Invoke BI Publisher Report with PeopleCode

PT 8.58
Randall Groncki

Introduction

BI Publisher is a powerful reporting tool available with PeopleTools versions 8.48 and higher. Though most demonstrations and training focus on Query based BI Publisher reports, the tool's real power is when it is embedded into the application pages where users would naturally expect the reports.

The delivered PeopleCode BI Publisher Application packages allow full control of the reports with more options to the designer. The data source structures are protected from end users while the developer has multiple options to invoke and deliver the reports.

For purposes of the demonstration, this paper is using PeopleTools 8.58, which is the latest tools available in the PUM environments at the time of its writing. We are using an XML File as the report data source.

This paper discusses using PeopleCode and other tools within the Application Designer to create and deliver BI Publisher reports within the PeopleSoft application. It assumes that the user is familiar with the fundamentals of creating a BI Publisher Report Definition:

- Define the data source and provide sample XML data file
- Create the BI Publisher Report template using the sample XML data file
- Tying all parts together with security in a BI Publisher Report Definition

The paper is divided into three sections:

- Invoking a BI Publisher report in the user's current session using a push button on the page
- Invoking a BI Publisher report from an App Engine on the Report Server as a batch process
- Demonstrates a few ways to present the report to the user including the Report Repository, popup window and emailing the report.

BI Publisher PeopleCode Object Documentation

PeopleBooks' PeopleCode API Reference has extensive documentation on the BI Publisher Classes, Methods and Properties.

See the API Reference for your PeopleTools version for details and code examples

PeopleBooks Path: PeopleTools > Development Tools > PeopleCode API Reference > BI Publisher Classes

Report Invocation and Creation – Page in User's Session

Generating the Report

Once the XML File is generated, it's very simple to create the report and give it to the user. The delivered PeopleCode Application Package handles all the BI Publisher Report functions.
This example displays the report to the user in a new popup window on their display. Given the report window, the user can print, save or do whatever they like with the result.

1) Import the application package and instantiate

```
import PSXP_RPTDEFNMANAGER:*;
&oXML_PUB = create PSXP_RPTDEFNMANAGER:ReportDefn("Report Definition name");
&oXML_PUB.Get();
```

2) Set the data source

```
&oXML_PUB.SetRuntimeDataXMLfile(&XML_Filename_path);
```

3) Process the report
This process creates the report object itself. After this step, the developer just has to decide how to present the report to the user.

```
/* Process Report */
&oXML_PUB.ProcessReport("", "", %Date, "");

/* save to clear think time functions */
CommitWork();
```

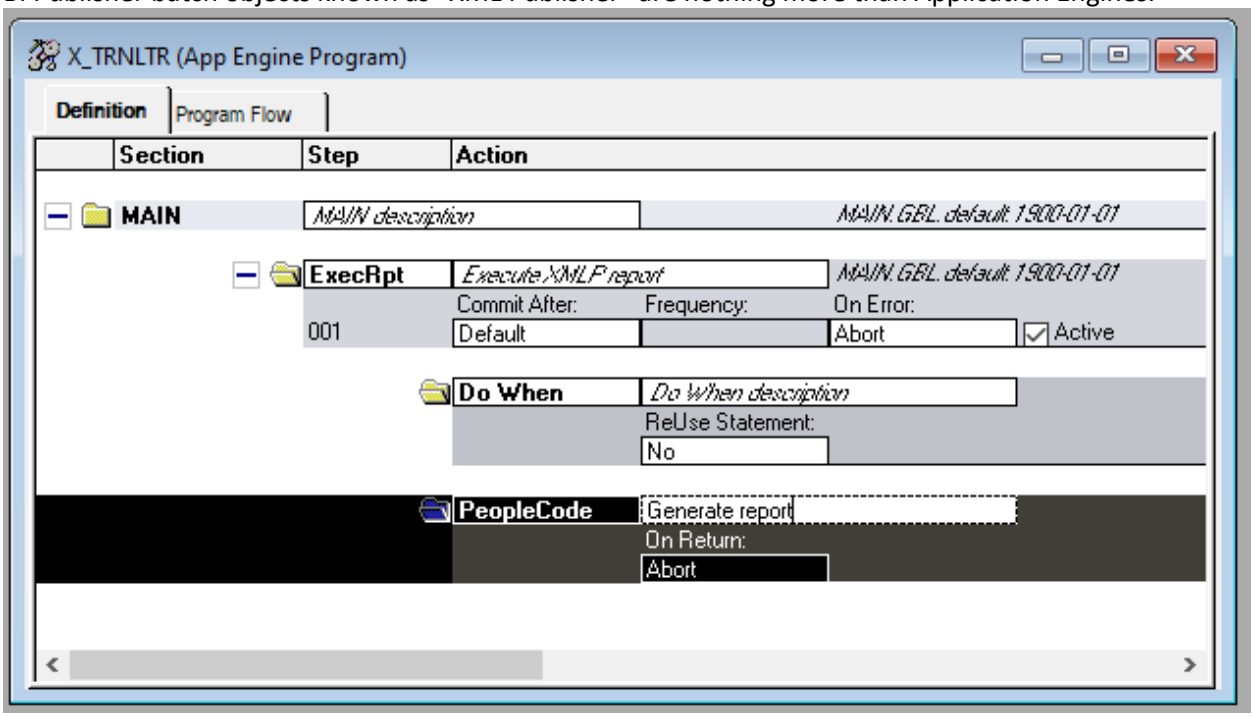
4) Display the output in a new popup window

```
&oXML_PUB.DisplayOutput();
```

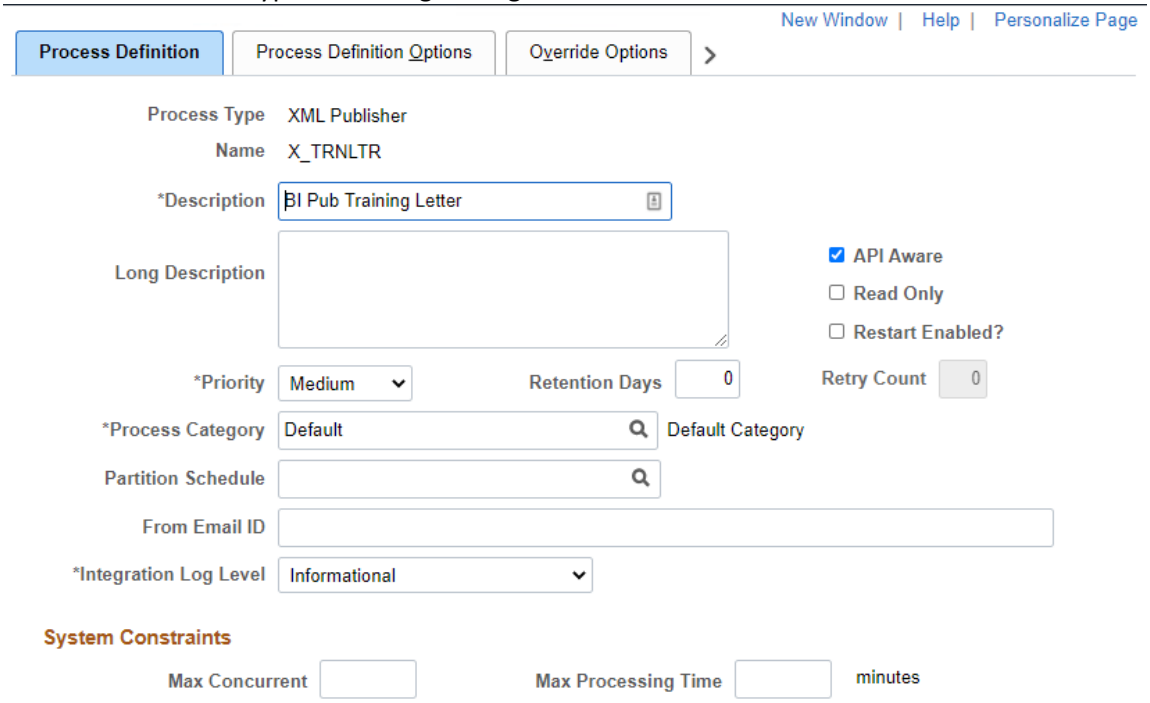
Report Invocation and Creation – App Engine in Batch

Creating the App Engine

BI Publisher batch objects known as “XML Publisher” are nothing more than Application Engines.



These App Engines have all the same capabilities as any other App Engine in the system. They just have a different “Process Type” when registering them in the Process Scheduler.



As App Engines, they should have a State Record, get run control parameters from the user and execute the report

Since an App Engine is running from the Process Scheduler and not the user's session, there are advantages for larger reports or reports using more resources to generate data

- The App Engine is not limited to the user's session time out limit
- Little or no impact to online performance
- Ability to schedule the report

Most BI Publisher App Engines use mostly dynamic "Boiler Plate" PeopleCode based on a PSQuery data source to generate the report.

To demonstrate that the AE BI Publisher PeopleCode is primarily the same as the Online PeopleCode, we are this process as an XML data source too.

1) Import the application package and instantiate

```
import PSXP_RPTDEFNMANAGER:*;

&oXML_PUB = create PSXP_RPTDEFNMANAGER:ReportDefn("Report Definition
name");
&oXML_PUB.Get();
```

2) Set the data source

```
&oXML_PUB.SetRuntimeDataXMLfile(&XML_Filename_path);
```

3) Set the destination information should the user have chosen file output (boiler plate code)

```
   /* set file path only for file output type - other types use
default temporary location */
   If %OutDestType = 2 Then /* file */
      &oXML_PUB.BurstValueAsOutSubDir = False;

      &oXML_PUB.OutDestination = %FilePath;
   End-If;
```

4) Set the Process Instance information

```
&oXML_PUB.ProcessInstance = X_TRNLTR_AET.PROCESS_INSTANCE;
```

5) Process the report

This process creates the report object itself. After this step, the developer just must decide how to present the report to the user.

```
/* Process Report */
&oXML_PUB.ProcessReport("", "", %Date, "");
```

6) Publish the report to the user's preference (boiler plate code)

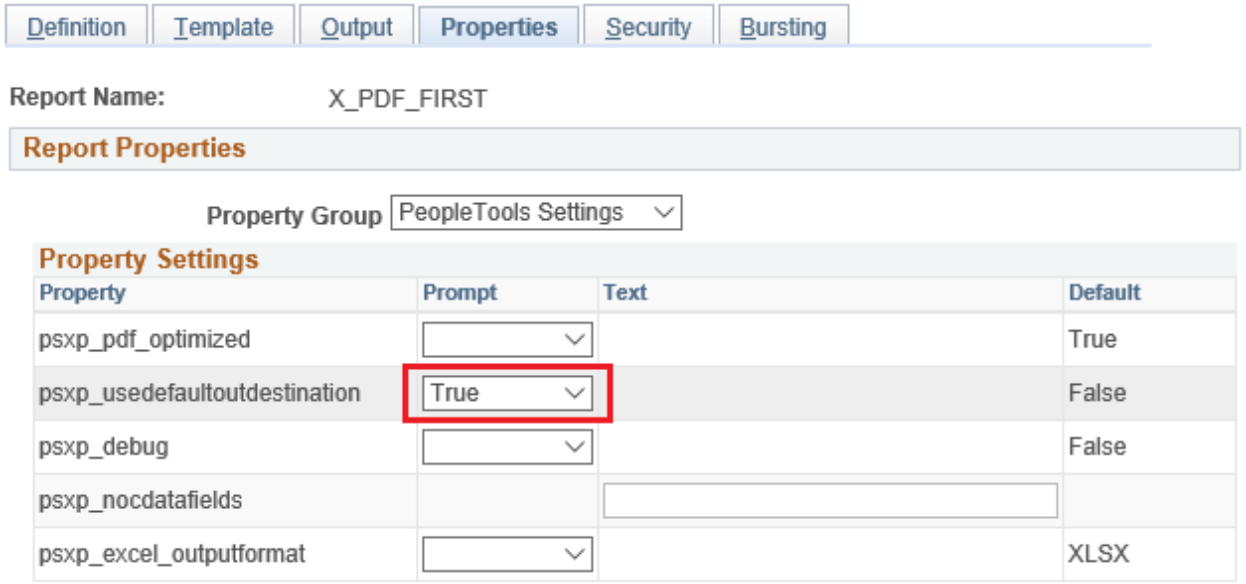
```
   /* publish */
   If %OutDestType = 6 Then /* Web */
      &oXML_PUB.Publish("", "", "", X_TRNLTR_AET.PROCESS_INSTANCE);

   Else
      If %OutDestType = 3 Then /* Printer */
         &oXML_PUB.PrintOutput(%FilePath);
      Else
         If %OutDestType = 5 Then /* Email */
            &bResult =
&oXML_PUB.EmailOutput(X_TRNLTR_AET.PROCESS_INSTANCE);
         End-If;
      End-If;
   End-If;
```

Generating the report to a file on the server

BI Publisher creates directories and files under the Application Server temp file space during operation. The fully qualified file name can be derived using a quick PeopleCode routine and properties of the XML_Publisher object.

*Note: In order for the PSXP_RPTDEFNMANAGER:ReportDefn.OutDestination class property to populate, the "*psxp_usedefaultoutdestination*" property on the report definition must be set to "True"



In the example below, the resulting report's fully qualified file name is placed in the "&ReportFilePath" string variable.

```
[...code creating report…]
&oXML_PUB.ProcessReport("", "", %Date, "");
CommitWork();

&ReportFileName = &oXML_PUB.ID | "." | Lower(&oXML_PUB.GetOutDestFormatString(2));
&ReportFilePath = &oXML_PUB.OutDestination | &sDirSep | "RptInst" | &sDirSep |
&ReportFileName;
```

Bi Publisher Output

The delivered application package contains several options for presenting the report:

1) Send to the PeopleSoft Report Repository:
   &XML_PUB_Object.Publish(&sServerName, &reportPath, &sFolderName, &processInstanceId);
2) Send the report to a printer:
   &XML_PUB_Object.PrintOutput(&DestPrinter As string);
3) Send the report to a new popup window (shown above)
   &XML_PUB_Object.DisplayOutput()
4) Mail the report
   a. If running from the Process Scheduler (Batch) the EmailOutput() uses data from the Run Control parameters to email the report
      &XML_PUB_Object.EmailOutput (*ProcessInstanceID*)

   b. If Interactive on a page, the following code sends the resultant BI Publisher report to the email address or addresses contained the "&MAIL_TO" string.

```
[...code creating report…]
&oXML_PUB.ProcessReport("", "", %Date, "");
DoSaveNow();

&ReportFileName = &oXML_PUB.ID | "." |
Lower(&oXML_PUB.GetOutDestFormatString(2));
&ReportFilePath = &oXML_PUB.OutDestination | &sDirSep | "RptInst" | &sDirSep |
&ReportFileName;

&MAIL_FLAGS = 0;
&MAIL_TO = "ReportReceiver@yourcompany.com";
&MAIL_CC = "";
&MAIL_BCC = "";
&MAIL_SUBJECT = &Email_Subject;
&MAIL_FILES = &ReportFilePath;
&MAIL_TITLES = "xml_pub.PDF";
&ret = SendMail(&MAIL_FLAGS, &MAIL_TO, &MAIL_CC, &MAIL_BCC, &MAIL_SUBJECT,
&MAIL_TEXT, &MAIL_FILES, &MAIL_TITLES, &MAIL_SENDER);

If Not (&ret = 0) Then
    MessageBox(0, "", 299, 1, "Return status from mail %1" | &ret);
End-If;
```