BillingCenter Direct Bill Invoice Generation

Reference Implementation

A DETAILED DESIGN DOCUMENT BY
GUIDEWIRE PROFESSIONAL SERVICES

Guidewire®
Deliver insurance your way™

Table of Contents

Overview

This document provides the detailed design for the reference implementation for direct bill invoice statement generation.

In this implementation, the invoice statements are generated and stored within BillingCenter. The invoice statements are generated in a PDF format using Apache FOP.

Who should read this document

1. Executives responsible for invoice management.
2. Project managers who need to meet requirements around automatically generating invoice statements within their BillingCenter implementation.
3. Functional and technical team members who will be implementing the functionality to automatically generate invoice statements with their BillingCenter implementation.

What is not included in this reference implementation

This reference implementation generates and stores the Invoice documents within Billing Center. It does not use any external document production system to generate invoices or document management system to store documents. The integration code will need to be extended if the invoice data needs to be sent to external systems for generation or storage.

Functional Design

This reference implementation demonstrates the capabilities in BillingCenter for direct bill invoice statement generation.

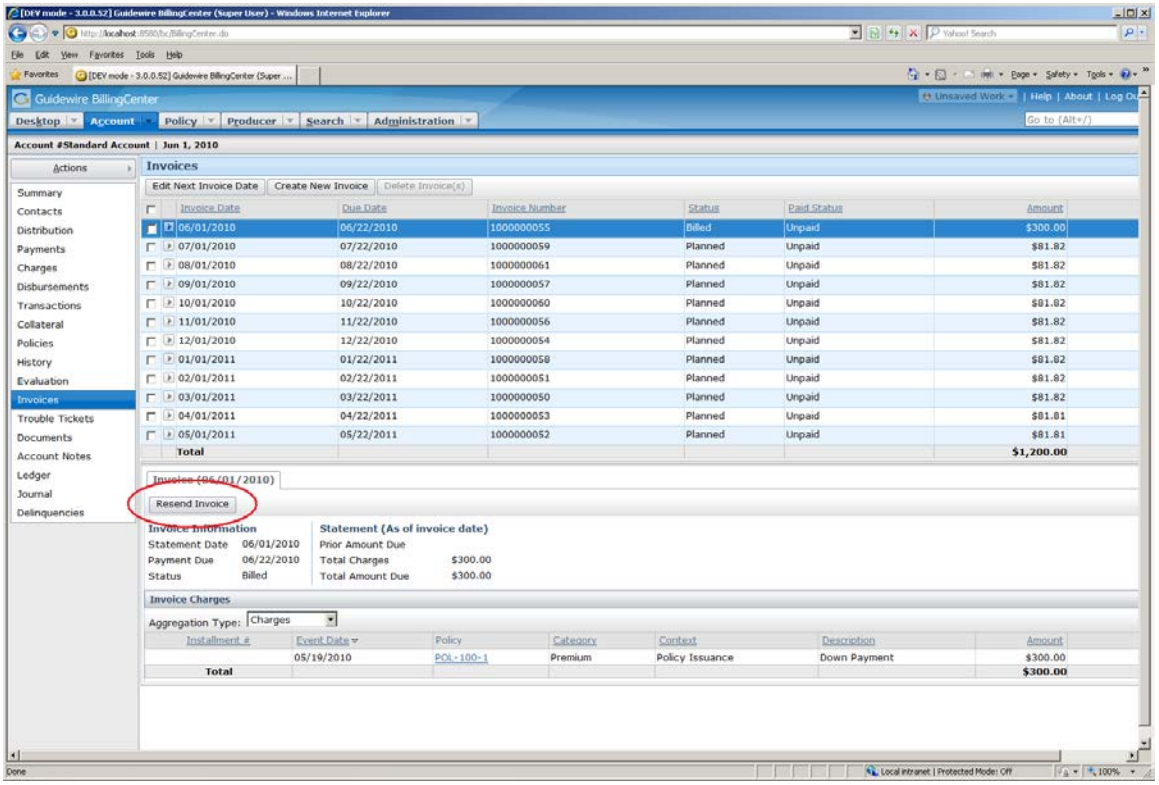
Functional Overview

BillingCenter receives billing instructions from a PAS (Policy Administration System). Billing instructions inform BillingCenter about charges or adjustments to an account. Billing instructions dictate the charges for an invoice period; invoices are added the account as a result of these charges. The parts of the charges are added to invoices as invoice items.

The invoices initially have a status of 'Planned'. Each invoice has an invoice date which when reached, changes the invoice status from 'Planned' to 'Billed'. This happens when the invoice batch process is run. When the invoice status becomes 'Billed', an invoice document is generated and sent to the Insured for payment.

Users can also resend the billed invoices by clicking on the 'Resend Invoice' button in the BillingCenter user interface (see screenshot below), which then causes the invoice statements to be regenerated.

The invoice statements can be generated in the format required by the carrier. In this reference implementation, the statements are generated in a PDF format using Apache FOP.



BillingCenter provides the following options for generating invoice documents -

1. Generate invoice document and store it in BillingCenter
2. Send invoice details to an external document production system and store it in the external document management system
3. Combination of option 1 and 2

This reference implementation demonstrates the capabilities BillingCenter has for supporting option 1, where invoice statements are generated and stored within BillingCenter. Option 2 requires BillingCenter to be integrated with external document production and management systems via integration plugins.

Technical Design

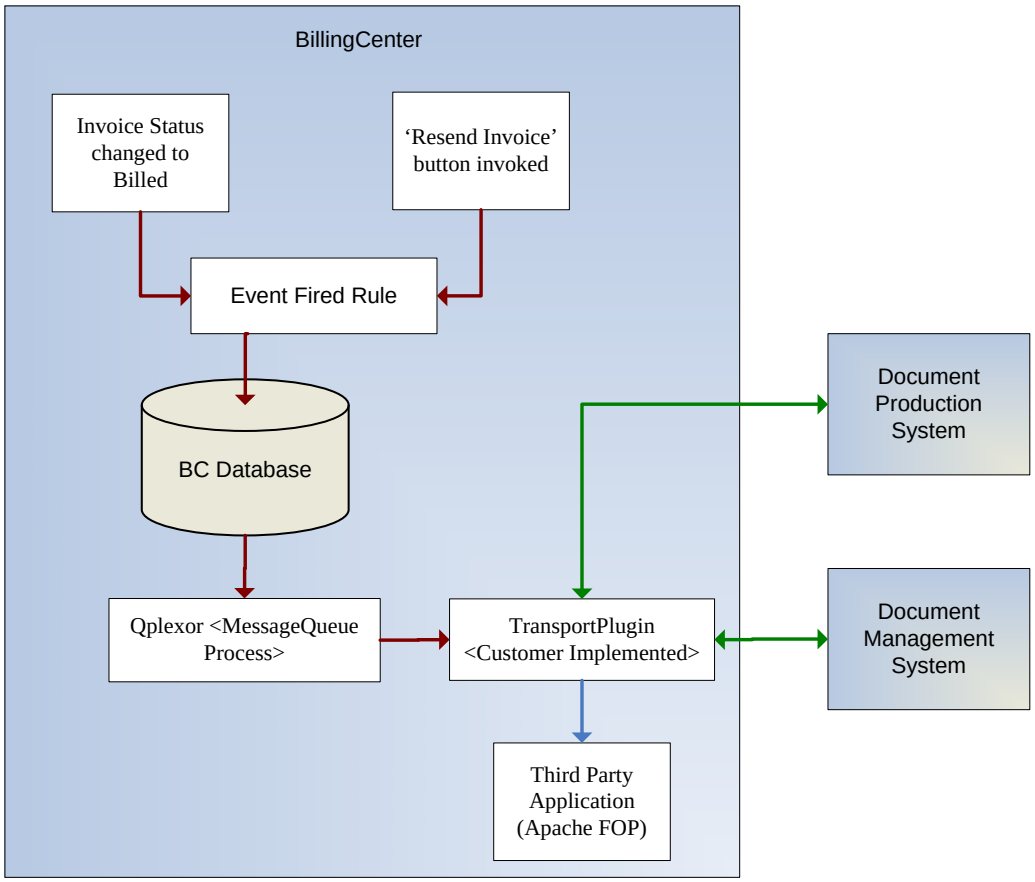
Technical Overview

The following two events are used to trigger direct bill invoice statement generation. –

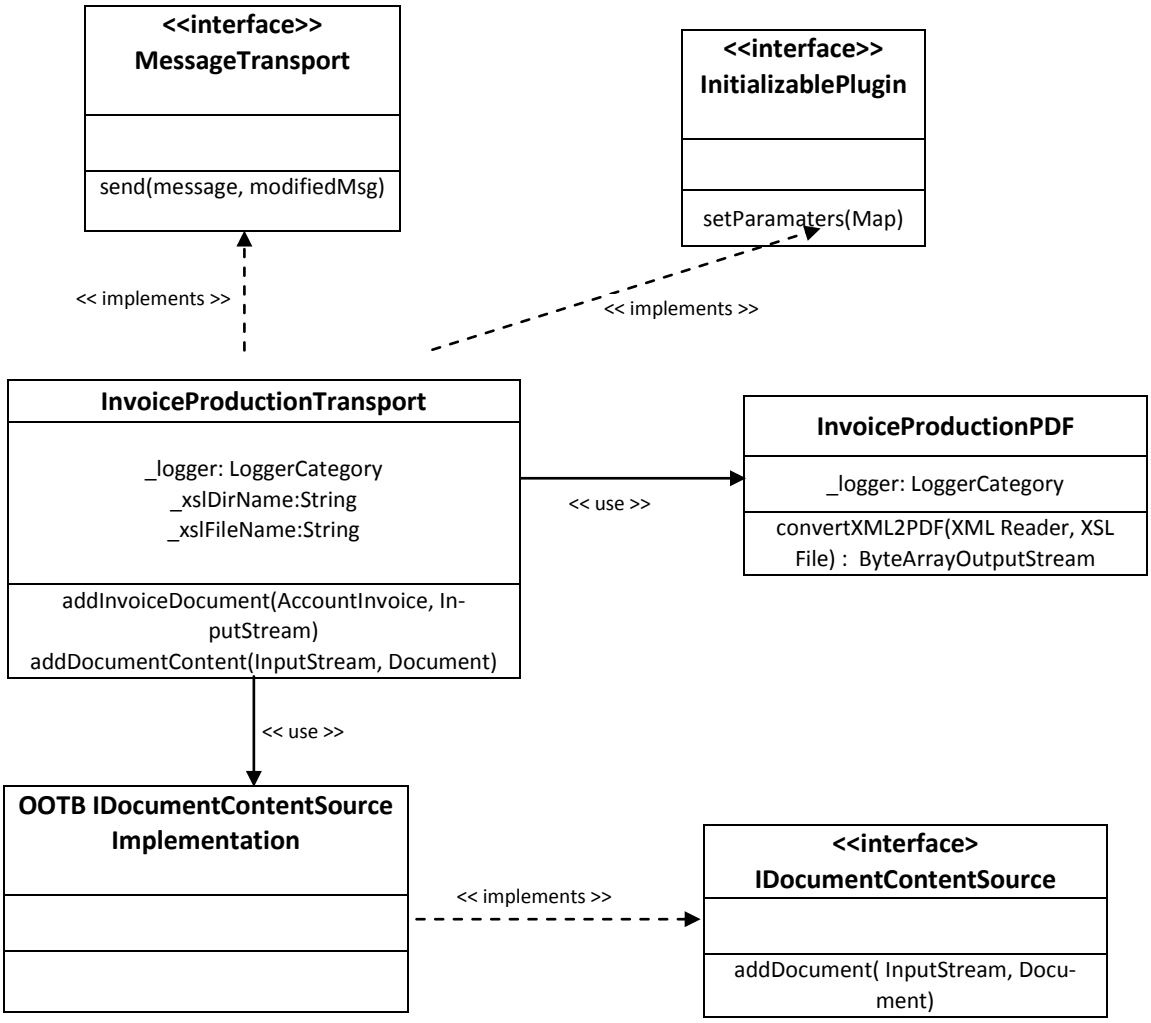
- InvoiceChanged – Event fired when invoice status is changed from 'Planned' to 'Billed'
- InvoiceResent – Event fired when the user clicks on the 'Resend Invoice' button on the user interface

Event fired rules triggered for the above events capture the invoice data, create new messages with invoice data in the payload, and submit these messages to the messaging send queue. The messages are then sent asynchronously to the InvoiceProduction destination. The MessageTransport plugin of the destination creates the invoice statements with invoice data using third party applications like Apache FOP. The generated invoice statement is then added as a document to the account. The message is acknowledged when the invoice statement has been successfully generated.

The invoice data can also be sent to a document production system and then a document management system from the MessageTransport plugin for invoice generation and storage.



Detailed Design Diagrams



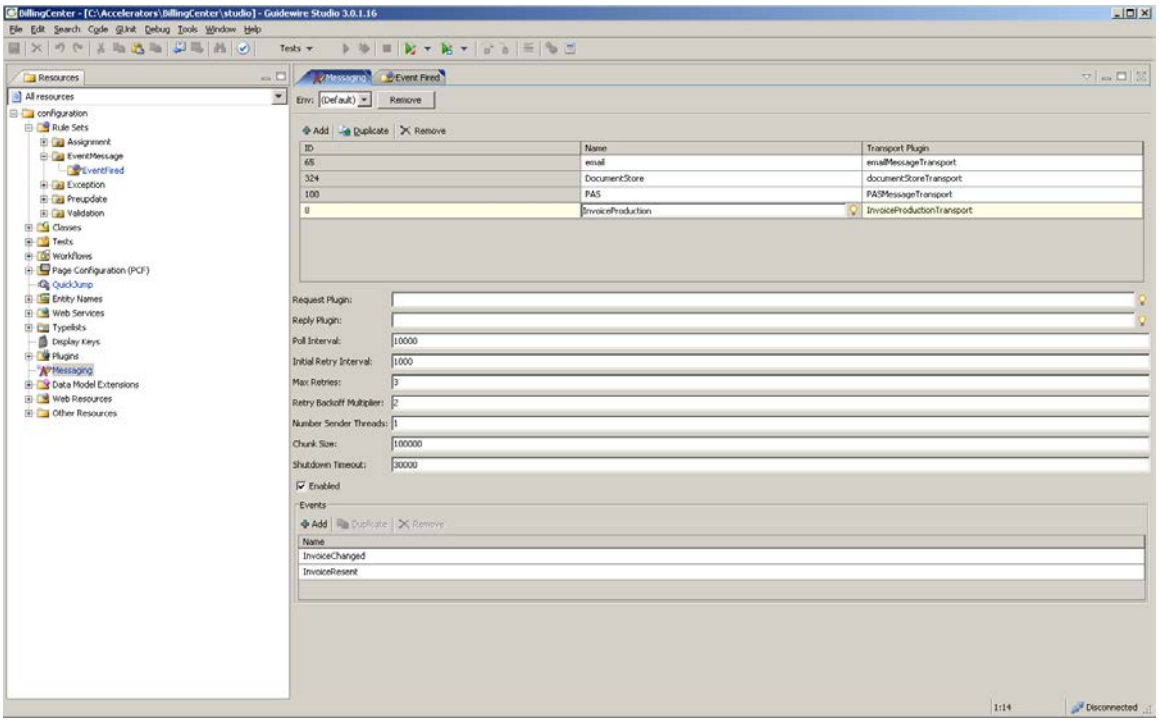
Messaging Plugin Implementation

Message Destination

A new destination is defined in BillingCenter to represent a system for Invoice Production. The destination receives messages created by the event fired rule. The message contains invoice XML data in the payload. The transport plugin InvoiceProductionTransport retrieves the invoice data from the message payload and generates invoice statements accordingly.

The InvoiceProduction destination receives notifications from the following two events:

1. InvoiceChanged – This event is fired when the invoice status is changed from 'Planned' to 'Billed' by the batch invoice process when the invoice date is reached.
2. InvoiceResent - This event is fired when the user clicks on the 'Resend Invoice' button in the BillingCenter user interface to resend the billed invoice.

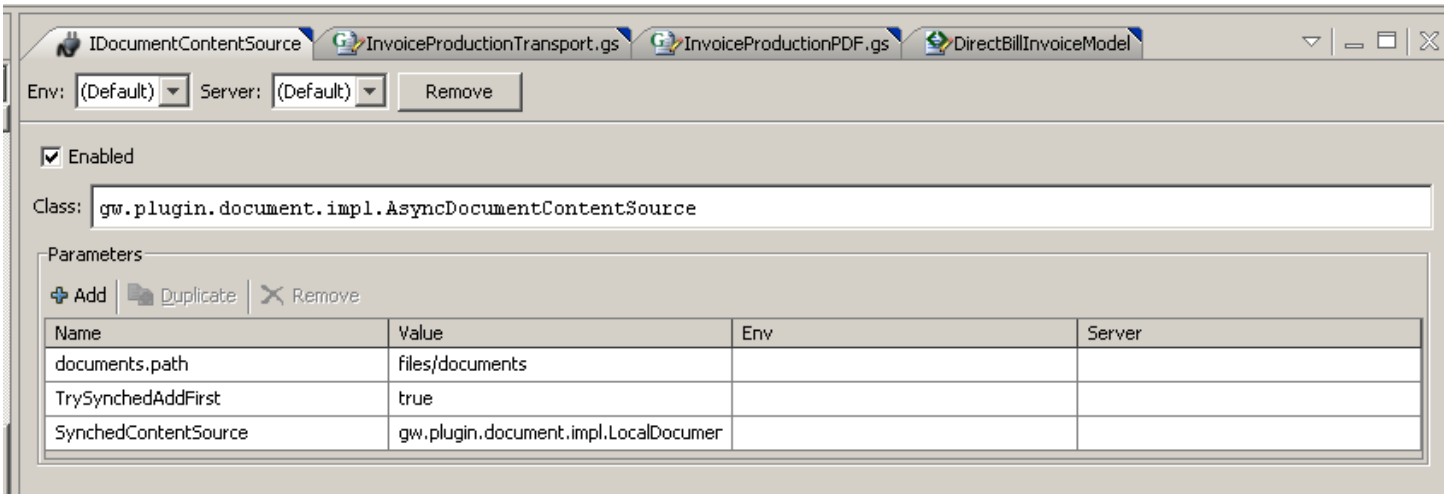


Plugin Implementation

Invoice Production Transport

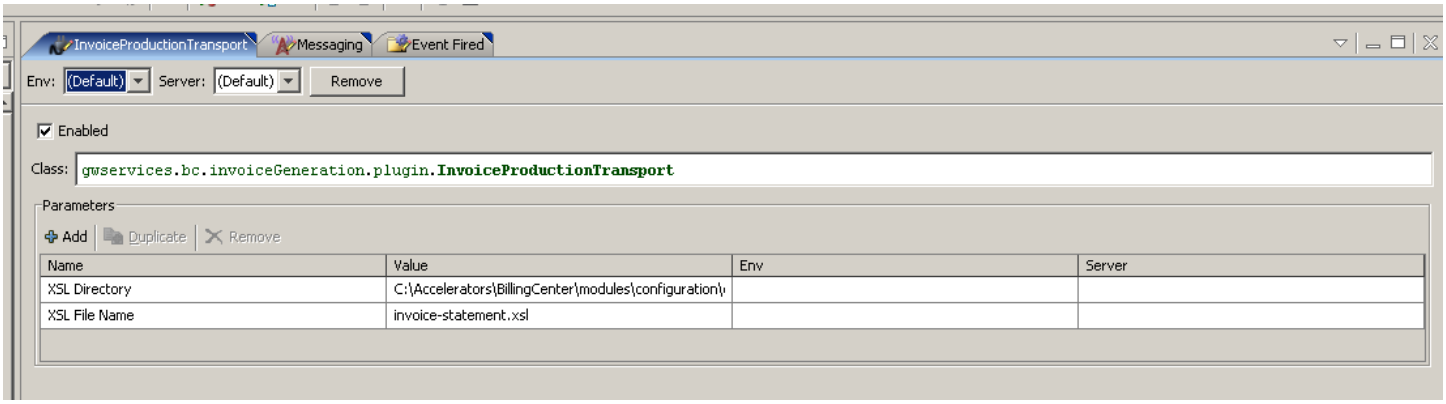
The class InvoiceProductionTransport (gwservices.bc.invoiceGeneration.plugin.InvoiceProductionTransport) is the Gosu implementation of a MessageTransport plugin. The class implements the send() method of MessageTransport interface which receives, as an argument, the message created by the Event Fired rule. The payload of the message contains the invoice data in XML format.

The transport class uses the class InvoiceProductionPDF to convert the invoice XML to a PDF statement using Apache FOP. The PDF statement is added to the account entity using the out of the box IDocumentContentSource plugin implementation. The invoice document is stored in the document path specified in the parameter for IDocumentContentSource plugin (see screenshot below). The parameter documents.path can be modified to the desired location path for document storage.



The InvoiceProductionTransport plugin class is registered in Studio with two initialization parameters:

- XSL Directory: The directory where XSL-FO format file used by Apache FOP to convert invoice XML to PDF statement is stored
- XSL File Name: The name of the XSL-FO format file



The class InvoiceProductionTransport has the following methods –

- function send(msg: Message, modifiedMsg: String) : void

  This is the overridden method of the MessageTransport interface. This method receives, as an argument, the message created in the Event Fired rule. The invoice data from the message payload is parsed and the invoice XML is converted to a PDF statement using the convertXML2PDF() method of the InvoiceProductionPDF class. The method returns an OutputStream (java.io.OutputStream) object containing the PDF statement. The OutputStream is converted to InputStream (java.io.InputStream) and then added to the Account by calling the method addInvoiceDocument (). The message is acknowledged if the document was successfully added to the Account; otherwise the message is marked with non-retryable error.

- addInvoiceDocument (inv: AccountInvoice, pdfInStream: InputStream) : void

  This method creates a new Document entity with the following data –
  Name = “<Invoice Number>.pdf”
  MimeType = "application/pdf"
  Author = "Reference Implementation"
  DocumentIdentifier = <Invoice Number>
  Description = “Invoice - <Invoice Number>, Statement Date:<Invoice Date>”
  Type = DocumentType.TC_OTHER
  SecurityType = DocumentSecurityType.TC_UNRESTRICTED
  Status = DocumentStatusType.TC_APPROVED
  The document created is added to the Account. The PDF invoice statement content is added to the Document by calling the method addDocumentContent ().

- addDocumentContent (pdfInStream: InputStream, aDocument:Document) : void

  This method adds the invoice PDF content to the new Document created for the Account.

InvoiceProductionPDF

This class InvoiceProductionPDF (gwservices.bc.invoiceGeneration.utility.InvoiceProductionPDF) is used by the InvoiceProductionTransport to convert XML into PDF using Apache FOP. The method convertXML2PDF() takes input XML and an XSL-FO file

format for FOP and returns a java.io.ByteArrayOutputStream object containing the converted PDF. Apache FOP uses the XSL-FO format XSL to convert the input invoice XML to a PDF statement.

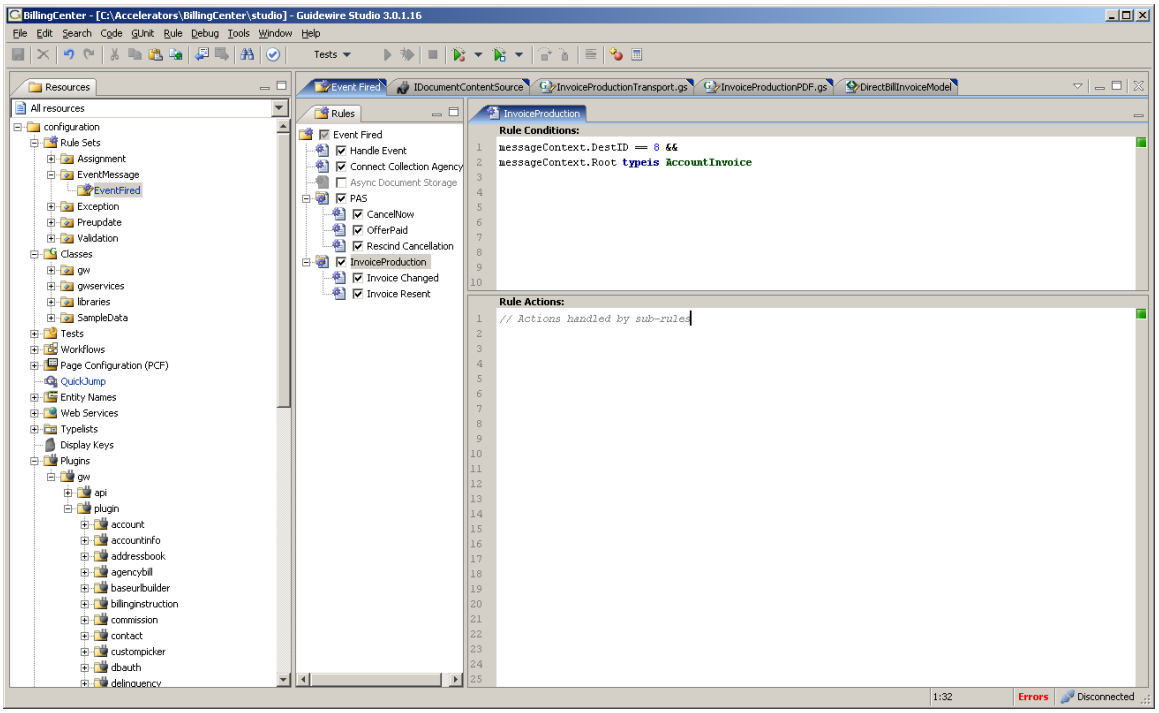
The class has the following method –

- convertXML2PDF(xmlIn: Reader, xslt: File): ByteArrayOutputStream

  This method takes in XML representing the invoice and converts it to a PDF invoice statement using Apache FOP. The XSL-FO format file is used by Apache FOP for the conversion.

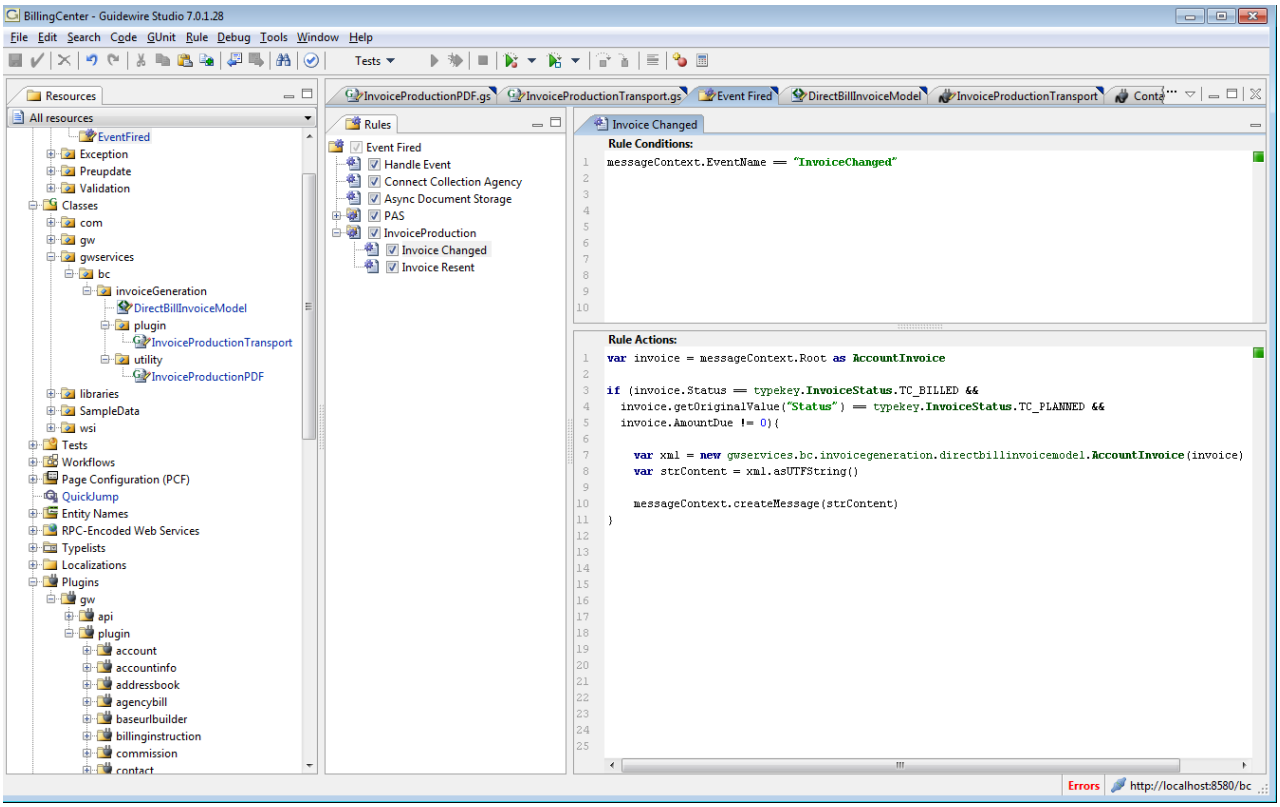
Event Rule

An Event Fired rule is defined for Invoice Production with two sub rules 'Invoice Changed' and 'Invoice Resent' that are triggered for the InvoiceChanged and InvoiceResent events respectively. The actions for the rules are executed when the destination is InvoiceProduction and the changed entity is AccountInvoice.

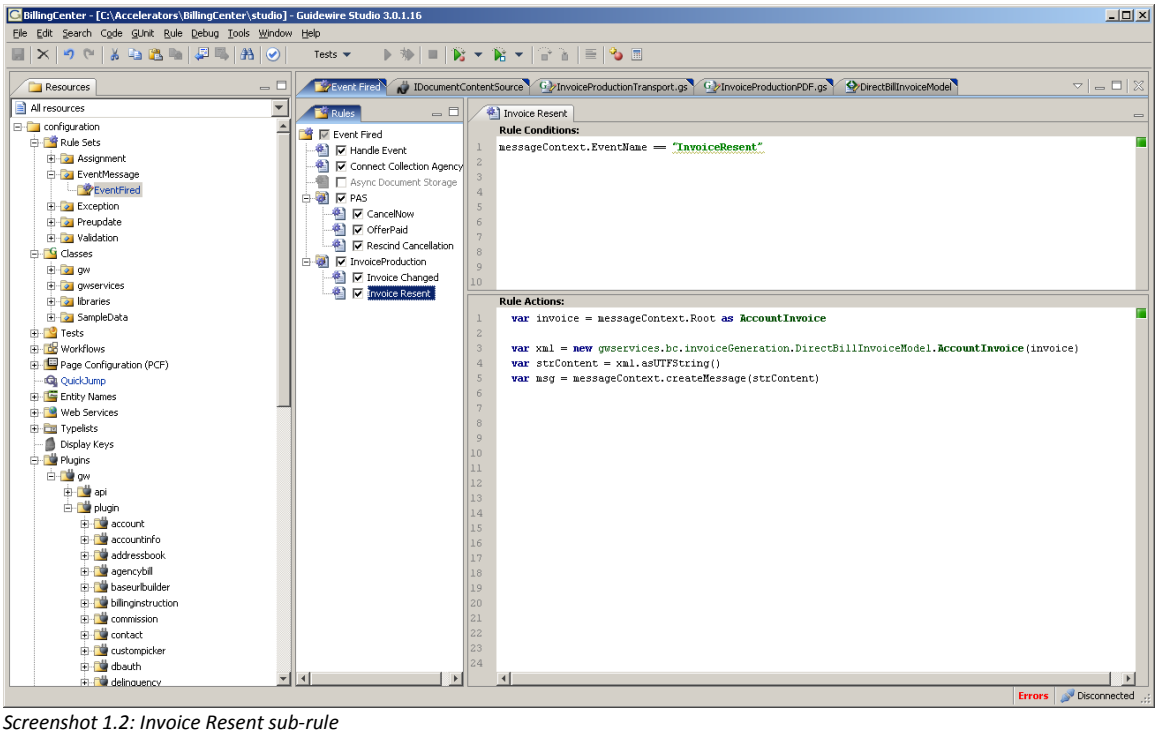


Screenshot 1: Invoice Production Rule

The rules use Guidewire XML model to create invoice data in XML. The rules also create new message containing the invoice XML in its payload. In case of Invoice Changed sub rule, the rule actions are executed only when the invoice status is changed from 'Planned' to 'Billed' and invoice paid status is not fully paid.



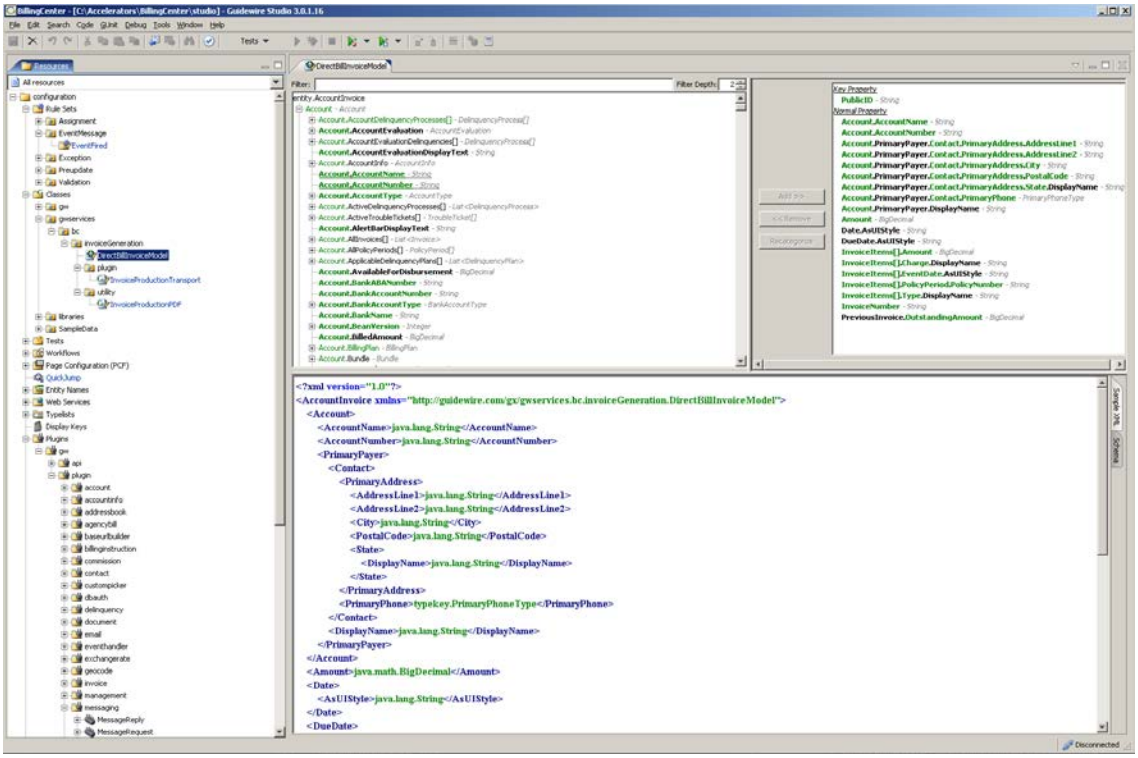
Screenshot 1.1: Invoice Changed sub-rule



Screenshot 1.2: Invoice Resent sub-rule

Required Templates and Resource Files

The XML Model (gwservices.bc.invoiceGeneration.**DirectBillInvoiceModel** under Classes) is used to create an XML model of the AccountInvoice entity that contains the required invoice data. The model can be edited in Studio to include additional data or remove data from the invoice XML if required.



The following file is the logo picture that is inserted into the generated invoice statement PDF file:
<BC Local Install Path>\modules\configuration\config\resources\xsl\brand_logo_big.jpg

The following file is passed to Apache FOP in the InvoiceProductionTransport plugin to transform the XML data into a PDF file:
<BC Local Install Path>\modules\configuration\config\resources\xsl\invoice-statement.xsl

Deployment

BillingCenter configuration files are included as part of the archive for this reference implementation.
The subfolder structure under this configuration is from a BC 7.0 install. The configuration directory contains the modified resources listed below. Refer to the 'Running this Reference Implementation' section below for step-by-step instructions for deplying this reference implementation. You will need to copy or merge these files into your deployment if you do not wish to overlay the entire configuration directory. Note that as usual, you will need to update config.xml and logging.properties with your deployment specific parameters. Also in order to test the reference implementation during development or QA, enable the Testing Clock plugin (refer to BillingCenter Guide for details on how to change the system clock for testing).

The reference implementation archive contains the following:

1. configuration – the configuration module of *Center
2. docs – contains this design document

BillingCenter

config

- messaging
  - **messaging-config.xml**– messaging registry with new InvoiceProduction destination defined
- plugin
  - registry
    - **InvoiceProductionTransport.xml** – plugin registry for InvoiceProductionTransport.
- resources
  - xsl
    - **invoice-statement.xsl** – XSL-FO file format used by Apache FOP to convert XML containing invoice details to Invoice PDF document
    - **brand_logo_big.jpg** –logo picture inserted into the invoice statement PDF file generated
- rules
  - rules
    - EventMessage
      - **EventFired.grs –** Define Event Fired rules
      - EventFired_dir
        - **InvoiceProduction.gr**– Event rule for Invoice Production
        - InvoiceProduction_dir
          - **InvoiceChanged.gr** – Sub rule for InvoiceChanged event
          - **InvoiceResent.gr** - Sub rule for InvoiceResent event

gsrc

- gwservices
  - bc
    - invoiceGeneration
      - plugin
        - **InvoiceProductionTransport.gs** – Gosu implementation class for MessageTransport plugin
      - utility
        - **InvoiceProductionPDF.gs** – Gosu class used by MessageTransport plugin to convert invoice XML to a PDF document using Apache FOP
    - **DirectBillInvoiceModel.gx** – Guidewire XML Model for the AccountInvoice entity used to generate XML containing invoice data

Running this Reference Implementation

The following are step by step instructions for deploying the reference implementation:

1. Unzip the reference implementation archive onto your local machine.
2. It is assumed that the local BillingCenter application is installed at C:\BillingCenter. If this is not the case, replace "C:\BillingCenter" in the steps below to the actual path of your local BC install.
3. Destination Set up: Copy or Merge the file configuration\config\messaging\messaging-config.xml from reference implementation to C:\BillingCenter\modules\configuration\config\messaging\messaging-config.xml. The file contains the entry for new destination defined for InvoiceProduction.
4. Plugin Configurations

- Copy the plugin registry file configuration configuration\config\plugin\registry\InvoiceProductionTransport.xml to C:\BillingCenter\modules\configuration\config\plugin\registry\InvoiceProductionTransport.xml
- Copy the directories and files under configuration\config\resources directory to C:\BillingCenter\modules\configuration\config\resources. Open the invoice-statement.xsl file under resources\xsl directory and change the path of the brand_logo_big.jpg to the absolute path where the logo image exists.
- Copy or merge the files and directories under configuration\gsrc directory to C:\BillingCenter\modules\configuration\gsrc. The folder contains the plugin classes and the XML Model file DirectBillInvoiceModel.xsd

5. Rules Configurations
   - Add Event Fired rules for Invoice Production. Copy or merge the directories and files under configuration\config\rules\rules\EventMessage to C:\BillingCenter\modules\configuration\config\rules\rules\EventMessage
6. Changes through Studio:
   - Modify the plugin parameters for InvoiceProductionTransport to the appropriate values:
     Name: XSL Directory
     Value: <dir containing XSL-FO format file>

     Name: XSL File Name
     Value: <Name of the XSL-FO file, invoice-statement.xsl>
7. Enable the ITesting Clock Plugin to modify the System Clock for testing. Refer to the Configuration Guide for enabling.
   Note: This should not be done for production servers
8. Start BillingCenter.