrieved, the details gathered must be populated into the POHeader structure and POLineItems array that form part of the user exit interface.

The user exit interface is as follows:

Import Parameter	Type	Description
pWorkdoc	Workdoc object	This is the current workdoc object.
POHeader	POHeaderStructure	This is the purchase order header structure that must be populated within the user exit.
POLineItems	POLineItemsStructure array	This is a 1-based array for the purchase order line items and must be populated if required within the user exit.
POKey	POKey	This contains the details of the purchase order that is to be retrieved.

Import Parameter	Type	Description
Client	ClientData	This is the client data object containing details of the document client.
Address	VendorAddress	This is the address structure containing details for the current invoice vendor.
strPOReadError	String	This must be set to an appropriate error message must a technical problem arise during the retrieval of the purchase order details. If a message is set, it stops a document in Verifier and also fails a document export.
blReadPOLines	Boolean	The system sets this flag to TRUE if the retrieval of line item information is required for the profile of the current document. This field is read only.
blDuplicatePO	Boolean	This flag must be set to TRUE if multiple records are returned during the retrieval of the purchase order header information.
blExport	Boolean	The system sets this flag to TRUE if the user exit is being called during document export. During validation on server side or in Verifier, it is set to FALSE. This field is read only.
blPONotFound	Boolean	This flag must be set to TRUE if the purchase order header does not exist in the external data source. It is optional to set it to TRUE if the purchase order header lookup is successful, but the corresponding line items do not exist in the data source.

To activate the user exit, the parameter **ReadPOHeaderViaUserExit** is set to `TRUE` in the **BRWPON** table for the required profile ID. If the user exit is also to be used to retrieve purchase order line item details, the parameter **ReadPOLinesViaUserExit** is also set to `TRUE` in the **BRWLPR** table for the profile ID.

It is not possible to mix and match a custom read of the purchase order header and a standard read of the line item data, or vice versa. If the user exit is used to read the purchase order header, it must also be used to read the line items. The user exit import parameter **blReadPOLines** is set to `TRUE` if the profile has been configured to read line items, thus notifying the developer that a read of the line item information is needed. If line items are not needed (i.e. **blReadPOLines** is set to `FALSE`), but are retrieved anyway, they are ignored by the system.

The user exit is called when a purchase order number is validated both on server side and also in Verifier. It is also called during document export in order to retrieve the current purchase order details for line pairing. The import parameter **blExport** provides a means for the developer to distinguish between the two. During validation, it is set to `FALSE`, and during document export, it is set to `TRUE`.

The key of the purchase order to be retrieved is passed into the function using the **POKey** import parameter. The POKey has the following structure:

Structure element	Type	Description
PONUMBER	String	This contains the formatted purchase order number to be queried.

Structure element	Type	Description
COMPANYCODE	String	This contains the invoice company code. This value is only populated if the company code forms part of the key to identify a unique purchase order.
POEXTENSION	String	This contains the JDE PO document type or Peoplesoft business unit, depending upon whether the profile has been configured for JD Edwards or Peoplesoft PO types respectively. If neither is used, the value is blank.

Within the user exit, the developer must use the details provided by the POKey structure to query the external system. If the profile ID is also required, then this is available in the **ProfileID** component of the Client structure, which is also passed into the user exit.

If the client profile has been configured to use a purchase order data partition, the partition ID for the profile is also passed into the user exit via the **POPartition** component of the Client structure. This must be done in tandem with checking that the profile has been set up to use partitions for the purchase order data.

```
For example DicVal("UsePOPartition", "PON") = "YES"
```

If a technical error occurs when performing the lookup to the purchase order data source, **strPOReadError** must be set to an appropriate error message. If this value is set to anything but blank, it has the effect of stopping the purchase order in Verifier with the error displayed in the field error text. During export, if **strPOReadError** is set to an error message, the export fails and the document is sent to `status 750`. The error is subsequently written into the export instance log file and also displayed to a user in Verifier against the invoice number field.

If the lookup to the external data source is successful, but the PO cannot be found, the **blPONotFound** import parameter must be set to `TRUE`. This stops the document in Verifier with an appropriate error message, but, if this occurs at time of document export, the export as a whole does not fail. It is not recommended to set an error in **strPOReadError** if the purchase order does not exist in the data source. If the profile ID has been configured to look for multiple purchase orders, this could lead to documents getting stuck needlessly at time of export. **strPOReadError** must be reserved exclusively for the handling of technical configuration or connectivity issues.

If the purchase order header can be read, but the purchase order line items cannot be retrieved, then the flag **blPONotFound** may be set to `TRUE` if it is appropriate that the document must remain in Verifier until the line items become available. If the flag remains set as `FALSE`, then the purchase order number is accepted during validation, but it is not possible to carry out line pairing during document export. This, however, does not let the document export as a whole to fail.

If multiple records are returned for the purchase order header lookup indicating that duplicate purchase orders exist in the external data source, the parameter **blDuplicatePO** must be set to `TRUE`. Doing so activates the standard purchase order duplicate handling functionality if **SkipDuplicatePOCheck** is set to `FALSE` in the BRWPON table.

Once the data is retrieved successfully, the components of the POHeader structure and the POLineItems array can be populated. The following table describes the structure of the purchase order header (POHeader), explains which of the components are mandatory and which are optional, and also provides hints for populating each component:

Structure element	Mandatory?	Type	Description
DOCTYPE	No	String	This is the purchase order document type. It must be populated if configuration exists to set the POType field to SERVICE based upon a purchase order document type.
COMPANYCODE	No	String	This is the purchase order company code. This needs to be populated if the profile configuration is set to default a company code based upon the purchase order number lookup.
VENDORID	Yes	String	This is the PO order-from vendor and is a mandatory item to populate. If the field is not populated, then the system raises a configuration error. Hence, the purchase order number field is set to invalid and the document is sent to Verifier. If this issue occurs during export, then the export fails and the document is sent to a 750 state.
SITEID	Yes if the document profile uses a site ID	String	This is the site ID for the purchase order vendor. It is mandatory to populate this component if the profile of the document requires a site ID. If the field is not populated, and a site ID is needed, the system raises a configuration error. Hence, the purchase order number field is set to invalid and the document is sent to Verifier. If this issue occurs during export, then the export fails and the document is sent to a 750 state.
CURR	No	String	This is the purchase order currency. This component must be populated if the configuration profile of the document is set to default a currency from the purchase order. The currency must also be populated for line pairing to be carried out.
RELEASEFLAG	No	String	This is the purchase order release flag. If purchase orders in the downstream ERP system are subject to a release strategy meaning that the invoice cannot be processed until the purchase order is released, and the purchase order has still not been released, then this field must be populated with a value. Under all other circumstances, it must be left blank. Populating this field with a value has the effect of notifying the Verifier user that the PO has not been released. During export, line pairing is not carried out.
DIFFINV	No	String	This is the remit-to vendor ID. If an order-from and a remit-to vendor are present on the purchase order, the remit-to vendor takes priority in terms of the vendor that is displayed to the Verifier user and exported.
STATUS	No	String	This is the purchase order document status.

Structure element	Mandatory?	Type	Description
EXRATE	No	Double	This is the exchange rate between the purchase order currency and the local currency of the company code in which the purchase order was created. It is expressed as the factor by which the purchase order totals must be multipled to convert from the PO currency to the company code currency. For example, if the purchase order document currency is CNY and the company code currency is GBP, then, assuming that 1 GBP = 10 CNY, the exchange rate must be set to 0.1. i.e. 10 CNY (total in PO currency) * 0.1 (exchange rate) = 1 GBP (total in company code currency) It is not mandatory to populate this field, but if the purchase order currency differs from the company code currency and the invoice is presented in the company code currency, then line pairing cannot be performed.

The following table describes the structure of the purchase order line items array (POLineItems), explains which of the components are mandatory and which are optional, and also provides hints for populating each component.

Although all but one of the fields are optional, it is recommended to populate as many relevant fields as possible in order to maximize the success of line pairing. The relevant fields that fall into this category are:

- MATERIALNO
- DESCRIPTION
- POQUANTITY
- UOM
- UNITPRICE
- PUOM
- PRICEUNIT
- TOTAL
- TOTALQUANTITYDELIVERED
- TOTALVALUEDELIVERED
- TOTALQUANTITYINVOICED
- TOTALVALUEINVOICED

Structure element	Mandatory?	Type	Description
LINENO	Yes	String	This is the purchase order line item number. This is a mandatory component used to identify a purchase order line item uniquely within the purchase order. The system raises a configuration error if it is left empty. Hence, the purchase order number field is set to invalid and the document sent to Verifier.If this issue occurs during export, then the export fails and the document is sent to a 750 state.
MATERIALNO	No	String	This is the purchase order line item material number.
MATERIALGROUP	No	String	This is the purchase order line item material group.
DESCRIPTION	No	String	This is the purchase order line item description.
POQUANTITY	No	Double	This is the line item order quantity.
UOM	No	String	This is the purchase order line item quantity unit of measure.

Structure element	Mandatory?	Type	Description
UNITPRICE	No	Double	This is the line item unit price.
PUOM	No	String	This is the purchase order line item price unit of measure.
PRICEUNIT	No	String	This is the purchase order line item price unit. If this value is not populated, the default value is 1.
TOTAL	No	Double	This is the line item order total.
TAXCODE	No	String	This is the purchase order line item tax code. It is recommended to populate this field if the automatic tax determination feature is being used.
TAXJURCODE	No	String	This is the purchase order line item tax jurisdiction code, which represents the ID of a tax office relevant to the location where the purchase order item is ultimately to be consumed or used. Tax jurisdiction codes are typically used in the US or any country where the rates of sales/use tax are set in law at the provincial level, as opposed to a fixed rate of sales tax set at the national level (such as VAT within the EU). If relevant for the downstream ERP system, It is recommended to populate this field.
TOTALQUANTITYDELIVERED	No	Double	This is the total quantity already delivered for the purchase order line item.
TOTALVALUEDELIVERED	No	Double	This is the total value of the goods already delivered for the purchase order line item.
TOTALQUANTITYINVOICED	No	Double	This is the total quantity that has already been invoiced for the purchase order line item.
TOTALVALUEINVOICED	No	Double	This is the total value of the goods already invoiced for the purchase order line item.
ITEMCATEGORY	No	String	This is the purchase order line item category. This must be populated if the profile of the document has been set up to determine whether the POType is SERVICE based upon the purchase order line item categories.
PLANT	No	String	This is the ID of the purchase order line item plant (i.e. where the goods are to be delivered). This must be populated if the automatic tax determination feature is activated, as it specifies the ship-to state/country for the purchase order line item.
CHARGECODE	No	String	This is the purchase order line item charge code.
CHARGECODEID	No	String	This is the purchase order line item charge code ID.
ERS	No	Boolean	This flag must be set to TRUE if the purchase order line item is marked for ERS (Evaluated Receipt Settlement). If the parameter StopERSPO is set to TRUE in the BRWPON table for the document profile, this has the effect of rejecting the invoice to a Verifier user. This is becaue for the ERS type purchase orders, no paper invoice is expected as it must be processed electronically as part of an EDI-type transaction.
MULTIPLEACCOUNTASSIGNMENT	No	String	This indicates if the purchase order line item has been set up with a multiple account assignment. It must be left blank if no multiple account assignment exists.

Structure element	Mandatory?	Type	Description
ACCOUNTASSIGNMENT CATEGORY	No	String	This is the purchase order line item account assignment category. It must be populated if the automatic tax determination feature is being used, and the configuration exists to default a tax code based upon the purchase order line item account assignment category.

It must be populated if the automatic tax determination feature is being used and the configuration exists to default a tax code based upon the purchase order line item account assignment category.

A suggested overview of the process flow in the script is as follows:

1. Connect to the purchase order header data source and query using the key provided in **POKey**.
2. If any connection or configuration issues arise, set **strPOReadError** to an appropriate error message and exit.
3. If the purchase order does not exist in the data source, set **blPONotFound** to `TRUE` and exit.
4. If multiple purchase order headers are retrieved, set **blDuplicatePO** to `TRUE` and exit.
5. Populate the **POHeader** object.
6. If **blReadPOLines** is set to `FALSE`, exit at this point.
7. Connect to the purchase order line item data source and query using the key provided in **POKey**.
8. If any connection or configuration issues arise, set **strPOReadError** to an appropriate error message and exit.
9. If no line item data exists for the purchase order, the **blPONotFound** flag may be set to `TRUE`, but this is optional depending upon project requirements.
10. Loop around the line items returned and copy the data into the **POLineItems** array.
11. Exit the subroutine.

If a custom validation is also required within the user exit where the desired behaviour is such that the document must stop in Verifier with an error message notifying the user of a problem, but the user may still accept the purchase order, then a new step may be inserted between steps 3 & 4 where the error message is passed via **strPOReadError** only if **blExport** is `FALSE` and **fnIsVerifier** is also `FALSE`.

```
If Not blExport And Not fnIsVerifier Then
  ' Report error to user from custom validation
strPOReadError = "Custom validation failed – please check the purchase order."
End If
```

M11. Use Purchase Order Partitions

Purchase order partitions provide the capability to contain all purchase order data within a single set of database tables. For example, if an AP Project project involves 100 clients, each of whom provide their own set of purchase order data, it was earlier necessary to configure the system to point to up to 200 tables – 100 purchase order header tables, and 100 purchase order line item tables.

It is now possible to include the purchase order data all within a single set of tables and then use purchase order partitions to ensure that the data for one client is not confused with data belonging to another client, thus reducing the maintenance burden for a system administrator.

The steps to create and assign a partition are as follows.

Register the PO Data Partition
To register a new PO data partition, complete the following steps:

1. In SQL Server, open the **BRWPONPartition** table.
2. Populate a row with a unique partition ID. This must be an integer.
3. Add a description. Adding a description of the partition is optional, but its naming must be indicative of what it represents.
4. Save the changes.

Assign the PO Data Partition ID to the Client
To assign the PO data partition ID to the client, complete the following steps:

1. In SQL Server, open the **BRWClient** table.
2. In the corresponding **POPartition** column, enter the ID of the newly registered partition for the appropriate client. The system does not permit an entry which does not exist in the BRWPONPartition table.
3. Save the changes.

Activate the PO Data Partition
To activate the PO data partition, complete the following steps:

1. In SQL Server, open the **BRWPON** table.
2. Ensure that the **ValidateFromDB** column is set to `TRUE`.
3. Set the column **UsePOPartition** to `TRUE`.
4. Enter the technical name of the purchase order header database table column which represents the partition ID in the column **DBPartition**.
5. Save the changes.

Map the Partition ID Column for the Purchase Order Line Item Data
If the client profile is also set to read purchase order line item data, the purchase order line item database column, which represents the partition ID, must also be mapped.

The steps to accomplish this are as follows:

1. In SQL Server, open the **BRWLPR** table.
2. Enter the technical name of the purchase order header database table column which represents the partition ID in column **DBPartition**.
3. Save the changes.

If the purchase order header data is set to use a partition, the system expects that the purchase order line item database table also uses a partition.

M12. Work with JD Edwards Purchase Order Numbers

Standard purchase order number validation must be used in instances where the purchase order number is guaranteed to be unique across all company numbers, purchase order types, and suffixes. JD Edwards purchase orders are different from standard purchase orders because the JD Edwards ERP system identifies a purchase order uniquely within the purchase order header table (F4311) using four keys.

The JD Edwards ERP system uses the following unique keys:

- PONUMBER
- COMPANYCODE
- POTYPE
- POSUFFIX

Multiple records in the PO_HEADER table may have the same purchase order number. Because of this, the database queries must be adapted to account for other key fields. As of build 1006, AP Packaged Project caters to the company code and purchase order type columns, but not the purchase order suffix.

When a purchase order number is extracted, the whole value is entered into the purchase order number field, and the purchase order type is either the first two or last two characters of the string, then the system automatically separates the purchase order number from the purchase order type on the server side and in Verifier. The purchase order type is placed into a mandatory PO extension field.

In instances where the purchase order type is always going to be **OP** or another constant determined by the client, the JD Edwards configuration approach is not appropriate. In this case, consider implementing an alternative approach that uses the purchase order number and the company code to perform the lookup into the purchase order header table. As the company code field forms part of the key for reading the purchase order from the database table, it cannot be defaulted from the purchase order. The strategy to derive the company code must either be to default it based on part of the image file name set during the scanning process in the **IMP** section of the system configuration, or to use the associative search engine to derive it based on the bill-to details on the invoice. We always recommend the former approach.

Activate PO Number Validation for JD Edwards
To activate validation of the purchase order number against a JD Edwards table schema, complete the following steps:

1. In the **BRWPON** table, set the **JDEPO** parameter to `TRUE`.
2. Map the following columns:

Column	Value
DBTableName	PO_HEADER
DBPO	PONUMBER
DBVendorID	VENDORID
DBCompanyCode	COMPANYCODE
DBDocType	POTYPE

Note: The system displays an error message if the mapping is insufficient or incorrect.

Configure the JD Edwards PO Number Format
On the purchase order field of the invoice, the system expects the vendor to include the purchase order number and the purchase order type. To configure JD Edwards purchase order number formats, complete the following steps:

1. In the **BRWPONFormats** table, set the following parameters:
   - `ProfileID`
   - `Index`
   - `Format`
   - `IgnoreCharacters`

For Example: If **Format** is set to `10####OP`, the system looks for a six digit number beginning with 10 and has OP at the end.

2. Optional. Configure a second **Format** parameter to `10####0P`. This causes the system autocorrect the PO number format to end in OP.
3. In the **BRWPON** table, add `OP` to the **JDEPOTypes** column. This is a comma-separated list of JD Edwards purchase order types.

M13. Work with Peoplesoft Purchase Order Numbers

The Peoplesoft ERP system uses a double key to identify a purchase order number uniquely within the purchase order header table, PS_PO_HDR. This key consists of the purchase order number and the purchasing business unit.

The Peoplesoft ERP system uses the following unique keys:

- `PO_ID`
- `BUSINESS_UNIT`

Peoplesoft uses a vendor ID/site ID combination (`VENDOR_ID/VNDR_LOC`) to identify a single vendor at a single address uniquely.

Additionally, the purchase order header table does not contain a company code. The general Peoplesoft equivalent of the company code is the payables business unit. The strategy to derive this must either be to default it based on part of the image file name set during the scanning process set in the **IMP** section of the system configuration, or to use the associative search engine to derive it based on the bill-to details on the invoice.

Activate PO Validation for Peoplesoft
To activate validation of the purchase order number against a Peoplesoft table schema, complete the following steps:

1. In the **BRWPON** table, set the **PeopleSoftPO** parameter to `TRUE`.
2. Map the following columns:

Column	Value
DBTableName	PO_HEADER
DBPO	PONUMBER
DBVendorID	VENDORID
DBCompanyCode	COMPANYCODE
DBDocType	POTYPE

Note: The system displays an error message if the mapping is insufficient or incorrect.

Configure Peoplesoft PO Number Format
In the purchase order field of the invoice, the system expects the vendor to include the purchase order number and the purchasing business unit. To configure Peoplesoft purchase order number formats, complete the following steps:

1. In the **BRWPONFormats** table, set the following parameters:
   - `ProfileID`
   - `Index`
   - `Format`
   - `IgnoreCharacters`

Example: If **Format** is set to `10####BWARE`, the system looks for a six digit number beginning with 10 that has BWARE at the end.

2. In the **BRWPON** table, add `BWARE` to the **PeopleSoftBusinessUnits** column. This is a comma-separated list of Peoplesoft purchase order types.

M14. Add Leading Zeros to a PO number for JD Edwards and Peoplesoft

You can configure the system to pad an extracted purchase order number with leading zeros so it matches data held in the purchase order header table. To add leading zeros to a PO number, complete the following steps:

In the **BRWFLD** table, under the PONumber entry for the profile ID, set the following parameters:

Column	Value
FieldName	PONumber
MaxLength	10
RightJustify	TRUE
PadChar	0

Example: If `1002334` is extracted from the invoice, the system formats the purchase order number to `0001002334`.

Note: All other aspects of the validation such as validating the PO vendor against the invoice and setting the currency and AP Packaged Project PO type fields operate in the same way as standard purchase order number validation.

Appendix N Configure Processing Instructions

You can add specific processing instructions for fields on the Dynamic Verifier form. Users access the instructions by clicking **Instructions** next to a field, and then the instructions appear in a dialog box on the form.

N1. Create Instructions for Dynamic Verifier Forms

To create a set of processing instructions for a Dynamic Verifier form, complete the following steps:

1. Open the **BRWINSTR** table.
2. In the **ProfileID** column, add the profile ID. This must be an integer and unique within the table.
3. In the **Instructions** column, enter the instructions that you want to be displayed when a user clicks I**nstruction** next to the field on the **Dynamic Verifier** form.
4. Save the changes.

N2. Assign Instructions to a Client

To assign instructions to a client, complete the following steps:

1. Open the **BRWINSTRClient** table.
2. In the **InstructionsProfileID** column, for the client ID that you are assigning the instructions to, add the ProfileID that you entered while following the steps in the *Create instructions for Dynamic Verifier forms* topic.

Appendix O What are Review States

A second verification step can be turned on or off at either the user or the client level. The purpose of this review step is to allow for additional quality control of either automatic extraction or user entry prior to the export of the document.

All fields are extracted by the system automatically, the document flow is as follows:

Import > OCR > Classification > Extraction > Review > Export

If one or more fields requires user attention, the document flow is as follows:

Import > OCR > Classification > Extraction > Verification > Review > Export

When a document goes to review, it is set to a specific state of `699` by default.

Note: This state must be accessible only by members of the BPO organization who are authorised to review documents.

When reviewers enter the batch using the Verifier application, they can either make changes to any fields if problems are detected.If no problems are found, they can press **Enter** on the first editable field. The document then moves to the regular export.

The before and after values for each field are stored in the reporting database along with the review start and end time, and the ID of the reviewer.

Set a Review State

To set a review state, comlete the following steps:

1. Open the **<project>.ini** file.
2. Navigate to the **GRL** section.
3. Set the **GRL_VL_ReviewState** parameter to the following:

   ```
   GRL_VL_ReviewState=699
   ```

   Note: It is reccommended that you use `699` for the parameter, but you can use any values between `650` to `699` or `701` to `749`.

Activate Document Review for a Client

You can activate document reviews at the client level to send every processed document in a specific client to the review state prior to data export.

To activate document review for a client, complete the following steps:

1. Open the **BRWClient** database table.
2. Set the **RequiresReview** column to `TRUE`.

Activate Document Review for Specific Users

You can also activate document reviews at the individual user level. If you implement review states for users, then all documents processed by that user are sent for further review, regardless of the setting at the client level in the BRWClient table.

This can be relevant for operations who would wish documents processed by less-experienced users to be subject to supervisory review until the user becomes more proficient.

To configure document review for a user, complete the following steps:

1. Open the **BRWUser** table.
2. Set the **RequiresReview** column for the user to `TRUE`.
3. Save the changes.

Appendix P Configuring AP Project Reporting

While WebCenter Forms Recognition does not provide a native reporting tool, the AP Packaged Project can be configured to generate comprehensive processing metrics and other data that can be used to create reports.

The **WFR_AP_Reporting_Create_Oracle.sql** or **WFR_AP_Reporting_Create_Sql_Server.sql** script provided with the AP Packaged Project can be (optionally) executed against an Oracle or SQL Server database to create a set of tables for this purpose. If the AP Project Reporting feature is enabled, these tables will be populated with data to allow customers to:

1. Obtain solution key performance metrics.
2. Monitor documents as they move through the system.
3. Identify solution bottlenecks.
4. Report on productivity at the project and client levels.
5. Report on user productivity.

Note: A sample AP Project Reporting application, created using Oracle Application Express earlier may not be supported with this version. Customers can use third party reporting tool view reports with the generated data.

Configure AP Project Reporting

To configure the connection to reporting database, complete the following steps:

1. Open the **<project>.ini** file.
2. Navigate to the REP section.
3. Set the **ConnectToReportingDB** parameter to the following:

   ```
   ConnectToReportingDB=Yes
   ```

4. Set the **SQLConnectionGroup** parameter to the following:

   ```
   REP_OP_ConnectToReportingDB=Yes
   REP_VL_SQLConnectionGroup=01
   ```

5. WebCenter Forms Recognition begins the reporting trail for each document upon the initial import of the document into the system.If you want reporting to start sooner, such as at scan time, then set the **REP_OP_StartNewRecordForImportedDocument** parameter to the following:

   ```
   REP_OP_StartNewRecordForImportedDocument=No
   ```

6. Navigate to the **SQL** section.
7. Set the **ConnectionString** parameter to the following for each string connection, `SQL_VL_01_ConnectionString, SQL_VL_02_ConnectionString`, and so on.

   ```
   SQL_VL_01_ConnectionString=Provider=SQLOLEDB.1;Password=test;Persist
   Security Info=TRUE;User ID=test;.initial Catalog=PICT;Data Source=W08-
   SERVER\SQLEXPRESS,1254
   ```

   Note: If no SQL connection group is specified, the system always defaults to group `01`. This applies to all SQL connection groups within the system.

8. Navigate to the **IMP** section.
9. Set the **IMP_VL_URN** parameter to the following:

   ```
   IMP_VL_URN=COMPONENT2
   ```

10. Navigate to the **REP** section.
11. Set the **REP_VL_ReportingKey** parameter to the following:

    ```
    REP_VL_ReportingKey=URN
    ```

12. Optional. If you want to use a specific naming convention for any reporting tables created in the databases, set the names of the tables using the following parameters:

    ```
    REP_VL_ReportingDBDocumentTable=OFRDOCUMENT
    REP_VL_ReportingDBFieldTable=OFRFIELDS
    REP_VL_ReportingDBHistoryTable=OFRDOCSTATUS
    REP_VL_ReportingDBImageTable=OFRDOCIMAGE
    ```

13. Optional. Information is not written to a database from any documents processed in the Designer module. However, you can have information written to a database from any document processed in the designer for testing and debugging purposes by setting the **REP_OP_ReportingInDesignerparameter** to the following:

    ```
    REP_OP_ReportingInDesigner=Yes
    ```

    Note: In production environments, this must always be set to **No**.

14. (Optional) If you want to use the Reporting database to include an image of the document, set the **REP_OP_StoreImageInReportingTables** parameter to the following:

    ```
    REP_OP_StoreImageInReportingTables=Yes
    ```

    Note: Configure this parameter only if you are using reporting for late archiving.

15. Optional. To retrieve a document stored within the reporting tables using a URL, which is both stored against the record in the reporting database and is also available for export to a downstream system, set the REP_VL_ArchiveURL parameter to the following

    ```
    REP_VL_ArchiveURL=http://archive.system.com/Page.aspx?URN=XXXXX
    ```

Document Splitting

Within the Verifier application, the user has the ability to split a single document image into two or more document images. If the documents are invoices, then one invoice has now become two or more individual invoices. This means that a corresponding number of new records need to be created in the reporting database.

The system will create these new records at the time where the user manually reclassifies the new individual documents, which will initially have a status of *unclassified* in the AP Packaged Project workflow. Before manual reclassification, the user should ensure that all necessary splits have taken place for a given original image file, otherwise orphan reporting records will remain in the database, which would have the effect of skewing processing statistics.

If usage of the document splitting capability is required, and the system is configured to export data using the standard database, CSV or XML methods, or the file is to be archived into the reporting database, it will not be possible to use the *URN* as mapped to a component of the document filename as the document key for those exports. This is because the URN will remain

constant for multiple documents, hence database conflicts will occur, and export files may overwrite one another.

Appendix Q E-Business Suite Database Views

Introduction

The AP Packaged Project is delivered preconfigured to perform lookups and validations against an E-Business Suite database containing four read-only views, which are described later in this section.

Creating the Views in the E-Business Suite Database

The four views that are configured in the AP Packaged Project are not standard views in the E-Business Suite database and must be created before they can be used by the project. An SQL script file is provided with the project to achieve this, which is located at:

<*Installation Folder*>**\Projects\AP Project 2803\DB Scripts\WFR_AP_EBS_Views_Create.sql**

This script is provided as an aid to simplify the implementation of the AP Packaged Project in environments where the customer uses Oracle E-Business Suite as their ERP system, and may be modified as necessary to meet customer requirements. It is not part of the WebCenter Forms Recognition product. For implementations that use an ERP system other than E-Business Suite; it is the responsibility of the customer to provide the appropriate data in a format that can be used by the project.

It is assumed that a schema that will be used by WebCenter Forms Recognition (typically called **AXF**) has already been created in the EBS database, and if this is not the case, this should be done prior to executing the script. The script will create the four views in the **APPS** schema of the EBS database and will grant *SELECT* privileges on them to the **AXF** schema to use them through synonyms.

Simply execute the script file using SQL*Plus or SQL Developer connected to the E-Business Suite as the **APPS** user. During the script execution, you will be prompted to enter the password for the **APPS** user, as well as the username and password for the schema to which you want to grant privileges to perform lookups against the views. This is typically a user called **AXF**, but any schema can be used as preferred.

Note: Because the username and password for the schema will potentially be entered into the project configuration file in plain text it is highly recommended that you use a schema that does not provide UID access to the EBS tables.

The XX_OFR_PO_HEADER_V View

This view provides header-level purchase order data that can be used for PO number validation as described in *BRWPON section*.

The view defined in the script provided with the AP Packaged Project excludes incomplete purchase orders from the results.

Column Name	Type	Description
PO_HEADER_ID	NUMBER	Contains the internal ID of the purchase order header.
PO_NUMBER	VARCHAR2(20)	Contains the purchase order number. If PO number validation is enabled, this column should be mapped to the *DBPO* parameter in the *BRWPON section.*

Column Name	Type	Description
VENDOR_ID	NUMBER	Contains the internal ID of the vendor that the purchase order was issued to. If PO number validation is enabled, this column should be mapped to the *DBVendorID* parameter in the *BRWPON section*.
VENDOR_SITE_ID	NUMBER	Contains the ID of the vendor site that the purchase order was issued to. If PO number validation is enabled, this column should be mapped to the *DBSiteID* parameter in the *BRWPON section*.
STATUS	VARCHAR2(4000)	Contains the approval status of the purchase order. If PO number validation is enabled, this column should be mapped to the *DBStatus* parameter in the *BRWPON section*.
APPROVED_FLAG	VARCHAR2(1)	Contains a flag value indicating whether the purchase order has been approved. In the E-Business Suite database, this column will typically contain a **Y** if the PO is approved, otherwise it will be blank.
CURRENCY_CODE	VARCHAR2(15)	Contains the three-character ISO currency code for the currency that the purchase order was issued in. If PO number validation is enabled, this column should be mapped to the *DBCurrency* parameter in the *BRWPON section*.
PO_TYPE	VARCHAR2(25)	Contains a value identifying what type of purchase order was issued, for example **STANDARD**, **BLANKET**, etc. If PO number validation is enabled, this column should be mapped to the *DBDocType* parameter in the *BRWPON* section.
CREATION_DATE	DATE	Contains the date that the purchase order was created.
ORG_ID	NUMBER	Contains the organization ID or company code for the organization that issued the purchase order. If PO number validation is enabled, this column should be mapped to the *DBCompanyCode* parameter in the *BRWPON* section.
ORG_NAME	VARCHAR2(240)	Contains the name of the organization or company that issued the purchase order.
ORG_ID_NAME	VARCHAR2(281)	Contains a concatenation of the organization ID and name in the format: <*org_id*> <*org_name*>

The XX_OFR_PO_LINES_V View

This view provides line-level purchase order detail, and can be used for line pairing, as described in *Section: Configure Line Pairing*.

Column Name	Type	Description
PO_DISTRIBUTION_ID	NUMBER	Contains the internal distribution ID for the purchase order line.
PO_HEADER_ID	NUMBER	Contains the internal ID of the corresponding purchase order header.
VENDOR_ID	NUMBER	Contains the internal ID of the vendor that the purchase order was issued to.
PO_NUMBER	VARCHAR2(20)	Contains the purchase order number. This column should be mapped to the *DBPO* parameter in the *Configure Line Pairing section* if the *GetPOLinesFromDB* parameter is set to **YES**.
PO_LINE_ID	NUMBER	Contains the internal ID of the purchase order line.
ITEM_DESCRIPTION	VARCHAR2(240)	Contains the item description for the purchase order line.

Column Name	Type	Description
		This column should be mapped to the *DBDESCRIPTION* parameter in the *Configure Line Pairing section* if the *GetPOLinesFromDB* parameter is set to **YES**.
MATERIAL_NUMBER	VARCHAR2(25)	Contains the material number for the purchase order line. This column should be mapped to the *DBMATERIALNO* parameter in the *Configure Line Pairing section* if the *GetPOLinesFromDB* parameter is set to **YES**.
UNIT_OF_MEASURE	VARCHAR2(25)	Contains the unit of measure for the purchase order line. This column should be mapped to the *DBUOM* parameter in the *Configure Line Pairing section* if the *GetPOLinesFromDB* parameter is set to **YES**.
UNIT_PRICE	NUMBER	Contains the unit price for the purchase order line. This column should be mapped to the *DBUNITPRICE* parameter in the *Configure Line Pairing section* if the *GetPOLinesFromDB* parameter is set to **YES**.
QUANTITY_ORDERED	NUMBER	Contains the quantity ordered for the purchase order line. This column should be mapped to the *DBPOQUANTITY* parameter in the *Configure Line Pairing section* if the *GetPOLinesFromDB* parameter is set to **YES**.
LINE_TOTAL	NUMBER	Contains the total price for the purchase order line. This is the result of the calculation: <*unit_price*> * <*quantity_ordered*> This column should be mapped to the *DBPOTOTAL* parameter in the *Configure Line Pairing section* if the *GetPOLinesFromDB* parameter is set to **YES**.
QUANTITY_RECEIVED	NUMBER	Contains the quantity that has been received for the purchase order line. This column should be mapped to the *DBTOTALQUANTITYDELIVERED* parameter in the *Configure Line Pairing section* if the *GetPOLinesFromDB* parameter is set to **YES**.
QUANTITY_INVOICED	NUMBER	Contains the quantity that has already been invoiced for the purchase order line. This column should be mapped to the *DBTOTALQUANTITYINVOICED* parameter in the *Configure Line Pairing section* if the *GetPOLinesFromDB* parameter is set to **YES**.
ORG_ID	NUMBER	Contains the organization ID or company code for ship-to organization.
BUSINESS_UNIT	VARCHAR2(25)	Contains the business unit segment from the GL code combination. This column should be mapped to the *DBBUSINESSUNIT* parameter in the *Configure Line Pairing section* if the *GetPOLinesFromDB* parameter is set to **YES**.
LINE_NUM	NUMBER	Contains the number of the purchase order line as it appears on the purchase order. This column should be mapped to the *DBLINE* parameter in the *Configure Line Pairing section* if the *GetPOLinesFromDB* parameter is set to **YES**.
PURCHASE_BASIS	VARCHAR2(30)	Contains the purchase basis of the line item, for example, **GOODS** or **SERVICES**, etc.

The XX_OFR_SUPPLIERS_V View

This view provides supplier information, and can be used to create the Vendor ASSA data pool, as described in Section: *Configure ASA Section in the <project>.ini File*.

Column Name	Type	Description
VENDOR_INDEX	VARCHAR2(81)	Contains a unique vendor identifier, in the format: <*vendor_id*>~<*vendor_site_id*> This column should be identified as the ID column when configuring the Vendor ASSA pool, as described in Appendix: Configure the Vendor ID Field without Using Partition. It should also be mapped to the *ID* parameter in the *Configure the BRWSRC Table section*.
VENDOR_ID	NUMBER	Contains the internal ID for the vendor. This column does not need to be mapped in the *Configure the BRWSRC Table section*.
VENDOR_NAME	VARCHAR2(240)	Contains the name of the vendor. This column should be mapped to the *Name* parameter in the *Configure the BRWSRC Table section*.
VENDOR_NUMBER	VARCHAR2(30)	Contains the external ID for the vendor. This column should be mapped to the *ExternalVendorID* parameter in the *Configure the BRWSRC Table section*.
VENDOR_SITE_ID	NUMBER	Contains the site ID for the vendor site. This column should be mapped to the *SiteID* parameter in the *Configure the BRWSRC Table section*.
ADDRESS_LINE1	VARCHAR2(240)	Contains the first line (street) of the address for the vendor site. This column should be mapped to the *Address1* parameter in the *Configure the BRWSRC Table section*.
ADDRESS_LINE2	VARCHAR2(240)	Contains the second line (street) of the address for the vendor site. This column should be mapped to the *Address2* parameter in the *Configure the BRWSRC Table section*.
CITY	VARCHAR2(60)	Contains the city for the vendor site. This column should be mapped to the *City* parameter in the *Configure the BRWSRC Table section*.
STATE	VARCHAR2(60)	Contains the state, county or region for the vendor site. This column should be mapped to the *State* parameter in the *Configure the BRWSRC Table section*. For states in the US, this should be expressed as the 2-character state identifier.
POSTAL_CODE	VARCHAR2(80)	Contains the postcode or zip code for the vendor site. This column should be mapped to the *Zip* parameter in the *Configure the BRWSRC Table section*.
COUNTRY	VARCHAR2(100)	Contains the country for the vendor site. This column should be mapped to the *Country* parameter in the *Configure the BRWSRC Table section*. This should be expressed as the 2-character ISO country code.
VAT_REGISTRATION	VARCHAR2(80)	Contains the VAT registration number for the vendor site, if applicable.

Column Name	Type	Description
		This column should be mapped to the *VATRegNo* parameter in the *Configure the BRWSRC Table section* if VAT number compliance validation is enabled, as described in *Appendix: Configuring the VAT Number Compliance Check*.

VENDOR_TYPE	VARCHAR2(30)	Contains a flag that identifies the type vendor type for the site.
		This can be used to identify a utility vendor by configuring the *UtilityFlag* parameter in the *Configure the BRWSRC Table section* with the column name, and the *UtilityAlias* parameter in the *BRWNUM section* with the value in this column that identifies a utility vendor. For example:
		`NUM_VL_UtilityAlias=UTILITY`
		`SRC_VL_UtilityFlag=VENDOR_TYPE`

INVOICE_TYPE	VARCHAR2(5)	Contains a value indicating whether the vendor site is expected to issue PO-based invoices, or whether they can legitimately issue invoices that do not reference a purchase order. This column can be mapped to the *InvoiceType* parameter in the *Configure the BRWSRC Table section* if required.
CURRENCY_CODE	VARCHAR2(15)	Contains the 2-character ISO currency code to identify the currency that the vendor site will issue invoices in. This column can be mapped to the *Currency* parameter in the *Configure the BRWSRC Table section*.
ORG_ID	NUMBER	Contains the organization ID or company code of the organization that the vendor site relates to.
ORG_NAME	VARCHAR2(240)	Contains the name of the organization that the vendor site relates to.
ORG_ID_NAME	VARCHAR2(281)	Contains a concatenation of the organization ID and name in the format: <*org_id*> <*org_name*>
SITE_NAME	VARCHAR2(15)	Contains the name of the vendor site.

The XX_OFR_INVOICES_V View

This view provides historical header-level invoice information. It can be used for invoice number format validation as described in *BRWNUM section*, and also for checking for duplicate invoices as described in *Section: Duplicate Invoice Number Checking*.

Column Name	Type	Description
VENDOR_ID	NUMBER(15)	Contains the internal ID of the vendor who issued the invoice. Where invoice number format validation is enabled, this column should be mapped to the *VendorID* parameter in the BRWNUM section. Where duplicate invoice number detection is enabled, this column should be mapped to the *CDIDBSupplierID* parameter in the *WFR section*.
VENDOR_SITE_ID	NUMBER(15)	Contains the site ID of the vendor site that issued the invoice. Where duplicate invoice number detection is enabled, this column should be mapped to the *CDIDBSupplierSite* parameter in the *WFR section*.
INVOICE_ID	NUMBER(15)	The internal ID of the invoice. Where invoice number format validation is enabled, this column should be mapped to the *RecID* parameter in the *BRWNUM section*.

Column Name	Type	Description
INVOICE_NUMBER	VARCHAR2(50)	The invoice number as provided by the vendor and as stated on the invoice. Where invoice number format validation is enabled, this column should be mapped to the *InvoiceNumber* parameter in the *BRWNUM section*. Where duplicate invoice number detection is enabled, this column should be mapped to the *CDIDBInvoiceNumber* parameter in the *WFR section*.
INVOICE_DATE	DATE	The date that the invoice was issued by the vendor, as stated on the invoice.
INVOICE_TYPE	VARCHAR2(25)	The type of invoice. For standard invoices, this value is typically **STANDARD**, and for credit notes, it is typically **CREDIT** in an E-Business Suite database. Where invoice number format validation is enabled, values found in this column should be mapped to the *InvoiceAlias* and *CreditAlias* parameters in the *BRWNUM section*.
ORG_ID	NUMBER(15)	The ID of the organization or company code that the invoice was issued to. Where duplicate invoice number detection is enabled, this column should be mapped to the *CDICBCompanyCode* parameter in the *WFR section*.

Appendix R AP Packaged Project Database Password Encryption

Password encryption for database connections is a standard feature in AP Packaged Project.

When configuring a database connection string, if you are using SQL-based authentication, then you need to include the database user name and password in the `<project>.ini` file. Your IT security policies may display a plain text password within this file, and therefore, a form of password encryption is required.

Generate an Encrypted AP Packaged Project Database Password

To generate an encrypted AP Packaged Project database password, complete the following steps:

1. Make sure that the **CdrCryptDLL** was registered correctly during the AP Packaged Project installation, by running the `Register Applications.bat` file.
2. In Notepad, open the **Password_Encrypt.bat** file.
3. The file is comprised of the following components:
   - The location of `DstCrypt.exe` file, which forms part of the core product installation.
   - The password to be encrypted.
   - The public key used for the encryption.
   - The location of the output file that contains the encrypted password.
4. Edit the file to reflect the appropriate file paths, the public key to be used, and the plain text password.
5. Save and run the BAT file.
6. Navigate to the location where the output file was created and open in Notepad. The encrypted string following `Text [password]` encoded to is the encrypted version of the plain text password.

Note: The encrypted string may change each time the `Password_Encrypt.bat` file is run, even if the plain text password remains the same. This is normal functionality.

Configure the AP Packaged Project Database to Use an Encrypted Password

To configure the database to use an encrypted password, complete the following steps:

1. Open the **<project>.ini** file.
2. Navigate to the **SQL** section.
3. Each SQL group contains an `EncryptedPassword` parameter. Map the encrypted password against the parameter as follows:

```
 SQL_VL_01_ConnectionString=Provider=SQLOLEDB.1;Password=test;Persist
Security Info=TRUE;User ID=PIC;Initial Catalog=PIC;Data
Source=myMachine\sqlexpress
 SQL_VL_01_EncryptedPassword=[password]
```

4. In the connection string for the password, the included password parameter can either be removed, or the password itself can be deleted. If the plain text password is left in the connection string, the system continues to use it instead of the encrypted password.

Appendix S Alternate Payees

An alternate payee is the party that must receive payment for an invoice that is different from the party that supplied the goods or services.

The Invoice Vendor Identification functionality includes the option of specifying an alternate payee.

For example, if an invoice is processed for vendor 1000, and an alternate payee of 2000 is specified. The invoice is created in the downstream ERP system against vendor 1000, but when the payment run is executed, vendor 2000 is the recipient of the funds.

In an ERP system, the relationship between a vendor and one or more alternate payees is tightly defined in the vendor master data. At the time of invoice entry, a user may only specify an approved alternate payee that has been set up against the invoice vendor. Possible alternate payees for a vendor are known as permitted payees. Alternate payees exist as vendor records within the vendor master data in their own right, but they are typically blocked for direct posting and may have a different account group. The alternate payee field and associated functionality can be activated per profile ID.

An alternate payee can be activated in the BRWFLD database table or in the BRWVND database table.

If you configure an alternate payee in the BRWFLD database table, the **Alternate Payee** field becomes active within the `<project>.sdp` file.It is included as a supplementary field to the vendor ID on the Dynamic Verifier form. However, the field only functions in relation to settings in the BRWFLD database table, effectively becoming an extra free text field that can be used as necessary. The `UserExitAlternatePayeeValidate` user exit is also available for any custom validation logic.

If you configure an alternate payee in the BRWFLD database table and in the BRWVND database table, then the field is available for use and the standard functionality around the field usage is activated. Activating an alternate payee in the BRWVND database table without configuring an alternate payee in the BRWFLD database table has no effect, because the alternate payee is a supplementary field to the vendor ID, the vendor ID must also be an active field within the profile.

Activate the Alternate Payee Field

To activate the Alternate Payee field, complete the following steps:

1. Open the **BRWFLD** table.
2. Set the **Active** column to `TRUE`.
3. Save the changes.

Alternate Payee Functionality

If the alternate payee functionality has been activated, when a vendor is identified, either by the ASE or derived from a purchase order, the system checks to see whether the vendor has any permitted payees assigned. If it does, and one of the permitted payees is identified on the invoice, then this payee is copied into the **Alternate Payee** field.

If none of the permitted payees appears to match the invoice details, the **Alternate Payee** field is marked as invalid, and the document is sent to Verifier for review.

Within Verifier, the user has the option to enter an alternate payee or to confirm that no alternate payee is appropriate for the document. A **Payee selection** button is provided on the Verifier form which invokes a dialog box displaying the permitted payees available for the vendor.

An alternate payee is only accepted by the system as long as it is in the list of permitted payees assigned to the vendor. The vendor record for the alternate payee need not exist in the vendor master extract, but if it does not, then no name and address details are displayed in Verifier. If a payee is entered who is not in the list of permitted payees, the system displays a message, and the content in the **Alternate Payee** field is removed.

If vendor partitioning is being used, it is not necessary to prefix the list of permitted payees with the partition ID and separator. The system assumes that the permitted payees belong to the same partition as the vendor to which they are assigned. For that reason, it is not possible to have permitted payees spanning multiple partitions.

Configure Alternate Payee Validation

To configure alternate payee validation, complete the following steps:

1. Open the **BRWVND** table.
2. Set the **CheckForAlternatePayees** column to `TRUE`.
3. Save the changes.

Assign Permitted Payees to Vendors

To assign permitted payees to vendors, complete the following steps:

1. Open the **BRWVendorMaster** table.
2. In the **Permitted Payee** column, enter the permitted payees as a comma-separated list. For example, `2000,4000`.
3. In the **Permitted Payee** column, enter the permitted payees as a comma-separated list. For example, `2000,4000`.
4. Save the changes.
5. Open the **BRWSRC** table.
6. In the **Permitted Payee** column, enter the names of the payee.
7. Save the changes.