precisely

EngageOne Enrichment

PCE Migration Guide

Version 7.4.1

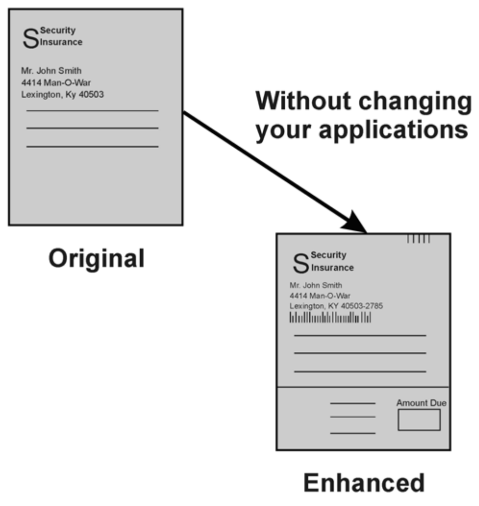
Table of Contents

1 - Introduction to EngageOne Enrichment

In this section

What is EngageOne Enrichment?

Enrichment processes and manipulates print streams to support advanced printing and distribution strategies. Enrichment modifies the output from existing applications independently; therefore, it requires no changes to your business applications.



Enrichment significantly increases productivity and reduces turnaround to implement new print processing applications.

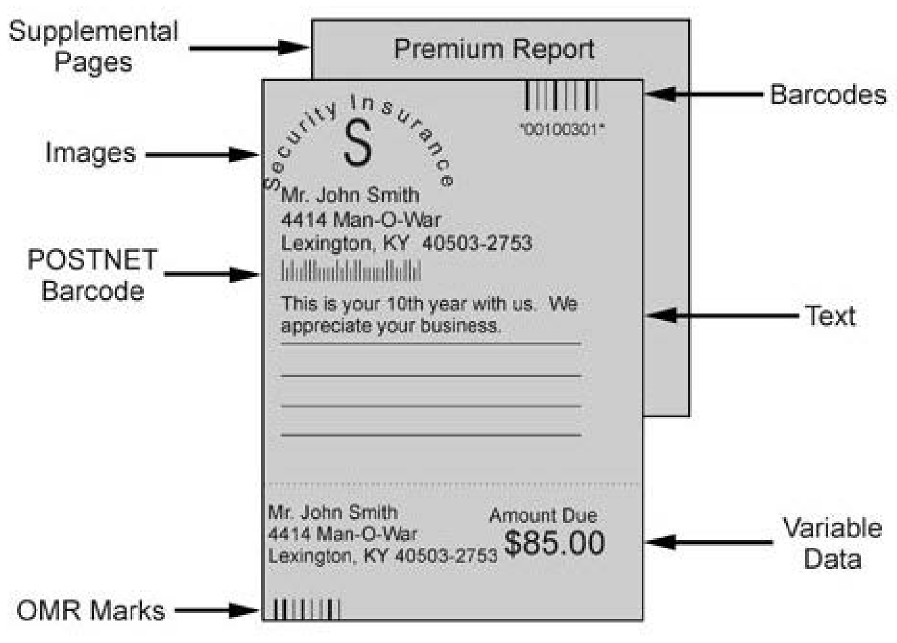
Enrichment allows you to perform many print stream enhancement functions, including the following:

- Content enhancement
  - Add personalized messages
  - Move text and change fonts
  - Add new text in white space
  - Move and modify barcodes
- Postal automation
  - Add POSTNET™ barcodes
  - Add 4-State and Intelligent Mail® barcodes
  - Add ChinaPost barcodes
  - Add ZIP+4®
  - Standardize addresses, when used in conjunction with an address validation product such as Finalist
  - Postal presorting, when used in conjunction with a postal presort product such as Mailstream Plus
  - Move update processing
  - Combine mail streams produced from multiple applications into a single mail stream

- Inserter control
  - Remove or modify old barcodes
  - Add advanced barcodes
  - Add sequence numbers
  - Build file-based insertion controls
- Sort print streams
- Consolidate/merge print streams

For sample applications that show how to accomplish some of these tasks, see the *Enrichment Sample Applications Guide*.

The following illustration demonstrates several kinds of enhancements you can make to a print stream using Enrichment.



Enrichment can handle a variety of output types from your business applications: flat files, AFP line-data files, AFP Mixed Mode files, fully composed AFPDS files, Xerox DJDE, Metacode files, PCL files, PostScript and so on. Because only the output from your business applications is processed, you do not need to change the business applications themselves.

Why migrate to Enrichment?

Enrichment offers the following advantages over PCE:

- Support for DJDE line data and PCL
- Enhanced barcode support and manipulation
- Better performance and application efficiency
- Better integration of document composition and document finishing processes

In addition, Enrichment offers optional features that are not available with PCE:

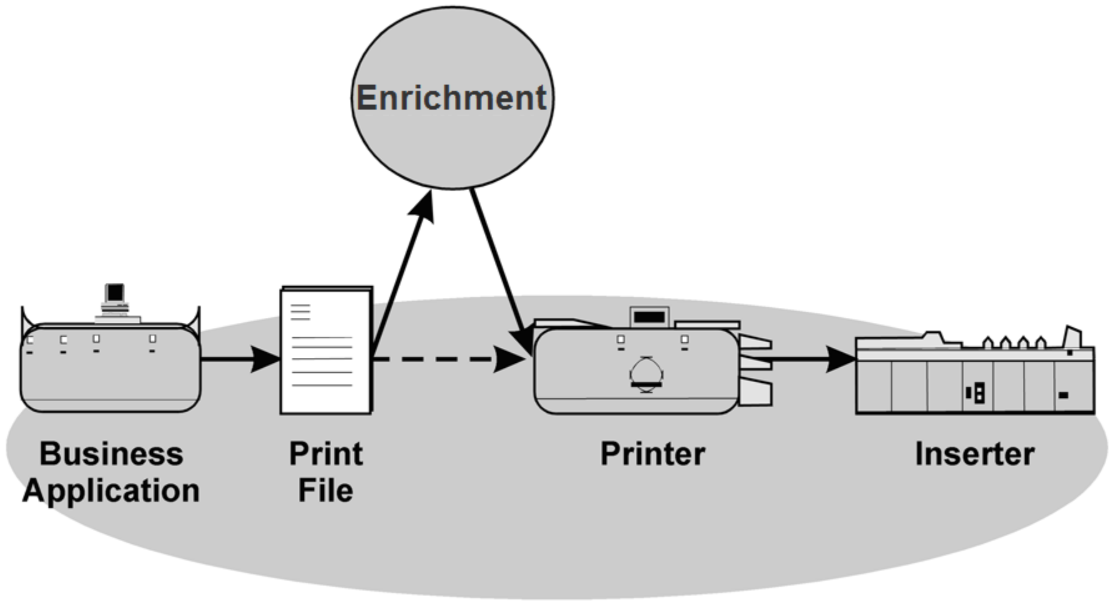
- Integration with postal tools for postal presort and address validation
- Support of Generate and non-Generate print streams
- Windows-based GUI development tool with print stream viewer

Key differences between PCE and Enrichment

- PCE can write text to the screen but Enrichment cannot. All communication is through a log file or a user-created output file.
- Enrichment is not supported on the IBM iSeries platform.
- Enrichment does not support VPS print streams.
- Enrichment does not support IJPDS data streams.

How does Enrichment work?

Enrichment typically plugs into your existing process. This usually means adding an extra step to the JCL, UNIX shell script, or Windows batch file you use for print processing. You can also run Enrichment as an independent application.



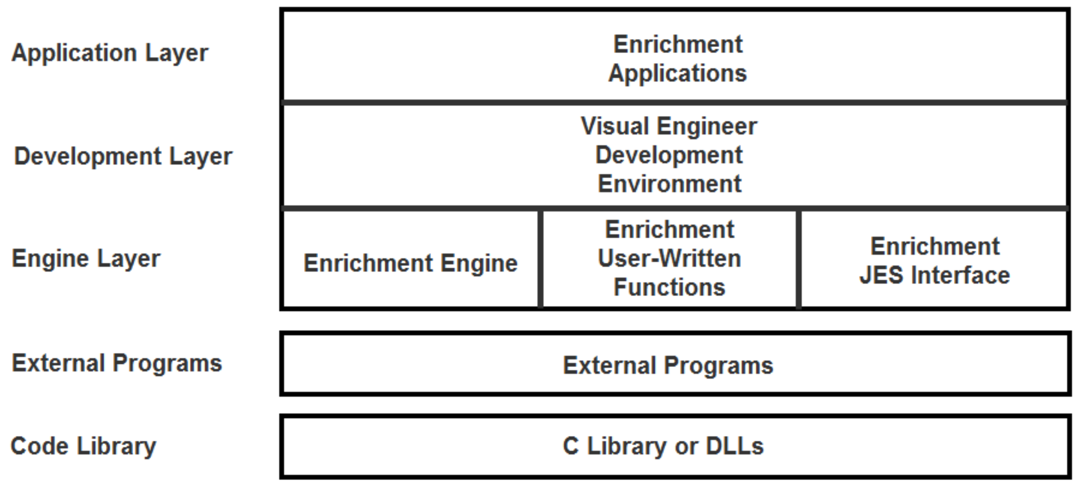
Without Enrichment, your business application creates a print file that prints and runs through an inserter. If changes to the content or appearance of the print file are required, you must go back to the business application and start over. This can be costly and time-consuming.

With Enrichment, your business application creates a print file that you run through Enrichment for the appropriate enhancements. The enhanced output prints and runs through the inserter.

> **Note:** Enrichment does not transform print streams from one format to another (for instance, from AFP to Metacode).

Enrichment architecture

The following diagram illustrates the Enrichment architecture:



C library or DLLs

The C library (for mainframe systems or UNIX) or DLLs (for Windows) serve as the code library accessed by the Enrichment engine when running applications. Depending on the platform, various libraries may be required.

External programs

External programs include presorting, sorting, and mail cleansing software called by Enrichment. These programs are not considered part of Enrichment.

Enrichment engine

The Enrichment engine is the component that processes print streams according to instructions coded in the Enrichment language (control file and rules). The engine runs on a variety of platforms. Code created to run on the engine under one platform will run on all others unless it accesses platform-dependent data, external programs, or user-written functions. This guide describes coding methods for the Enrichment engine.

Enrichment user-written functions

Users can extend Enrichment functionality by writing their own functions in a variety of languages. The object code for the user-written functions must be executable code that has been compiled and linked. On Windows, it may be a DLL.

Enrichment JES interface

The JES Interface provides the ability for JES spool volume data to be used as input to a Enrichment application. There are two JES interfaces: a non-SAPI interface and a SAPI interface. The non-SAPI interface directly accesses the JES spool volumes. The SAPI interface uses the IBM Sysout Application Programming Interface (SAPI) to retrieve spool data, and does not directly access spool volumes.

Enrichment Visual Engineer development environment

The Enrichment Visual Engineer development environment is a tool used by developers to build and test Enrichment applications. It runs on Windows and can improve development productivity compared to manually coding and testing applications in a standard editor.

Enrichment applications

Programmers or printer specialists write Enrichment applications to perform specific functions on specific print files. Each application is coded using the Enrichment language, which includes two basic components:

- Control File — Defines the objects to be processed by Enrichment using object-oriented constructs.
- Rule File — Defines conditional processing logic for the application in traditional programming code. While a rule file is optional, it is used for most applications. The rule file is composed of sections executed at different points in the application. In many cases the rule file is embedded in the control file so it is not an actual file. However, the rule file can be a separate file that is called from a control file.

Some users build programs that automatically write Enrichment code and generate Enrichment applications.

Supported print streams

Enrichment supports the following print stream formats:

- AFP
- DJDE
- Fixed length record line date
- Line data
- Metacode
- PCL
- PostScript

AFP print streams

AFP is a printer control language for AFP-compatible printers. There are several different forms of AFP:

- AFP mixed mode, which contains a mix of line printer data and AFP records.
- AFP line data, which is simply a data file with an associated map (PAGEDEF) that tells the printer where to put each data field on the page.
- AFPDS, which contains a series of print commands in compact form.

An AFP record contains one or more AFP commands. There are many different types of AFP commands. Each command has a specified layout and associated data. You can use Enrichment to modify the contents of these commands or add new AFP records. Modification of commands is normally done with a field replacement process. New records are automatically added when you use the Add group tags. However, you can format records and add them yourself using a variety of methods.

Note: You can use Enrichment Visual Engineer’s Explain function to view and get more information on the AFP commands in a print stream.

Given that AFP is an IBM printer control language and is normally associated with mainframe systems, AFP files are normally in EBCDIC format.

Refer to your IBM documentation for more information on AFP commands.

DJDE print streams

Xerox DJDE data is a special record which is placed within existing line data printer files to add special Xerox print features. These records contain Dynamic Job Descriptor Entries (DJDEs). DJDEs simply instruct the printer to perform a function. These commands are not printed.

Records containing DJDEs are identified to the printer by specifying a special character sequence in a specific range of columns. In the following example, the characters #+#+DJDE are used in columns 3 through 10 to indicate a DJDE record.

```
  #+#+DJDE ASSIGN=(1,3);
  #+#+DJDE ASSIGN=(2,11);
  #+#+DJDE ASSIGN=(6,23);
  #+#+DJDE ASSIGN=(7,73);
  #+#+DJDE ASSIGN=(8,80);
  #+#+DJDE BEGIN=(9.0,0.4);
  #+#+DJDE BEGIN=(.3,0);
  #+#+DJDE FORMAT=P0635;
  #+#+DJDE MARGIN=(.05,IN);
  #+#+DJDE DUPLEX=NO,FORMS=(A064),END;
13             HEALTH INSURANCE, INC.
 3             444 CORPORATE DR
 3             BATAVIA, IL  60510
03             Telephone No. 123-223-5800
```

As shown above, DJDEs are used to specify the format of the page, margins, and duplex information. Refer to the Xerox Production Print Mode PDL/DJDE Reference for your printer for more information on DJDE commands.

Line data

Line data print streams are print streams that are prepared for printing on line printers. Line data can contain carriage control characters and table reference characters (TRC) for spacing and font selections.

Fixed-length

A fixed-length print stream is a line data print stream with no end-of-record indicator. Records are defined as a specific number of bytes rather than some other inherent record structure such as in AFPDS, or using an end of line indicator such as Line Data.

Metacode print streams

The Xerox printer control language Metacode allows users to place commands in their print stream that give printing instructions. Metacode commands control many things, including positioning of data on a page, orientation of data on the page, and font selection. Metacode print files can be very large, so Metacode commands and command options are represented in hexadecimal codes. The table below lists valid Metacode control codes. Refer to your Xerox documentation for more information on Metacode printing.

Note: Metacode print streams can also contain Xerox DJDE commands. You should process Xerox print streams without Metacode commands as DJDE line data.

PCL

Printer Command Language (PCL) is a Hewlett-Packard language.

PostScript

PostScript is a page description language supported by Adobe Systems. For complete information, see **http://www.adobe.com/products/postscript/main.html**.

2 - Enrichment equivalents of PCE commands

Most PCE commands have an equivalent Enrichment tag or function. However, not all PCE commands have a direct equivalent in Enrichment. In these cases, there are workarounds.

For complete information on the Enrichment tags and functions mentioned in this chapter, see the *Enrichment Language Reference Guide*.

In this section

Overview

Most PCE commands have an equivalent Enrichment tag or function. However, not all PCE commands have a direct equivalent in Enrichment. In these cases, there are workarounds.

For complete information on the Enrichment tags and functions mentioned in this chapter, see the *Enrichment Language Reference Guide*.

Print stream and other commonly-used commands

The following table lists the Enrichment equivalents for PCE commands used to work with documents, pages, objects, and other print stream elements, as well as other commonly-used PCE functions.

Note: Enrichment tags are in the format <TAG> and functions are in the format FUNCTION. This is the convention followed in Enrichment documentation.

PCE command	Enrichment Equivalent	Comments
add document name	<INSERTREC>	Preconstruct a TLE in the control file and use <INSERTREC>.
add DJDE	<DJDE>	The <DJDE> tag group contains tags that allow you to add DJDE commands to a print stream.
add medium map	<INSERTREC>	Preconstruct an IMM record and use <INSERTREC>.
atrim	STRIP with option B	STRIP can trim leading, trailing, and both leading and trailing characters.
barcode...using	<ADD> tag group	The <ADD> tag group is used to add a variety of objects, including text, graphics, and barcodes.
close	CLOSE	Closes a file that was previously opened by the READ or WRITE function. In Enrichment, all files are implicitly opened and closed. CLOSE for users who process so many files that system limits are a problem.

PCE command	Enrichment Equivalent	Comments
contains	POSITION or "@" operator	POSITION returns the position of a substring within a string. The “@” operator is a comparison operator that identifies whether or not one string exists within another.
convert resolution	N/A	Enrichment does not support changing the resolution.
date	DATE	DATE returns the date using the format you specify.
docoffset	N/A	Enrichment does not support obtaining file alternates.
document name	Define a field	Document name can be extracted by defining a field for document name. It is populated as it is read in.
document TLE	Define a field	TLE information can be extracted by defining a field.
equals	"=" operator	The “=” operator, like the PCE command “equals”, compares two strings.
extract document page	Logic in <RULE>	You can write a <RULE> tag group that performs this function.
include	<GETFILE>	The <GETFILE> tag works the same way as the “include” command.
in range	IF statements	There is no direct equivalent of the “in range” command but you can accomplish the same thing by using a series of IF statements.
insert object	<ADD> <APPEND> <INSERTREC> <INSERTPAGE>	These tags and tag groups can insert a variety of objects.
length	LENGTH	Both the PCE command and the Enrichment function return the length of a string.
ltrim	STRIP with option L	Removes leading characters from a string.
mapp	LOOKUP <REPLACE> PATTERN	The LOOKUP function returns a record from an external file or table in the control file. PATTERN can be used to work with substrings. The <REPLACE> tag specifies how the field being updated should be resized to handle the replacement value.

PCE command	Enrichment Equivalent	Comments
merge	<INSERTREC>	To merge two pages, remove the end-of-page record, beginning-of-page record, and records in the Active Environment Group from the page you want to merge with another. Then, use <INSERTREC> to insert the contents of the page into the merged page. For example, if you want to merge page A into page B, you would remove the end-of-page record, beginning-of-page record, and Active Environment Group records from page A, then use <INSERTREC> to insert the contents of page A into page B.
mixc	N/A	There is no equivalent function in Enrichment.
move	<SHIFT> (AFPDS only)	For AFPDS the <SHIFT> tag allows shifting of all printable elements by either a fixed or variable amount.
move page	N/A	There is no equivalent function in Enrichment.
nop	N/A	Enrichment does not provide this functionality.
number of fonts	N/A	Enrichment does not provide this functionality.
open	<INPUT> tag group <OUTPUT> tag group READ	The tags in the <INPUT> tag group specify the properties of input files. The tags in the <OUTPUT> tag group specify properties of output files. The READ function reads individual records from a specified file. Enrichment does not support delimited and DIJ files.
overwrite	<FIELD> <RULE> <OVERWRITE>	Use the <FIELD> tag group to define the strings on the page that you want to overwrite. Then, write a <RULE> with the logic you want to perform the replacement. The <OVERWRITE> function can replace arbitrary text in PTX records in AFPDS print streams.
page count	%%PAGE_NO %%TOTAL_PAGES	%%PAGE_NO contains the current page number. %%TOTAL_PAGES contains total number of pages in the current document.
pageoffset	Reprint	Use the system variables available in Reprint. For more information, see the *Enrichment Reprint User Guide*.
quit	QUIT	Note that in Enrichment QUIT is an instruction, not a function. QUIT halts execution of the current rule file section.

PCE command	Enrichment Equivalent	Comments
read	READ	In Enrichment, READ is used only for secondary input files. Primary input (printstreams) are read automatically. The Enrichment function READ returns a record from a file. **Note:** The behavior of the READ function depends on the setting of the <DOCUMENT> tag.
read...document	<DOCUMENT>	Use the <DOCUMENT> tag to identify the first and last pages in a document.
release	Set the variable to an empty string.	Enrichment does not provide variable-level memory management.
replace	<FIELD> <RULE>	Use the <FIELD> tag group to define the strings on the page that you want to replace. Then, write a <RULE> with the logic you want to perform the replacement.
rtrim	STRIP with option R	Removes trailing characters from a string.
set page name	N/A	Enrichment does not support this functionality.
string	N/A	Enrichment does not have an equivalent function.
substring	SUBSTR	Extracts part of a string.
time	TIME	The Enrichment function TIME can return the current time in several formats.
TLE	<FIELD>	Use the <FIELD> tag group to identify index values.
TLE add	<ADDTYPE>	Use the <ADD> tag group to add objects, including indexes, to a print stream. In the <ADD> tag group, use the <ADDTYPE> tag to specify the type of object (TLE).
TLE delete	<FIELD>	Use the <FIELD> tag group to identify the index as a field, then delete the field.
TLE replace	N/A	Enrichment does not support this functionality.
trace	WRITE <SIDEFILE>	The WRITE command can be used to write a message to a sidefile. Use the <SIDEFILE> tag group to specify the properties of the sidefile.

PCE command	Enrichment Equivalent	Comments
translate	LOOKUP	The LOOKUP function returns a record from an external file or table in the control file.
value	RGET	RGET converts strings to numbers.
write	WRITE	WRITE writes a record to a specified sequential or partitioned data set.
write document	<OUTPUT>	The <OUTPUT> command (which is different from the <OUTPUT> tag group) indicates which output to send the document to.

Generate commands

The following commands are specific to PCE and have no Enrichment equivalent:

- change DIJelement
- DIJelement
- add document id
- document id
- font
- get resources
- read...DIJentry
- write DIJentry

Note: Enrichment provides the ability to create a Journal file, but entries are created at the document level through Enrichment's XML support, not with a command.

Procedure constructs

Procedural processing in Enrichment is handled by the rule file. A rule file defines conditional processing. It is an optional set of code that offers additional control and user processing for individual documents and pages. In the rule file you can change field values based on certain conditions, perform calculations, create and use variables, route documents to outputs, read and write files, and add inserts or banner pages.

While the rule file can be a separate file, developers often embed rule file code within the control file (using the <RULE> tag group). When it is embedded in this manner, there is no separate file that contains the rule file code. Nevertheless, rule file code that is located in the control file is still referred to as the rule file.

See the *Enrichment Developer Guide* for more information on rule file processing.

Rule file processing takes the place of the following PCE commands:

- begin procedure
- call procedure
- declare procedure
- end procedure
- return
- call user exit
- on error call

Loop constructs

Enrichment supports more types of loop constructs than PCE, including DO...WHILE, DO...UNTIL, and FOR...NEXT. These loop constructs can be used to mimic these PCE loop constructs:

- begin loop
- end loop
- exit loop
- for next

INI file preferences

In PCE, you can set date settings and other regional settings using the PCE INI file's <Preferences> section. Enrichment does not use an INI file. To set regional settings in Enrichment, use an array in the rule file. For an example of how this is done, see

Composition edit commands

The following table lists PCE commands used to create barcodes and draw lines, and their Enrichment equivalents.

PCE command	Enrichment equivalent	Comments
COLR	<COLOR>	The <COLOR> tag specifies the color for the object.
DBX	<INSERTREC>	Use <INSERTREC> to draw a box using a single PTX command (in AFP), or with similar drawing commands in PCL and Postscript.
DIL	N/A	Enrichment does not support image lists.
DHR	<ADDTYPE> Line <ORIENT> 0	Use these two Enrichment tags together to create a horizontal rule.
DO BARCODES	<ADDTYPE>	The <ADDTYPE> tag specifies the kind of barcode to add.
DPOL	N/A	Enrichment does not use overlay lists.
DVR	<ADDTYPE> with value "line" <ORIENT> with value "0"	This combination of <ADDTYPE> and <ORIENT> values will draw a vertical line.
PI	<ADDTYPE> with value "I"	The value "I" means the object being added is an image.

3 - Examples

This section contains example applications that illustrate how to perform common functions in Enrichment. Each example contains the PCE code that accomplishes the task along with the equivalent Enrichment code. For more examples of Enrichment applications, see the *Enrichment Sample Applications Guide*.

In this section

Introduction

This section contains example applications that illustrate how to perform common functions in Enrichment. Each example contains the PCE code that accomplishes the task along with the equivalent Enrichment code. For more examples of Enrichment applications, see the *Enrichment Sample Applications Guide*.

Example 1: Replacing text

This example illustrates how to replace text in a print stream.

PCE code

To replace text using PCE, you would use code similar to this:

```
* ---------------------------------------------------------------
*  Description: read entire file with loop - replace text
*  Author     : Precisely
*  Date       : Tue May 20 10:21:15 2013
* ---------------------------------------------------------------
DECLARE <N>;
DECLARE <VAR1>;
DECLARE <VAR2>;
DECLARE <VAR3>;
DECLARE <done>;

DECLARE PROCEDURE <MAIN> IS MAIN;

*****************************************************************
BEGIN PROCEDURE <MAIN>;

OPEN "letter.txt" FOR INPUT AS FILE 1 LINE;
OPEN "letterOut.txt" FOR OUTPUT AS FILE 2;

LET <VAR1> = "Internet";
LET <VAR2> = "Web 2.0";

BEGIN LOOP;

   let <N> = 1;
   READ <N> ITEMS FROM FILE 1 INTO <VAR3>;
   let <done> = <N> NE 1;
   exit loop when <done>;
   REPLACE <VAR1> IN <VAR3> WITH <VAR2>;
   // TRACE "@@<VAR3>";
   WRITE 1 ITEM INTO FILE 2 FROM <VAR3> ;

END loop;
```

```
CLOSE INPUT 1;
CLOSE OUTPUT 2;

END PROCEDURE;
```

Enrichment code

To replace text using Enrichment, first create an <Input> tag group to define the input print stream. In the <Input> tag group, create a <Field> tag group to identify the text you want to replace, and assign a user-defined variable to use when referencing the field in the <Rule> section. In the example, the variable name is *%%TextToReplace*. Next, the <Rule> section identifies the field to update and the text you want to use. In this case, “Web 2.0”. Finally, the <Output> group identifies the output file for the job.

```
/*
   Description: Use the replace tag to find the string "Internet"
                and change it to "Web 2.0"
   Author      : Precisely
   Date        : May 20 2013
*/

<input>
 <name> input
 <file> c:\MyFiles\letter.txt
 <type> I
 <doc> 1
  <field>%%TextToReplace    R
   <ref> '1' 'Internet'
   <loc> 0  -7  8
   <replace> * Y L ' '
  </field>
</input>

<rule>
    <content>
    %%TextToReplace = "Web 2.0"
    </content>
</rule>

<output>
 <name> output
 <file> C:\MyFiles\letterOut.txt
</output>
```

Example 2: Localizing date format

This example shows how to modify the format of a date for Spanish, and how you can perform a similar operation in Enrichment. This example references a list of the months of the year in Spanish. The application places the month in the print stream.

PCE code

To define regional settings in PCE, you set regional preferences in the PCI.INI file then reference that file from the PCE control file.

INI file

This is an example control file that contains the names of the months in Spanish.

```
; us-spanish.ini
; Precisely
; May 20 2013

<Symbols>
hostpath=c:\program files\group1\doc1\work center

<LicenceInfo>
CustomerName=Precisely
Keycode=PN-1111-1111-1111-1111-1111-1111-1111-1111

<Files>
Input=us-spanish.ctl
Messages=%hostpath%\client\MESSAGES.DAT
TranslationTable=%hostpath%\TEMPLATE\DOC1TTAB.ETT

<PrintDevice>
PrintStream=AFPDS
Resolution=240

<Preferences2>
monthname1="enero"
monthname2="febrero"
monthname3="marzo"
monthname4="abril"
monthname5="mayo"
monthname6="junio"
monthname7="julio"
monthname8="agosto"
monthname9="septiembre"
monthname10="octubre"
monthname11="noviembre"
monthname12="diciembre"
```

PCE file

To modify a file to use the Spanish name of the month, the PCE control file references the INI file, as shown in this example:

```
* ---------------------------------------------------------------
*  Description: read entire file with loop - replace text
*  Author     : Precisely
*  Date       : Tue May 20 10:21:15 2013
* ---------------------------------------------------------------
DECLARE <N>;
DECLARE <MONTH>;
DECLARE <DAY>;
DECLARE <DATA>(5);
DECLARE <done>;
DECLARE <PREF>;
DECLARE <COMP>;

DECLARE PROCEDURE <MAIN> IS MAIN;
```

```
*****************************************************************
BEGIN PROCEDURE <MAIN>;

let <PREF> = "SPANISH";
let <COMP> = <PREF> eq "SPANISH";

IF <COMP>;
     SET PREFERENCES TO 2;
END IF;

begin loop;

   let <N> = 5;
   READ <N> ITEMS FROM FILE 1 INTO <DATA>;
   let <done> = <N> NE 5;
   exit loop when <done>;
   let <MONTH> = value <DATA>(2);   // Month
   let <DAY> = value <DATA>(3);     // Day
   let <MONTH> = MONTHNAME <MONTH>; // Full month name
   TRACE "@@<MONTH>";
   let <COMP> = <PREF> eq "SPANISH";

   IF <COMP>;
      Let <DATA>(2) = <DATA>(3);
      Let <DATA>(3) = <MONTH>;
   ELSE;
     // USA
      Let <DATA>(2) = <MONTH>;
   END IF;

   WRITE 5 ITEM INTO FILE 2 FROM <VAR3> ;

END loop;

CLOSE INPUT 1;
CLOSE OUTPUT 2;

END PROCEDURE;
```

Enrichment code

To localize the date format using Enrichment, create a table of the month names within the control file itself. (Enrichment does not use an INI file for lookup data.) The following control file identifies the month and day fields in a print stream then modifies the month to use the Spanish month. The user-defined variable *%%style* could be changed to “English” and the English month name would be used. This control file also changes the date format from “Month Day Year” to “Day Month Year”.

```
/*
   Description:
   Author     : Precisely
   Date       : May 20 2013
*/
/* extract and replace date with month name */

<input>
  <file> c:\MyFiles\dates.txt
  <NAME> Input1
  <type> I
  <doc> 1
  <field>%%month    R
   <loc> 1P  9P  2
   <replace> * Y L ' '
  </field>

  <field>%%day    R
   <loc> 1P  12P  2
   <replace> * Y L ' '
  </field>
</input>

<table>
<name> english
<content>
1 January
2 Feburary
3 March
4 April
5 May
6 June
7 July
8 August
9 September
10October
11November
12December
</content>
</table>


<table>
<name> spanish
<content>
1 enero
2 febrero
3 marzo
4 abril
5 mayo
6 junio
7 julio
8 agosto
9 septiembre
10octubre
11noviembre
12diciembre
</content>
</table>


<rule>
<content>

  start:
  //   %%style = "ENGLISH"
  %%style = "SPANISH"

  document:
  if %%style = "SPANISH" then
    %%month = LOOKUP(table:spanish, %%month, 1,2, N)
    %%month = SUBSTR(%%month, 3)
    // Make date format Day Month Year
    %%tmp  = %%month
    %%month = %%day
    %%day =  %%tmp
  else
    %%month = LOOKUP(table:english, %%month, 1,2, N)
    %%month = SUBSTR(%%month, 3)
  endif

</content>
</rule>

<output>
   <name> Output
   <file> output.txt
</output>
```

Notices