

Oracle® Payments

Implementation Guide

Release 12.2

Part No. E48768-05

February 2020

Oracle Payments Implementation Guide, Release 12.2

Part No. E48768-05

Copyright © 2000, 2020, Oracle and/or its affiliates.

Primary Author: Sudha Seshadri

Contributing Author: Jonathan Leybovich, Aalok Shah

Contributor: Victoria Anderson, Ramasubramanian Balasundaram, Cynthia Bilbie, Lynn Kwan, Jonathan Leybovich, Julianna Litwin, Robert MacIsaac, Sanjay Mall, Muhannad Obeidat, Padma Rao, Yidner Salazar, Lauren Scott, Ramesh. R. Shankar, Omar Tahboub, Feiyun Zhang, Sanotsh Vaze, Kondaiah Mandadi, Mitesh Kumbhat

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc>.

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info> or visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs> if you are hearing impaired.

Contents

Send Us Your Comments

Preface

1 Overview

Introduction to Oracle Payments	1-1
General Features	1-2
Funds Disbursement Features	1-4
Funds Capture Features	1-7

2 Planning Your Implementation

Shared Planning	2-1
What are Your Security Needs?.....	2-1
Do You Want to Use Funds Capture or Funds Disbursement Functionality?.....	2-2
Which Payment System do You Want to Use?.....	2-2
What Source Products are You Using?.....	2-3
What Formats do You Need to Support?.....	2-4
Does Your Application Need to Present Information in Different Languages?.....	2-4
Funds Capture Planning	2-6
How do You Need to Organize Your Settlement Batches?.....	2-6
What Credit Card Brands do You Want to Support?.....	2-6
Which APIs do You Need to Use for Funds Capture?.....	2-7
Which Bank Account Transfer Operations do You Want to Implement for Funds Capture?	2-8
Which Risk Factors do You Want to Implement?.....	2-8
Funds Disbursement Planning	2-9
What Disbursement Payment Methods do You Want to Support?.....	2-9

3 Understanding Oracle Payments

Functionality Common to Both Funds Capture and Funds Disbursement	3-1
Understanding Access Control.....	3-1
Understanding Oracle Payments APIs.....	3-3
Understanding Oracle Payments Servlets.....	3-4
Understanding Transmission.....	3-4
Understanding Oracle Payments Security.....	3-5
Understanding Funds Capture	3-8
Understanding Funds Capture Process Profile.....	3-8
Understanding Payment Methods (Funds Capture).....	3-8
Understanding Payees.....	3-9
Integration with Other Oracle Applications.....	3-9
Understanding Credit Card Transactions.....	3-11
Understanding Purchase Cards.....	3-12
Understanding PINless Debit Card Transactions.....	3-14
Understanding Funds Capture Bank Account Transfers.....	3-15
Understanding Gateway-Model and Processor-Model Payment Systems.....	3-16
Understanding Terminal-Based and Host-Based Merchants.....	3-17
Understanding Offline and Online Payments.....	3-17
Understanding Risk Management.....	3-18
Risk Factors Shipped with Oracle Payments.....	3-19
Oracle Payments Routing and Operation.....	3-21
Routing Rule Conditions.....	3-22
Understanding Transaction Reporting.....	3-24
Understanding Funds Disbursement	3-26
Understanding Documents Payable.....	3-26
Understanding Payments.....	3-26
Understanding Payment Methods (Funds Disbursement).....	3-27
Understanding Payment Process Profiles.....	3-27
Understanding Payment Process Requests.....	3-28
Understanding Payment Grouping.....	3-29
Understanding Payment Instructions.....	3-29
Understanding Validations.....	3-30

4 Shared Setup Tasks for Funds Capture and Funds Disbursement

Overview	4-1
Step 1. Creating Oracle Payments Users	4-5
Step 2. Setting Up System Security Options	4-7
Step 3. Setting Up Oracle XML Publisher Templates	4-9

Step 4. Setting Up Formats.....	4-10
Step 5. Setting Up Validations.....	4-10
Step 6. Setting Up Transmission Configurations.....	4-11
Step 7. Configuring Tunneling.....	4-11
Step 8. Setting Up Payment Systems.....	4-13
Step 9. Configuring Oracle Payments Sample Servlet.....	4-15
Step 10. Setting Up SSL Security for Payment System Servlet Communication.....	4-17
Step 11. Configuring the XML Framework.....	4-18

5 Setup Tasks for Funds Disbursement

Overview.....	5-1
Step 12. Setting Up Funds Disbursement Payment Methods.....	5-2
Step 13. Setting Up Payment Method Defaulting Rules.....	5-6
Step 14. Setting Up Bank Instruction Codes.....	5-7
Step 15. Setting Up Delivery Channel Codes.....	5-8
Step 16. Setting Up Payment Reason Codes.....	5-9
Step 17. Setting Up Payment Process Profiles.....	5-9
Step 18. Setting Up Disbursement System Options.....	5-21

6 Setup Tasks for Funds Capture

Overview.....	6-1
Step 19. Configuring the ECApp Servlet.....	6-2
Step 20. Setting Up Funds Capture Payment Methods.....	6-3
Step 21. Setting Up Funds Capture Process Profiles.....	6-4
Step 22. Setting Up First Party Payees.....	6-9
Step 23. Setting Up Credit Card Brands.....	6-12
Step 24. Loading Risky Instruments.....	6-12

7 Transaction Testing

Testing Transactions.....	7-1
---------------------------	-----

8 Using Oracle Payments with External Payment Systems

Overview of Payment System Integration.....	8-1
Payment System Integration Components.....	8-1
Developing a Custom Payment System Integration for Funds Capture.....	8-3
Developing a Custom Payment System Integration for Funds Disbursement.....	8-12
Defining a Payment System.....	8-16
Account Options.....	8-17
Formats.....	8-18

Extract Generator.....	8-19
Extract Formatter.....	8-19
Transmission Functions.....	8-19
Acknowledgment Parser.....	8-23
Creating Custom Disbursement Validation Sets.....	8-32
Creating Custom Funds Capture Validation Sets.....	8-46

A Payment Formats

Viewing Payment Formats.....	A-1
Common Disbursement Payment Formats.....	A-1
Argentina.....	A-4
Belgium.....	A-5
Brazil.....	A-7
Chile.....	A-8
Colombia.....	A-9
Finland.....	A-11
France.....	A-13
Germany.....	A-16
Italy.....	A-22
Japan.....	A-24
Netherlands.....	A-25
New Zealand.....	A-26
Norway.....	A-27
Poland.....	A-28
Portugal.....	A-30
Spain.....	A-32
Sweden.....	A-37
Switzerland.....	A-41
UK.....	A-43
US.....	A-43
US Federal.....	A-47

B Validations

Other Validations.....	B-1
Country-Specific Validations.....	B-3

C Transmission Protocol Parameters

Transmission Protocol Parameters.....	C-1
---------------------------------------	-----

D Risk Management

Using Risk Management.....	D-1
Risk Management Test Scenarios.....	D-3

E Funds Capture Error Codes

Error Handling During Payment Processing.....	E-1
---	-----

F Integrating with Oracle Payments Using Public Funds Capture APIs

Overview of Oracle Payments APIs.....	F-1
Implementing Electronic Commerce Applications APIs.....	F-1
Payment Instrument Registration APIs.....	F-2
Payment Processing APIs.....	F-3
Risk Management APIs.....	F-6
Credit Card Validation APIs.....	F-7
Status Update API.....	F-9
Java APIs for Electronic Commerce Application.....	F-12
PL/SQL APIs for Electronic Commerce Applications.....	F-21
Security Considerations.....	F-26

G Oracle Payments Back-End APIs for Gateways

Integrating with Non-Oracle Applications.....	G-1
---	-----

H Profile Options

Profile Options.....	H-1
Oracle Payments Profile Options.....	H-4

I Customizations

Search Pages.....	I-1
-------------------	-----

J Funds Capture Extract

Extract Structure.....	J-1
Extract Components.....	J-2
Funds Capture Extract.....	J-2
Common Elements.....	J-21

K Funds Disbursement Extract

Extract Structure Overview.....	K-1
---------------------------------	-----

Payment Instruction Level Elements.....	K-3
Payment Level Elements.....	K-13
Document Payable Level Elements.....	K-33
Document Line Level Elements.....	K-40
Common Elements.....	K-42

Index

Send Us Your Comments

Oracle Payments Implementation Guide, Release 12.2

Part No. E48768-05

Oracle welcomes customers' comments and suggestions on the quality and usefulness of this document. Your feedback is important, and helps us to best meet your needs as a user of our products. For example:

- Are the implementation steps correct and complete?
- Did you understand the context of the procedures?
- Did you find any errors in the information?
- Does the structure of the information help you with your tasks?
- Do you need different information or graphics? If so, where, and in what format?
- Are the examples correct? Do you need more examples?

If you find any errors or have any other suggestions for improvement, then please tell us your name, the name of the company who has licensed our products, the title and part number of the documentation and the chapter, section, and page number (if available).

Note: Before sending us your comments, you might like to check that you have the latest version of the document and if any concerns are already addressed. To do this, access the new Oracle E-Business Suite Release Online Documentation CD available on My Oracle Support and www.oracle.com. It contains the most current Documentation Library plus all documents revised or released recently.

Send your comments to us using the electronic mail address: appsdoc_us@oracle.com

Please give your name, address, electronic mail address, and telephone number (optional).

If you need assistance with Oracle software, then please contact your support representative or Oracle Support Services.

If you require training or instruction in using Oracle software, then please contact your Oracle local office and inquire about our Oracle University offerings. A list of Oracle offices is available on our Web site at www.oracle.com.

Preface

Intended Audience

Welcome to Release 12.2 of the *Oracle Payments Implementation Guide*.

This guide assumes you have a working knowledge of the following:

- The principles and customary practices of your business area.
- Computer desktop application usage and terminology.

If you have never used Oracle E-Business Suite, we suggest you attend one or more of the Oracle E-Business Suite training classes available through Oracle University.

See Related Information Sources on page xii for more Oracle E-Business Suite product information.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc>.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info> or visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs> if you are hearing impaired.

Structure

- 1 Overview
- 2 Planning Your Implementation

3	Understanding Oracle Payments
4	Shared Setup Tasks for Funds Capture and Funds Disbursement
5	Setup Tasks for Funds Disbursement
6	Setup Tasks for Funds Capture
7	Transaction Testing
8	Using Oracle Payments with External Payment Systems
A	Payment Formats
B	Validations
C	Transmission Protocol Parameters
D	Risk Management
E	Funds Capture Error Codes
F	Integrating with Oracle Payments Using Public Funds Capture APIs
G	Oracle Payments Back-End APIs for Gateways
H	Profile Options
I	Customizations
J	Funds Capture Extract
K	Funds Disbursement Extract

Related Information Sources

This book is included in the Oracle E-Business Suite Documentation Library, which is supplied in the Release 12.2 Media Pack. If this guide refers you to other Oracle E-Business Suite documentation, use only the latest Release 12.2 versions of those guides.

Online Documentation

All Oracle E-Business Suite documentation is available online (HTML or PDF).

- **Online Help** - Online help patches (HTML) are available on My Oracle Support.
- **PDF Documentation** - See the Oracle E-Business Suite Documentation Library for current PDF documentation for your product with each release.
- **Release Notes** - For information about changes in this release, including new features, known issues, and other details, see the release notes for the relevant product, available on My Oracle Support.
- **Oracle Electronic Technical Reference Manual** - The Oracle Electronic Technical Reference Manual (eTRM) contains database diagrams and a detailed description of database tables, forms, reports, and programs for each Oracle E-Business Suite product. This information helps you convert data from your existing applications and integrate Oracle E-Business Suite data with non-Oracle applications, and write custom reports for Oracle E-Business Suite products. The Oracle eTRM is available on My Oracle Support.

Related Guides

You should have the following related books on hand. Depending on the requirements of your particular installation, you may also need additional manuals or guides.

Oracle Alert User's Guide

This guide explains how to define periodic and event alerts to monitor the status of your Oracle E-Business Suite data.

Oracle Application Framework Developer's Guide

This guide contains the coding standards followed by the Oracle E-Business Suite development staff to produce applications built with Oracle Application Framework. This guide is available in PDF format on My Oracle Support and as online documentation in JDeveloper 10g with Oracle Application Extension.

Oracle Application Framework Personalization Guide

This guide covers the design-time and run-time aspects of personalizing applications built with Oracle Application Framework.

Oracle Fusion Middleware Adapter for Oracle Applications User's Guide (Oracle Application Server Adapter for Oracle Applications User's Guide)

This guide covers the use of Adapter for Oracle Applications in developing integrations between Oracle E-Business Suite and trading partners.

Please note that the user's guide can be found in the following documentation libraries:

- As part of the Oracle Fusion Middleware and SOA Suite in 11g, *Oracle Fusion Middleware Adapter for Oracle Applications User's Guide* is available in the Oracle Fusion Middleware 11g Documentation Library.
- As part of the Oracle Application Server in 10g, *Oracle Application Server Adapter for Oracle Applications User's Guide* is available in the Oracle Application Server 10g Documentation Library.

Oracle Diagnostics Framework User's Guide

This manual contains information on implementing and administering diagnostics tests for Oracle E-Business Suite using the Oracle Diagnostics Framework.

Oracle E-Business Suite Concepts

This book is intended for all those planning to deploy Oracle E-Business Suite Release 12.2, or contemplating significant changes to a configuration. After describing the Oracle E-Business Suite architecture and technology stack, it focuses on strategic topics, giving a broad outline of the actions needed to achieve a particular goal, plus the installation and configuration choices that may be available.

Oracle E-Business Suite CRM System Administrator's Guide

This manual describes how to implement the CRM Technology Foundation (JTT) and use its System Administrator Console.

Oracle E-Business Suite Desktop Integration Framework Developer's Guide

Oracle E-Business Suite Desktop Integration Framework is a development tool that lets you define custom integrators for use with Oracle Web Applications Desktop Integrator. This guide describes how to define and manage integrators and all associated supporting objects, as well as how to download and upload integrator

definitions.

Oracle E-Business Suite Developer's Guide

This guide contains the coding standards followed by the Oracle E-Business Suite development staff. It describes the Oracle Application Object Library components needed to implement the Oracle E-Business Suite user interface described in the *Oracle E-Business Suite User Interface Standards for Forms-Based Products*. It provides information to help you build your custom Oracle Forms Developer forms so that they integrate with Oracle E-Business Suite. In addition, this guide has information for customizations in features such as concurrent programs, flexfields, messages, and logging.

Oracle E-Business Suite Flexfields Guide

This guide provides flexfields planning, setup, and reference information for the Oracle E-Business Suite implementation team, as well as for users responsible for the ongoing maintenance of Oracle E-Business Suite product data. This guide also provides information on creating custom reports on flexfields data.

Oracle E-Business Suite Installation Guide: Using Rapid Install

This book is intended for use by anyone who is responsible for installing or upgrading Oracle E-Business Suite. It provides instructions for running Rapid Install either to carry out a fresh installation of Oracle E-Business Suite Release 12.2, or as part of an upgrade to Release 12.2.

Oracle E-Business Suite Maintenance Guide

This guide contains information about the strategies, tasks, and troubleshooting activities that can be used to help ensure an Oracle E-Business Suite system keeps running smoothly, together with a comprehensive description of the relevant tools and utilities. It also describes how to patch a system, with recommendations for optimizing typical patching operations and reducing downtime.

Oracle E-Business Suite Security Guide

This guide contains information on a comprehensive range of security-related topics, including access control, user management, function security, data security, and auditing. It also describes how Oracle E-Business Suite can be integrated into a single sign-on environment.

Oracle E-Business Suite Setup Guide

This guide contains information on system configuration tasks that are carried out either after installation or whenever there is a significant change to the system. The activities described include defining concurrent programs and managers, enabling Oracle Applications Manager features, and setting up printers and online help.

Oracle E-Business Suite User's Guide

This guide explains how to navigate, enter data, query, and run reports using the user interface (UI) of Oracle E-Business Suite. This guide also includes information on setting user profiles, as well as running and reviewing concurrent requests.

Oracle E-Business Suite User Interface Standards for Forms-Based Products

This guide contains the user interface (UI) standards followed by the Oracle E-Business Suite development staff. It describes the UI for the Oracle E-Business Suite products and how to apply this UI to the design of an application built by using Oracle Forms.

Oracle E-Business Suite Integrated SOA Gateway User's Guide

This guide describes the high level service enablement process, explaining how users can browse and view the integration interface definitions and services residing in Oracle Integration Repository.

Oracle E-Business Suite Integrated SOA Gateway Implementation Guide

This guide explains how integration repository administrators can manage and administer the Web service activities for integration interfaces including native packaged integration interfaces, composite services (BPEL type), and custom integration interfaces. It also describes how to invoke Web services from Oracle E-Business Suite by employing the Oracle Workflow Business Event System, and how to manage Web service security, configure logs, and monitor SOAP messages.

Oracle E-Business Suite Integrated SOA Gateway Developer's Guide

This guide describes how system integration developers can perform end-to-end service integration activities. These include orchestrating discrete Web services into meaningful end-to-end business processes using business process execution language (BPEL), and deploying BPEL processes at run time.

This guide also explains how to invoke Web services using the Service Invocation Framework. This includes defining Web service invocation metadata, invoking Web services, and testing the Web service invocation.

Oracle e-Commerce Gateway User's Guide

This guide describes the functionality of Oracle e-Commerce Gateway and the necessary setup steps in order for Oracle E-Business Suite to conduct business with trading partners through Electronic Data Interchange (EDI). It also describes how to run extract programs for outbound transactions, import programs for inbound transactions, and the relevant reports.

Oracle e-Commerce Gateway Implementation Guide

This guide describes implementation details, highlighting additional setup steps needed for trading partners, code conversion, and Oracle E-Business Suite. It also provides architecture guidelines for transaction interface files, troubleshooting information, and a description of how to customize EDI transactions.

Oracle iSetup Developer's Guide

This manual describes how to build, test, and deploy Oracle iSetup Framework interfaces.

Oracle iSetup User's Guide

This guide describes how to use Oracle iSetup to migrate data between different instances of the Oracle E-Business Suite and generate reports. It also includes configuration information, instance mapping, and seeded templates used for data

migration.

Oracle Report Manager User's Guide

Oracle Report Manager is an online report distribution system that provides a secure and centralized location to produce and manage point-in-time reports. Oracle Report Manager users can be either report producers or report consumers. Use this guide for information on setting up and using Oracle Report Manager.

Oracle Web Applications Desktop Integrator Implementation and Administration Guide

Oracle Web Applications Desktop Integrator brings Oracle E-Business Suite functionality to a spreadsheet, where familiar data entry and modeling techniques can be used to complete Oracle E-Business Suite tasks. You can create formatted spreadsheets on your desktop that allow you to download, view, edit, and create Oracle E-Business Suite data, which you can then upload. This guide describes how to implement Oracle Web Applications Desktop Integrator and how to define mappings, layouts, style sheets, and other setup options.

Oracle Workflow Administrator's Guide

This guide explains how to complete the setup steps necessary for any product that includes workflow-enabled processes. It also describes how to manage workflow processes and business events using Oracle Applications Manager, how to monitor the progress of runtime workflow processes, and how to administer notifications sent to workflow users.

Oracle Workflow Developer's Guide

This guide explains how to define new workflow business processes and customize existing Oracle E-Business Suite-embedded workflow processes. It also describes how to define and customize business events and event subscriptions.

Oracle Workflow User's Guide

This guide describes how users can view and respond to workflow notifications and monitor the progress of their workflow processes.

Oracle Workflow API Reference

This guide describes the APIs provided for developers and administrators to access Oracle Workflow.

Oracle Workflow Client Installation Guide

This guide describes how to install the Oracle Workflow Builder and Oracle XML Gateway Message Designer client components for Oracle E-Business Suite.

Oracle XML Gateway User's Guide

This guide describes Oracle XML Gateway functionality and each component of the Oracle XML Gateway architecture, including Message Designer, Oracle XML Gateway Setup, Execution Engine, Message Queues, and Oracle Transport Agent. It also explains how to use Collaboration History that records all business transactions and messages

exchanged with trading partners.

The integrations with Oracle Workflow Business Event System, and the Business-to-Business transactions are also addressed in this guide.

Oracle XML Publisher Report Designer's Guide

Oracle XML Publisher is a template-based reporting solution that merges XML data with templates in RTF or PDF format to produce a variety of outputs to meet a variety of business needs. Using Microsoft Word or Adobe Acrobat as the design tool, you can create pixel-perfect reports from the Oracle E-Business Suite. Use this guide to design your report layouts.

This guide is available through the Oracle E-Business Suite online help.

Oracle XML Publisher Administration and Developer's Guide

Oracle XML Publisher is a template-based reporting solution that merges XML data with templates in RTF or PDF format to produce a variety of outputs to meet a variety of business needs. Outputs include: PDF, HTML, Excel, RTF, and eText (for EDI and EFT transactions). Oracle XML Publisher can be used to generate reports based on existing Oracle E-Business Suite report data, or you can use Oracle XML Publisher's data extraction engine to build your own queries. Oracle XML Publisher also provides a robust set of APIs to manage delivery of your reports via e-mail, fax, secure FTP, printer, WebDav, and more. This guide describes how to set up and administer Oracle XML Publisher as well as how to use the Application Programming Interface to build custom solutions.

This guide is available through the Oracle E-Business Suite online help.

Oracle E-Business Suite Upgrade Guide: Release 12.0 and 12.1 to 12.2:

This guide provides information for DBAs and Applications Specialists who are responsible for upgrading Release 12.0 and 12.1 Oracle E-Business Suite system (techstack and products) to Release 12.2. In addition to information about applying the upgrade driver, it outlines pre-upgrade steps and post-upgrade steps, and provides descriptions of product-specific functional changes and suggestions for verifying the upgrade and reducing downtime.

Oracle Advanced Global Intercompany System User's Guide:

This guide describes the self service application pages available for Intercompany users. It includes information on setting up intercompany, entering intercompany transactions, importing transactions from external sources and generating reports.

Oracle Advanced Collections User Guide:

This guide describes how to use the features of Oracle Advanced Collections to manage your collections activities. It describes how collections agents and managers can use Oracle Advanced Collections to identify delinquent customers, review payment history and aging data, process payments, use strategies and dunning plans to automate the collections process, manage work assignments, and handle later-stage delinquencies.

Oracle Advanced Collections Implementation Guide:

This guide describes how to configure Oracle Advanced Collections and its integrated products. It contains the steps required to set up and verify your implementation of Oracle Advanced Collections.

Oracle Assets User Guide:

This guide provides you with information on how to implement and use Oracle Assets. Use this guide to understand the implementation steps required for application use, including defining depreciation books, depreciation method, and asset categories. It also contains information on setting up assets in the system, maintaining assets, retiring and reinstating assets, depreciation, group depreciation, accounting and tax accounting, budgeting, online inquiries, impairment processing, and Oracle Assets reporting. The guide explains using Oracle Assets with Multiple Reporting Currencies (MRC). This guide also includes a comprehensive list of profile options that you can set to customize application behavior.

Oracle Bill Presentment Architecture User's Guide:

This guide provides you information on using Oracle Bill Presentment Architecture. Consult this guide to create and customize billing templates, assign a template to a rule and submit print requests. This guide also provides detailed information on page references, seeded content items and template assignment attributes.

Oracle Cash Management User Guide:

This guide describes how to use Oracle Cash Management to clear your receipts, as well as reconcile bank statements with your outstanding balances and transactions. This manual also explains how to effectively manage and control your cash cycle. It provides comprehensive bank reconciliation and flexible cash forecasting.

Oracle Credit Management User Guide:

This guide provides you with information on how to use Oracle Credit Management. This guide includes implementation steps, such as how to set up credit policies, as well as details on how to use the credit review process to derive credit recommendations that comply with your credit policies. This guide also includes detailed information about the public application programming interfaces (APIs) that you can use to extend Oracle Credit Management functionality.

Oracle Customer Data Librarian User Guide:

This guide describes how to use Oracle Customer Data Librarian to establish and maintain the quality of the Trading Community Architecture Registry, focusing on consolidation, cleanliness, and completeness. Oracle Customer Data Librarian has all of the features in Oracle Customers Online, and is also part of the Oracle Customer Data Management product family.

Oracle Customer Data Librarian Implementation Guide:

This guide describes how to implement Oracle Customer Data Librarian. As part of implementing Oracle Customer Data Librarian, you must also complete all the implementation steps for Oracle Customers Online.

Oracle Customers Online User Guide:

This guide describes how to use Oracle Customers Online to view, create, and maintain your customer information. Oracle Customers Online is based on Oracle Trading Community Architecture data model and functionality, and is also part of the Oracle Customer Data Management product family.

Oracle Customers Online Implementation Guide:

This guide describes how to implement Oracle Customers Online.

Oracle E-Business Suite Multiple Organizations Implementation Guide:

This guide describes the multiple organizations concepts in Oracle E-Business Suite. It describes in detail on setting up and working effectively with multiple organizations in Oracle E-Business Suite.

Oracle E-Business Tax User Guide:

This guide describes the entire process of setting up and maintaining tax configuration data, as well as applying tax data to the transaction line. It describes the entire regime-to-rate setup flow of tax regimes, taxes, statuses, rates, recovery rates, tax jurisdictions, and tax rules. It also describes setting up and maintaining tax reporting codes, fiscal classifications, tax profiles, tax registrations, configuration options, and third party service provider subscriptions. You also use this manual to maintain migrated tax data for use with E-Business Tax.

Oracle E-Business Tax Implementation Guide:

This guide provides a conceptual overview of the E-Business Tax tax engine, and describes the prerequisite implementation steps to complete in other applications in order to set up and use E-Business Tax. The guide also includes extensive examples of setting up country-specific tax requirements.

Oracle E-Business Tax Reporting Guide:

This guide explains how to run all tax reports that make use of the E-Business Tax data extract. This includes the Tax Reporting Ledger and other core tax reports, country-specific VAT reports, and Latin Tax Engine reports.

Oracle E-Business Tax: Vertex Q-Series and Taxware Sales/Use Tax System Implementation Guide

This guide explains how to setup and use the services of third party tax service providers for US Sales and Use tax. The tax service providers are Vertex Q-Series and Taxware Sales/Use Tax System. When implemented, the Oracle E-Business Tax service subscription calls one of these tax service providers to return a tax rate or amount whenever US Sales and Use tax is calculated by the Oracle E-Business Tax tax engine. This guide provides setup steps, information about day-to-day business processes, and a technical reference section.

Oracle Embedded Data Warehouse User Guide:

This guide describes how to use Embedded Data Warehouse reports and workbooks to analyze performance.

Oracle Embedded Data Warehouse Implementation Guide:

This guide describes how to implement Embedded Data Warehouse, including how to set up the intelligence areas.

Oracle Embedded Data Warehouse Install Guide:

This guide describes how to install Embedded Data Warehouse, including how to create database links and create the end user layer (EUL).

Oracle Financial Accounting Hub Implementation Guide:

This guide provides detailed implementation information that leverages the features of Oracle Subledger Accounting to generate accounting.

Oracle Financial Services Reference Guide:

This guide provides reference material for Oracle Financial Services applications in Release 12, such as Oracle Transfer Pricing, and includes technical details about application use as well as general concepts, equations, and calculations.

Oracle Financial Services Implementation Guide:

This guide describes how to set up Oracle Financial Services applications in Release 12.

Oracle Financial Services Reporting Administration Guide:

This guide describes the reporting architecture of Oracle Financial Services applications in Release 12, and provides information on how to view these reports.

Oracle Financials and Oracle Procurement Functional Upgrade Guide: Release 11i to Release 12:

This guides provides detailed information about the functional impacts of upgrading Oracle Financials and Oracle Procurement products from Release 11i to Release 12. This guide supplements the *Oracle E-Business Suite Upgrade Guide: Release 12.0 and 12.1 to 12.2* .

Oracle Financials Concepts Guide:

This guide describes the fundamental concepts of Oracle Financials. The guide is intended to introduce readers to the concepts used in the applications, and help them compare their real world business, organization, and processes to those used in the applications.

Oracle Financials Country-Specific Installation Supplement:

This guide provides general country information, such as responsibilities and report security groups, as well as any post-install steps required by some countries.

Oracle Financials for the Americas User Guide:

This guide describes functionality developed to meet specific business practices in countries belonging to the Americas region. Consult this user guide along with your financial product user guides to effectively use Oracle Financials in your country.

Oracle Financials for Asia/Pacific User Guide:

This guide describes functionality developed to meet specific business practices in countries belonging to the Asia/Pacific region. Consult this user guide along with your financial product user guides to effectively use Oracle Financials in your country.

Oracle Financials for Europe User Guide:

This guide describes functionality developed to meet specific business practices in countries belonging to the European region. Consult this user guide along with your financial product user guides to effectively use Oracle Financials in your country.

Oracle Financials for India User's Guide:

This guide provides information on how to use Oracle Financials for India. Use this guide to learn how to create and maintain setup related to India taxes, defaulting and calculation of taxes on transactions. This guide also includes information about accounting and reporting of taxes related to India.

Oracle Financials for India Implementation Guide:

This guide provides information on how to implement Oracle Financials for India. Use this guide to understand the implementation steps required for application use, including how to set up taxes, tax defaulting hierarchies, set up different tax regimes, organization and transactions.

Oracle Financials Glossary:

The glossary includes definitions of common terms that are shared by all Oracle Financials products. In some cases, there may be different definitions of the same term for different Financials products. If you are unsure of the meaning of a term you see in an Oracle Financials guide, please refer to the glossary for clarification. You can find the glossary in the online help or in the *Oracle Financials Implementation Guide*.

Oracle Financials Implementation Guide:

This guide provides information on how to implement the Oracle Financials E-Business Suite. It guides you through setting up your organizations, including legal entities, and their accounting, using the Accounting Setup Manager. It covers intercompany accounting and sequencing of accounting entries, and it provides examples.

Oracle Financials RXi Reports Administration Tool User Guide:

This guide describes how to use the RXi reports administration tool to design the content and layout of RXi reports. RXi reports let you order, edit, and present report information to better meet your company's reporting needs.

Oracle General Ledger Implementation Guide:

This guide provides information on how to implement Oracle General Ledger. Use this guide to understand the implementation steps required for application use, including how to set up Accounting Flexfields, Accounts, and Calendars.

Oracle General Ledger Reference Guide

This guide provides detailed information about setting up General Ledger Profile Options and Applications Desktop Integrator (ADI) Profile Options.

Oracle General Ledger User's Guide:

This guide provides information on how to use Oracle General Ledger. Use this guide to learn how to create and maintain ledgers, ledger currencies, budgets, and journal entries. This guide also includes information about running financial reports.

Oracle Incentive Compensation Implementation Guide:

This guide provides Compensation Administrators with guidance during implementation of Oracle Incentive Compensation. The procedures are presented in the recommended order that they should be performed for successful implementation. Appendixes are included that describe system profiles, lookups, and other useful information.

Oracle Incentive Compensation User Guide:

This guide helps Compensation Managers, Compensation Analysts, and Plan administrators to manage Oracle Incentive Compensation on a day-to-day basis. Learn how to create and manage rules hierarchies, create compensation plans, collect transactions, calculate and pay commission, and use Sales Credit Allocation.

Oracle Internet Expenses Implementation and Administration Guide:

This book explains in detail how to configure Oracle Internet Expenses and describes its integration with other applications in the E-Business Suite, such as Oracle Payables and Oracle Projects. Use this guide to understand the implementation steps required for application use, including how to set up policy and rate schedules, credit card policies, audit automation, and the expenses spreadsheet. This guide also includes detailed information about the client extensions that you can use to extend Oracle Internet Expenses functionality.

Oracle iAssets User Guide

This guide provides information on how to implement and use Oracle iAssets. Use this guide to understand the implementation steps required for application use, including setting up Oracle iAssets rules and related product setup steps. It explains how to define approval rules to facilitate the approval process. It also includes information on using the Oracle iAssets user interface to search for assets, create self-service transfer requests and view notifications.

Oracle iProcurement Implementation and Administration Guide:

This manual describes how to set up and administer Oracle iProcurement. Oracle iProcurement enables employees to requisition items through a self-service, Web interface.

Oracle iReceivables Implementation Guide:

This guide provides information on how to implement Oracle iReceivables. Use this guide to understand the implementation steps required for application use, including how to set up and configure iReceivables, and how to set up the Credit Memo Request workflow. There is also a chapter that provides an overview of major features available in iReceivables.

Oracle iSupplier Portal User Guide:

This guide contains information on how to use Oracle iSupplier Portal to enable secure transactions between buyers and suppliers using the Internet. Using Oracle iSupplier Portal, suppliers can monitor and respond to events in the procure-to-pay cycle.

Oracle iSupplier Portal Implementation Guide:

This guide contains information on how to implement Oracle iSupplier Portal and enable secure transactions between buyers and suppliers using the Internet.

Oracle Loans User Guide:

This guide describes how to set up and use Oracle Loans. It includes information on how to create, approve, fund, amortize, bill, and service extended repayment plan and direct loans.

Oracle Partner Management Implementation and Administration Guide:

This guide helps Vendor administrators to set up and maintain relationships and programs in the Partner Management application. The main areas include setting up the partner and channel manager dashboards, partner setup, partner programs and enrollment, opportunity and referral management, deal registration, special pricing management, and partner fund management.

Oracle Partner Management Vendor User Guide:

This guide assists vendor users in using Partner Management on a daily basis. This includes interaction with the partner and channel manager dashboards, working with partners and partner programs, managing opportunities and referrals, registering deals, and working with special pricing and partner funds.

Oracle Payables User's Guide:

This guide describes how to use Oracle Payables to create invoices and make payments. In addition, it describes how to enter and manage suppliers, import invoices using the Payables open interface, manage purchase order and receipt matching, apply holds to invoices, and validate invoices. It contains information on managing expense reporting, procurement cards, and credit cards. This guide also explains the accounting for Payables transactions.

Oracle Payables Implementation Guide:

This guide provides you with information on how to implement Oracle Payables. Use this guide to understand the implementation steps required for how to set up suppliers, payments, accounting, and tax.

Oracle Payables Reference Guide:

This guide provides you with detailed information about the Oracle Payables open interfaces, such as the Invoice open interface, which lets you import invoices. It also includes reference information on purchase order matching and purging purchasing information.

Oracle Payments Implementation Guide:

This guide describes how Oracle Payments, as the central payment engine for the Oracle E-Business Suite, processes transactions, such as invoice payments from Oracle Payables, bank account transfers from Oracle Cash Management, and settlements against credit cards and bank accounts from Oracle Receivables. This guide also describes how Oracle Payments is integrated with financial institutions and payment systems for receipt and payment processing, known as funds capture and funds disbursement, respectively. Additionally, the guide explains to the implementer how to plan the implementation of Oracle Payments, how to configure it, set it up, test transactions, and how use it with external payment systems.

Oracle Payments User's Guide:

This guide describes how Oracle Payments, as the central payment engine for the Oracle E-Business Suite, processes transactions, such as invoice payments from Oracle Payables, bank account transfers from Oracle Cash Management, and settlements against credit cards and bank accounts from Oracle Receivables. This guide also describes to the Payment Administrator how to monitor the funds capture and funds disbursement processes, as well as how to remedy any errors that may arise.

Oracle Procurement Buyer's Guide to Punchout and Transparent Punchout:

This guide contains necessary information for customers implementing remote catalog content on a supplier's Web site or on Oracle Exchange.

Oracle Procurement Contracts Online Help:

This guide is provided as online help only from the Oracle Procurement Contracts application and includes information about creating and managing your contract terms library.

Oracle Procurement Contracts Implementation and Administration Guide:

This guide describes how to set up and administer Oracle Procurement Contracts. Oracle Procurement Contracts enables employees to author and maintain complex contracts through a self-service, Web interface.

Oracle Public Sector Financials User Guide:

This guide describes how to set up and administer Oracle Public Sector Advanced Features. It describes Encumbrance Reconciliation Reports, GASB 34/35 Asset Accounting, and Funds Available Enhancements.

Oracle Purchasing User's Guide:

This guide describes how to create and approve purchasing documents, including requisitions, different types of purchase orders, quotations, RFQs, and receipts. This guide also describes how to manage your supply base through agreements, sourcing rules, and approved supplier lists. In addition, this guide explains how you can automatically create purchasing documents based on business rules through integration with Oracle Workflow technology, which automates many of the key procurement processes.

Oracle Receivables User Guide:

This guide provides you with information on how to use Oracle Receivables. Use this guide to learn how to create and maintain transactions and bills receivable, enter and apply receipts, enter customer information, and manage revenue. This guide also includes information about accounting in Receivables. Use the Standard Navigation Paths appendix to find out how to access each Receivables window.

Oracle Receivables Implementation Guide:

This guide provides you with information on how to implement Oracle Receivables. Use this guide to understand the implementation steps required for application use, including how to set up customers, transactions, receipts, accounting, tax, and collections. This guide also includes a comprehensive list of profile options that you can set to customize application behavior.

Oracle Receivables Reference Guide:

This guide provides you with detailed information about all public application programming interfaces (APIs) that you can use to extend Oracle Receivables functionality. This guide also describes the Oracle Receivables open interfaces, such as AutoLockbox which lets you create and apply receipts and AutoInvoice which you can use to import and validate transactions from other systems. Archiving and purging Receivables data is also discussed in this guide.

Oracle Sourcing Implementation and Administration Guide:

This guide contains information on how to implement Oracle Sourcing to enable participants from multiple organizations to exchange information, conduct bid and auction processes, and create and implement buying agreements. This allows professional buyers, business experts, and suppliers to participate in a more agile and accurate sourcing process.

Oracle Subledger Accounting Implementation Guide:

This guide provides setup information for Oracle Subledger Accounting features, including the Accounting Methods Builder. You can use the Accounting Methods Builder to create and modify the setup for subledger journal lines and application accounting definitions for Oracle subledger applications. This guide also discusses the reports available in Oracle Subledger Accounting and describes how to inquire on subledger journal entries.

Oracle Supplier Scheduling User's Guide:

This guide describes how you can use Oracle Supplier Scheduling to calculate and maintain planning and shipping schedules and communicate them to your suppliers.

Oracle iProcurement Implementation and Administration Guide:

This manual describes how to set up and administer Oracle iProcurement. Oracle iProcurement enables employees to requisition items through a self-service, Web interface.

Oracle Procurement Contracts Implementation and Administration Guide:

This manual describes how to set up and administer Oracle Procurement Contracts.

Oracle Procurement Contracts enables employees to author and maintain complex contracts through a self-service, Web interface.

Oracle Trading Community Architecture User Guide:

This guide describes the Oracle Trading Community Architecture (TCA) and how to use features from the Trading Community Manager responsibility to create, update, enrich, and cleanse the data in the TCA Registry. It also describes how to use Resource Manager to define and manage resources.

Oracle Trading Community Architecture Administration Guide:

This guide describes how to administer and implement Oracle Trading Community Architecture (TCA). You set up, control, and manage functionality that affects data in the TCA Registry. It also describes how to set up and use Resource Manager to manage resources.

Oracle Trading Community Architecture Reference Guide:

This guide contains seeded relationship types, seeded Data Quality Management data, D&B data elements, Bulk Import interface table fields and validations, and a comprehensive glossary. This guide supplements the documentation for Oracle Trading Community Architecture and all products in the Oracle Customer Data Management family.

Oracle Trading Community Architecture Technical Implementation Guide:

This guide explains how to use the public Oracle Trading Community Architecture application programming interfaces (APIs) and develop callouts based on Oracle Workflow Business Events System (BES). For each API, this guide provides a description of the API, the PL/SQL procedure, and the Java method, as well as a table of the parameter descriptions and validations. For each BES callout, this guide provides the name of the logical entity, its description, and the ID parameter name. Also included are setup instructions and sample code.

Oracle U.S. Federal Financials User's Guide:

This guide describes the common concepts for an integrated financial management solution for federal agencies to comply with the requirements of the U.S. Federal government. It describes the product architecture and provides information on Budget Execution, Prompt Payment, Treasury payments, Third party payments, Interagency transactions, Receivables management, Federal reports, CCR Integration, and Year End Closing.

Oracle U.S. Federal Financials Implementation Guide:

This guide describes the common concepts for an integrated financial management solution for federal agencies. It includes a consolidated setup checklist by page and provides detailed information on how to set up, maintain, and troubleshoot the Federal Financial application for the following functional areas: Sub Ledger Accounting, Budget Execution, Prompt Payment, Treasury payments, Third party payments, Interagency transactions, Receivables management, Federal reports, CCR Integration, and Year End Closing.

Oracle Projects Documentation Set

Oracle Projects Implementation Guide:

Use this guide to implement Oracle Projects. This guide also includes appendices covering function security, menus and responsibilities, and profile options.

Oracle Project Costing User Guide:

Use this guide to learn detailed information about Oracle Project Costing. Oracle Project Costing provides the tools for processing project expenditures, including calculating their cost to each project and determining the General Ledger accounts to which the costs are posted.

Oracle Project Billing User Guide:

This guide shows you how to use Oracle Project Billing to define revenue and invoicing rules for your projects, generate revenue, create invoices, and integrate with other Oracle Applications to process revenue and invoices, process client invoicing, and measure the profitability of your contract projects.

Oracle Project Management User Guide:

This guide shows you how to use Oracle Project Management to manage projects through their lifecycles - from planning, through execution, to completion.

Oracle Project Portfolio Analysis User Guide:

This guide contains the information you need to understand and use Oracle Project Portfolio Analysis. It includes information about project portfolios, planning cycles, and metrics for ranking and selecting projects for a project portfolio.

Oracle Project Resource Management User Guide:

This guide provides you with information on how to use Oracle Project Resource Management. It includes information about staffing, scheduling, and reporting on project resources.

Oracle Grants Accounting Documentation

Oracle Grants Accounting User Guide:

This guide provides you with information about how to implement and use Oracle Grants Accounting. Use this guide to understand the implementation steps required for application use, including defining award types, award templates, allowed cost schedules, and burden set up. This guide also explains how to use Oracle Grants Accounting to track grants and funded projects from inception to final reporting.

Oracle Property Manager Documentation

Oracle Property Manager User Guide:

Use this guide to learn how to use Oracle Property Manager to create and administer properties, space assignments, and lease agreements.

Oracle Property Manager Implementation Guide:

Use this guide to learn how to implement Oracle Property Manager and perform basic setup steps such as setting system options and creating lookup codes, contacts, milestones, grouping rules, term templates, and a location hierarchy. This guide also describes the setup steps that you must complete in other Oracle applications before you can use Oracle Property Manager.

Integration Repository

The Oracle Integration Repository is a compilation of information about the service endpoints exposed by the Oracle E-Business Suite of applications. It provides a complete catalog of Oracle E-Business Suite's business service interfaces. The tool lets users easily discover and deploy the appropriate business service interface for integration with any system, application, or business partner.

The Oracle Integration Repository is shipped as part of the Oracle E-Business Suite. As your instance is patched, the repository is automatically updated with content appropriate for the precise revisions of interfaces in your environment.

Do Not Use Database Tools to Modify Oracle E-Business Suite Data

Oracle **STRONGLY RECOMMENDS** that you never use SQL*Plus, Oracle Data Browser, database triggers, or any other tool to modify Oracle E-Business Suite data unless otherwise instructed.

Oracle provides powerful tools you can use to create, store, change, retrieve, and maintain information in an Oracle database. But if you use Oracle tools such as SQL*Plus to modify Oracle E-Business Suite data, you risk destroying the integrity of your data and you lose the ability to audit changes to your data.

Because Oracle E-Business Suite tables are interrelated, any change you make using an Oracle E-Business Suite form can update many tables at once. But when you modify Oracle E-Business Suite data using anything other than Oracle E-Business Suite, you may change a row in one table without making corresponding changes in related tables. If your tables get out of synchronization with each other, you risk retrieving erroneous information and you risk unpredictable results throughout Oracle E-Business Suite.

When you use Oracle E-Business Suite to modify your data, Oracle E-Business Suite automatically checks that your changes are valid. Oracle E-Business Suite also keeps track of who changes information. If you enter information into database tables using database tools, you may store invalid information. You also lose the ability to track who has changed your information because SQL*Plus and other database tools do not keep a record of changes.

Overview

Introduction to Oracle Payments

Oracle Payments is a product in the Oracle E-Business Suite of applications, which serves as a funds capture and funds disbursement engine for other Oracle applications. As the central payment engine, Oracle Payments processes transactions, such as invoice payments from Oracle Payables, bank account transfers from Oracle Cash Management, and settlements against credit cards and bank accounts from Oracle Receivables. Oracle Payments provides the infrastructure needed to connect these applications and others with third party payment systems and financial institutions.

The centralization of payment processing in Oracle Payments offers many benefits to deploying companies. Companies can efficiently centralize the payment process across multiple organizations, currencies, and regions. Better working capital management can be achieved by providing cash managers visibility into cash inflows and outflows. Additionally, a full audit trail and control is supported through a single point of payment administration.

Oracle Payments is integrated with financial institutions and payment systems for receipt and payment processing, known as funds capture and funds disbursement, respectively. Funds capture refers to the electronic retrieval of funds, typically by a payment system on behalf of the deploying company, from payers, such as customers, who owe debts to the deploying company. The payer, in this case, provides Oracle Payments with pertinent payment information, such as a credit card, debit card, or bank account number. Funds disbursement, on the other hand, is the process of paying funds owed to creditors, such as suppliers.

Oracle Payments supports the following electronic payment methods for funds capture payments:

- credit cards
- purchase cards

- PINless debit cards
- bank account transfers

Oracle Payments supports several payment methods for funds disbursement payments, including:

- checks
- wires
- electronic funds transfers

General Features

The following sections describe the general features of Oracle Payments.

Advanced and Highly Configurable Formatting Framework

Financial institutions and payment systems require compliance with specific payment formats to disburse or capture funds. A large part of the effort and cost involved in establishing a relationship with a payment system is ensuring that a valid, correctly formatted payment file or message can be produced. To minimize this effort, Oracle Payments provides a formatting solution based on standard XML technology.

Formats are created as templates in Oracle XML Publisher and applied to an XML data file produced by Oracle Payments. These templates can be created or modified with minimal effort, using a standard text editor, such as Microsoft Word. Consequently, when a payment system or financial institution requires a change to its format, you can easily and quickly make the change.

Special consideration has been given to the complexity of creating fixed position and delimited formats. Oracle XML Publisher's eText feature is used for these format types. eText allows the format layout to be presented in an understandable tabular structure.

Oracle Payments offers an extensive library of payment formats that support various types of payment files and messages. EFT disbursements, printed checks, ACH debits, bills receivable remittances, and credit card authorizations and settlements are all supported. Deploying companies can use any format in this library. If a format is required, but it is not provided, it is easy to copy a similar format template and use it as the basis for creating a new format. This feature greatly speeds implementation and testing, as well as reduces implementation costs.

Flexible Validation Model

In payment processing, it is critical to ensure that payment messages and files sent to third party payment systems and financial institutions are both valid and correctly formatted. If this does not occur, the payment process is slowed and a cost is incurred to

resolve the issue. To achieve straight through processing, Oracle Payments introduces a flexible way to ensure that payment-related validations are implemented.

Oracle Payments provides an extensive library of payment validations that are associated with the supported payment formats. Payment validations are implemented using a flexible framework that allows you to add new validation rules. Deploying companies can choose between using the prepackaged library of validations, using newly added validations, or using a combination of prepackaged and newly added validation rules.

The timing of validation execution also has flexibility. Payment validation rules can be assigned early or late in the payment process. Alternatively, a combination of early and late validation rule assignment can be used. You have flexibility in assigning validation rules to support your business model. For example, assigning and executing validation rules early in the payment process may be best for a decentralized payment environment. On the other hand, validation execution later in the process may be optimal in a shared service environment, where payment specialists can resolve any validation failures.

Secure Payment Data Repository

Oracle Payments serves as a payment data repository on top of the Trading Community Architecture (TCA) data model. The TCA model holds party information. Oracle Payments stores all of the party's payment information and its payment instruments, such as credit cards and bank accounts. This common repository for payment data provides data security by allowing central encryption management and masking control of payment instrument information.

Electronic Transmission Capability

Oracle Payments provides secured electronic payment file and payment message transmission, as well as transmission result processing. All previously existing electronic transmission features in Oracle Payments, Oracle Payables, and Oracle Globalizations are obsolete and have been replaced by the central Oracle Payments engine.

The following industry-standard transmission protocols are supported:

- FTP
- HTTP
- HTTPs
- AS/2

Flexible Support for Various Business Payment Models

Companies model their business units in various ways to obtain performance improvements and cost savings. Oracle Payments can be configured to support a variety of payment models. It can operate in a completely decentralized model where it is part of accounts payable or collection administration within each business unit. If your company has centralized financial activities in a shared service center, Oracle Payments also works to efficiently support the shared service center model.

Oracle Payments' flexibility also provides support to companies that wish to use a payment factory model. A payment factory allows operating units to maintain their own accounts payable and other payment administrative functions. The role of the payment factory is to handle communication and transactions with the deploying company's banking partners. Invoice selection can be done in Oracle Payables within a single operating unit. Then a payment factory administrator using Oracle Payments can consolidate payments from different operating units into a single payment file for transmission and settlement, thereby reducing transaction costs.

Oracle Payments supports Multi-Org Access Control throughout its processes to build payments and create payment files, as well as in the dashboard pages, which are provided for you to monitor and manage the payment process. Multi-Org Access Control enables you to efficiently process business transactions by processing and reporting on data that resides in an unlimited number of operating units, within a single applications responsibility. Data security is maintained using security profiles, which are defined for a list of operating units and data access privileges.

Funds Disbursement Features

Oracle Payments' funds disbursement features support the process to pay funds owed to creditors, such as suppliers. Oracle Payments supports functionality to achieve straight through processing of electronic funds transfers and the ability to format and manage check or other payment document printing.

Flexible Disbursement Process

The Oracle Payments disbursement engine enables you to greatly simplify your procedures for managing complex payment processes that span multiple payment methods, formats, check stocks, transmission protocols, currencies, organizations, and bank accounts.

The Oracle Payments funds disbursement process enables users of Oracle Payables and other products to focus on the invoice selection process before submitting the documents payable to Oracle Payments for payment processing. This is achieved by splitting the payment build process into two processes. The first process creates payments by grouping documents according to various rules, such as the payment method and currency. The second process then aggregates payments into payment instruction files, formats the files, and handles additional processing, such as printing

and transmission. This aggregation process can be done for payments created from multiple document selections and submissions to Oracle Payments.

With Oracle Payments, accounts payable managers can simplify their processes by submitting fewer invoice selection batches, with each one spanning multiple payment methods, formats, bank accounts, and payment currencies. Invoices can be selected for payment based on business reasons, such as maximizing discounts. Additionally, payment administrators can lower the cost of the disbursement process by creating check runs and EFT payment files that span multiple invoice selection batches and multiple organizations.

Electronic Payment Processing

Oracle Payments offers end-to-end electronic payment processing that includes validation, aggregation, formatting, and secure transmission of payments to financial institutions and payment systems. Straight through processing results in lower costs associated with the disbursement process. The formatting framework, flexible validation model, and electronic transmission capability greatly minimizes any need for writing payment customizations.

Configurability for Funds Disbursement Processing

Oracle Payments offers flexible setup to configure funds disbursement processing. Flexible setup ensures an implementation that best supports a controlled and efficient disbursement flow. Configuration options are available in the following areas:

- **Payment Methods:** Each document to be paid, such as an invoice, requires the specification of a payment method to indicate how it will be handled in the funds disbursement process. Payment methods can be defined as broadly or as narrowly as appropriate. Rules can be defined to specify when payment methods can be used on documents and how to default payment methods on documents when they are created.
- **Processing Rules:** The payment method on a document is linked to user-defined funds disbursement processing rules configured in Oracle Payments. Oracle Payments holds these funds disbursement processing rules in an entity called the Payment Process Profile. You can configure as many payment process profiles as you need for your payment processes. Each payment process profile contains rules for building documents into payments, aggregating payments into payment instruction files, and formatting payment instruction files. Processing rules for printing checks, transmitting electronic files, generating separate remittance advice notifications, and other options can be easily configured.

Payment Processing User Interface

Oracle Payments provides a funds disbursement dashboard for managing the funds disbursement process. The funds disbursement dashboard allows payment

administrators to manage every aspect of the process across multiple organizations from a central location in the application. This benefits companies by providing full visibility of a payment as it moves through the financial supply chain. Additionally, the single location for monitoring the payment process also helps streamline process management.

Payment administrators can use the funds disbursement dashboard to monitor the payment process and ensure that it is running smoothly. If any problems arise during the process, they are highlighted in a pending actions region. This is also the place where notification of any required actions is shown. Actions may be required based on the configuration choices made during implementation. The funds disbursement dashboard enables the payment administrator to navigate to the appropriate area to take corrective or required action, based on the pending issue selected.

Payment administrators can take the following actions from the funds disbursement dashboard:

- review validation errors
- review and optionally modify proposed payments
- transmit payment files or retry failed file transmission processes
- initiate check printing and record printing results

User-Friendly Check Printing Process

Oracle Payments enables you to easily initiate check printing, recover from printing errors, and record print results.

In Oracle Cash Management, check stock can be set up as prenumbered or numbering can be printed as part of the check format, which is common for blank checks. Oracle Payments uses this setup to present the payment administrator with the appropriate print actions for different check stock types. The funds disbursement dashboard guides the payment administrator to each printing action that can be performed.

Global Features in Oracle Payments

Oracle Payments provides country-specific payment features. These global features are available in all deployments of Oracle Payments without any limitation by geography, while still enabling you to configure processing by region, where appropriate.

Most of the global payment features support the funds disbursement process. For example, Oracle Payments enable you to:

- provide payment-level text messages for the payee
- conditionally report payments to a country central bank

- specify payment codes required by a financial institution for instructions on transaction handling, payers of bank charges, and statutory payment reasons
- configure the payment process to place formatted payment files in secure file directories

Library of Payment Formats

All payment formats used by Oracle E-Business Suite applications for the funds disbursement process have been consolidated as Oracle XML Publisher templates and are used by Oracle Payments. These formats include all those previously supported in Oracle Payables, Oracle U.S. Federal Financials, and Oracle Global Financials.

The following industry standard payment formats are supported for use by all products that integrate with Oracle Payments:

- EDIFACT PAYMUL
- ANSI X12 820
- U.S. NACHA

Remittance Advice Reporting

Oracle Payments supports remittance advice reporting and notifying a payee of the remittance detail when a payment is made. The notification process enables you to set conditions for producing remittance advice. Configuration is also supported to set how the remittance advice is to be delivered to a payee: via e-mail, fax, or printing and sending manually. All of these features enable you to better support your payment relationships with suppliers or other payees.

Funds Capture Features

Funds capture features support the process to electronically receive funds owed deploying companies by debtors, such as customers. Oracle Payments supports authorization and settlement of funds against credit cards and PINless debit cards, refunds to credit cards, electronic funds transfers from bank accounts, and formatting of bills receivable. A refund can be made only once and subsequent refunds are considered as chargebacks.

Note: Oracle Receivables retains the functionality of lockbox processing and electronic upload of remittance messages.

Flexible Payment Methods

Businesses and their customers require flexible payment methods when paying for goods and services. Oracle Payments supports credit cards, purchase cards (levels 2 and 3), PINless debit cards, funds capture electronic funds transfers, and the transmission of bills receivable transactions.

Oracle Payments processes credit card validation, authorization, and funds capture. Credit card validation can be done independently of authorization. This helps the merchant save transaction fees by preempting authorization requests on invalid credit cards. Oracle Payments also processes funds capture for credit card transactions that have been manually authorized over the telephone, which is a requirement for certain businesses.

Oracle Payment's support for funds capture electronic fund transfers (EFT) and bills receivable transmissions enables you to eliminate paper-based transactions, thus reducing administrative manual processing and errors. You will also receive funds faster with EFT than with traditional paper checks.

PINless Debit Card Transactions

Oracle Payments supports PINless debit card funds capture transactions. Sometimes referred to as Debit Bill Pay, this payment method is allowed by the debit networks in certain industries, including utilities, telecom, cable/satellite, government, education, and financial services.

Direct Debits

Online validation for electronic funds transfers is supported by Oracle Payments through APIs. The validation service is provided by some payment systems to perform validity checks on the payer bank account to be debited. Typically this service verifies that the bank account number is valid and not cited for fraudulent payment activity.

Oracle Payments extends its support of electronic funds transfer by adding third party certifications for Paymentech and Concord EFSnet.

Integrated with Oracle E-Business Suite

Oracle Payments is a key component of Oracle's E-Business Suite and integrates with other Oracle applications, such as Oracle Receivable, iStore, and Order Management. Through these integration points, Oracle Payments supports the payment processing needs for a variety of business activities.

Credit Card Security Features

Credit Card Encryption is an advanced security feature within Oracle Payments that enables Oracle Applications to encrypt credit card data. You can access this credit card security feature setup easily by selecting the Oracle Payments Payment Administrator

responsibility and then clicking the System Security Management link. Oracle Payments consolidates the varying levels of credit card security support within different products of the Oracle E-Business Suite so that setup and management of address verifications, capture of card security codes, and masking of credit card numbers, ensures a consistent implementation of credit card security functions throughout the funds capture process.

Note: Any internal credit card numbers, such as company-owned or employee credit cards, are not impacted by the Credit Card Encryption feature.

Encryption

Use of the Credit Card Encryption feature assists with your compliance with the cardholder data protection requirements of the Payment Card Industry (PCI) Data Security Standard and with Visa's Cardholder Information Security Program (CISP). The Visa program is based on the PCI Data Security Standard. When the feature is enabled, credit card numbers for external third party payers, such as customers or students, are encrypted.

Masking

When Credit Card Encryption is enabled, masking functionality differs between Oracle Applications. For example, in Oracle Receivables and Oracle Service Contracts, when Credit Card Encryption is enabled, credit card numbers are masked in the following format with the last four digits displayed: XXXXXXXXXXXXXXXX, whereas, in Oracle Order Management, a changed profile option no longer controls masking of credit card numbers.

Your Company's Security Policy

Adoption of the Credit Card Encryption feature should be part of the implementation of a complete security policy, specific to your organization. For example, your security policy should include a regular schedule to rotate keys to secure your credit card data.

For general guidelines on securing Oracle E-Business Suite applications products, see *Best Practices for Securing Oracle E-Business Suite*, Oracle MetaLink Document 189367.1.

Supports Both Processor and Gateway Model Payment Systems

Oracle Payments support for both processor and gateway models enables customers to choose the payment processing options that suit their business.

Processor and gateway model payment systems can be integrated with Oracle Payments for credit cards, PINless debit cards, and bank account transfers, using the product's format and transmission model. Oracle Payments payment system integration model has extensible fields that can be used in a custom implementation with payment

systems that require additional information. This extensibility can be particularly useful for international implementations of Oracle Payments.

In addition, Oracle Payments supports out-of-the-box integration with leading third party payment systems. These payment systems include Paymentech, First Data Merchant Services, and Concord EFS. Other payment systems, such as Verisign, offer their own out-of-the-box integrations with Oracle Payments.

Routing to Multiple Payment Systems for Funds Capture

Oracle Payments supports multiple payment processing systems operating simultaneously for funds capture transactions and provides a powerful routing system that gives businesses and merchants control over funds capture transaction processing. Payment requests are routed to payment processing systems based on flexible business rules defined by the merchant. For example, you can route payment transactions based on the type of transaction or amount of payment

Centralized Configuration

Oracle Payments offers a centralized, configurable funds capture processing setup. This ensures an implementation that supports consistent and seamless funds capture processing. The configurable funds capture processing setup can be grouped into the following areas:

- **Payee Configuration:** A payee is defined for one or more operating units in the deploying company that processes payments. The payee setup provides various options for payment processing and links operating units to the payee.
- **Routing Rules Configuration:** Routing rules can be configured to specify how a transaction is processed. For example, this setup determines the payment system to which a transaction is sent.
- **Funds Capture Processing Rules Configuration:** Oracle Payments holds funds capture processing rules in an entity called the Funds Capture Process Profile. You can configure as many funds capture process profiles as you need for your payment processes. Each profile contains the configuration for formatting and transmitting authorization messages and settlement files. Additionally, specifying rules for aggregating settlements into batches, limiting the number or amount of settlements in a batch, notifying payers of settlements, and processing acknowledgements can be easily configured.

Payment Processing User Interface

Oracle Payments offers a user interface, known as the funds capture dashboard, for managing the funds capture process. The funds capture dashboard provides an overview of the payment process status. This results in greater insight into rejections received from payment systems and into process failures, such as communication

errors.

Library of Payment Formats

All payment formats used by E-Business Suite applications for the funds capture process have been consolidated as Oracle XML Publisher templates and are used by Oracle Payments. These formats include Oracle Globalizations' direct debits, bills receivable, and Oracle Receivables' remittance formats.

Scheduled Payments

Oracle Payments processes payment requests when requested or via an offline scheduled process, based on the type of payment request and payment system. A scheduling program enables Oracle Payments to process gateway-bound payments in an offline mode when a source product requests it. When an E-Business application calls Oracle Payments to make an offline payment request, the payment information is stored in Oracle Payments and picked up by the scheduling program so that the payment gets settled by the due date (or settlement date). At the same time, Oracle Payments handles processor-model payment requests as required by most processors: online requests for authorizations and batched offline requests for follow-up operations.

Payer Notifications of Settlement

Oracle Payments has consolidated notification letters from Oracle Globalizations into an Oracle XML Publisher format. These notification letters are generated and sent when a payer's bank account is debited to capture funds for payment. The Oracle XML Publisher format supports notification for all types of automatic funds capture settlements supported by Oracle Payments, including card payments or bank account transfers. The notification process enables you to configure how a notification is to be delivered to a payer: via e-mail, fax, or printing and sending manually. These notifications help deploying companies better support their payment relationships with customers or other payers.

Planning Your Implementation

Shared Planning

Note: Before implementing Oracle Payments, it is advisable to answer the questions in this section to ensure that your implementation meets your business needs.

This section presents implementation questions common to both funds capture and funds disbursement functionality and are, therefore, relevant to shared planning.

What are Your Security Needs?

System security functionality is common to both funds capture and funds disbursement. When implementing Oracle Payments, you must specify appropriate security options for:

- encrypting sensitive information about payment instruments
- masking payment instruments and external bank accounts
- verifying credit cards using the Address Verification System

Security issues to address before implementation include the following:

- internal security requirements (data storage)
- external security requirements (transmission)
- fraud protection

Before implementing Oracle Payments, ensure that you understand the transmission security requirements of your payment system(s) and make sure that your network supports those requirements.

Note: Oracle Payments provides support for SSL, Secure FTP, and AS2, but you must mold this support into a comprehensive transmission security solution.

For information on setting up security options, see Step 2. Setting Up System Security Options, page 4-7.

For information on enabling third party payment systems to verify credit card owners, see Verifying Credit Card Owners, page 4-9.

For information on creating risk formulas to evaluate the risk of customers who wish to buy products or services from a payee, see Creating a Risk Formula, page 6-11.

This is the system-wide key used to encrypt all non-payee associated sensitive data (that is, basically registered credit card and bank account instruments) in the first version of the encryption functionality. It contains the "hash" of the system-wide key and is used to ensure correctness of the key provided at startup time. The IBY: System Security Key profile option is obsolete with the credit card encryption patch.

Do You Want to Use Funds Capture or Funds Disbursement Functionality?

Funds capture functionality involves the automated retrieval of payment from third party payers who owe debts to the first party payee, using electronic payment methods, such as direct debits of bank accounts, credit cards, and remittance of bills receivable.

Funds disbursement functionality involves making payments to third party payees. A payment can take an electronic form, such as EFT or wire, or a printed form such as a check.

You can choose to use either functionality or both, depending on your business needs.

Related Topics

For information on funds capture functionality, see Understanding Funds Capture, page 3-8.

For information on funds disbursement functionality, see Understanding Funds Disbursement, page 3-26.

Which Payment System do You Want to Use?

For electronic payment processing, you must decide which payment system you want to use. Oracle Payments requires partnering with a third party payment system for processing electronic funds capture and funds disbursement transactions.

The table below lists the payment systems that are integrated and shipped with Oracle Payments for funds capture functionality, along with the payment methods that each payment system supports.

Payment Systems Integrated and Shipped with Oracle Payments for Funds Capture Functionality, along with their Associated Payment Methods

Payment System	Credit Card	Purchase Card	PINless Debit Card	Bank Account Transfer	Electronic Funds Transfer Online Validation
Paymentech	Yes	Yes	Yes	Yes	Yes
First Data (North)	Yes	Yes	No	No	No
Concord EFS	Yes	Yes	Yes	Yes	No

** U.S. ACH only

Note: The supported operations listed in the preceding table may change. For current information, contact your payment system.

There are other payment systems that support their own Oracle Payments integrations. The table below lists a payment system that provides its own integration with Oracle Payments, along with the payment methods this payment system supports.

Payment Methods that VeriSign Supports

Payment System	Credit Card	Purchase Card	PINless Debit Card	Bank Account Transfer	Electronic Funds Transfer
VeriSign	Yes	No	No	Yes	No

Related Topics

For information on setting up payment systems, see Step 8: Setting Up Payment Systems, page 4-13.

What Source Products are You Using?

The source products you choose to use with Oracle Payments can be:

- Preintegrated Oracle applications

- Non-Oracle applications

Many Oracle applications, such as iStore, Order Capture, Telesales, Order Management, Oracle Receivables, Oracle Payables, and Collections are preintegrated with Oracle Payments out of the box.

Note: To ensure that preintegrated Oracle applications are set up correctly, see the setup documentation for each preintegrated Oracle application.

For non-Oracle applications, you must implement the appropriate APIs.

Related Topics

For information on APIs to implement with non-Oracle applications, see the Oracle Integration Repository at <http://irep.oracle.com/>.

What Formats do You Need to Support?

Payment format requirements usually originate from payment systems or central banks. Formats are created by deploying companies to format funds capture transactions and funds disbursement payment instructions. Oracle Payments uses Oracle XML Publisher formatting templates to format electronic transactions and payment instructions according to the formatting requirements of specific financial institutions. Oracle Payments also provides seeded payment validations that you can apply to the supported payment formats.

Related Topics

For information on creating payment formats and applying them to an XML data file produced by Oracle Payments, see Step 3. Setting Up XML Publisher Formatting Templates, page 4-9.

For information on setting up country-specific codes and identifiers provided by a country's government or central bank that relate to how the country-specific payment is to be processed, see Step 14. Setting Up Bank Instruction Codes, page 5-7.

For information on setting up country-specific codes and identifiers provided by a country's government or central bank that relate to how the country-specific payment is to be delivered to a payee, see Step 15. Setting Up Delivery Channel Codes, page 5-8.

For information on setting up country-specific codes and identifiers provided by a country's government or central bank that relate to the reason for the payments, see Step 16. Setting Up Payment Reason Codes, page 5-9.

Does Your Application Need to Present Information in Different Languages?

Before implementing Oracle Payments, you must determine whether your application

needs to present information in different languages.

Does Your Application Need to Use National Language Support (NLS)?

Your application may need to use NLS if either of the following is true:

- The source product and the payment system use different languages or character sets. For example, the source product may use a Japanese EUC character set, while the payment system uses a Japanese Shift-JIS character set.
- Clients of the source product use different languages. For example, a web site that expects customers to visit from all over the world might want to present its source product in different languages for different customers.

To enable character conversion in all these environments, the source product and the payment system must convey the language and character set information to Oracle Payments.

How do Applications Convey Language Information to Oracle Payments?

To communicate information about the language and character set to Oracle Payments, a source product and payment system servlet must pass a special parameter (NlsLang). This parameter is a part of every API included in this implementation guide.

NlsLang is an optional parameter. If your source product does not need to handle non-Latin1 character set parameters and does not need to communicate to clients or payment systems in different languages, you do not need to use this parameter.

How does Oracle Payments Use NlsLang?

If the source product does not pass the NlsLang parameter, Oracle Payments passes information from the source product to the payment service servlet without performing any conversion of character sets.

If the source product does pass a value for NlsLang to Oracle Payments, then Oracle Payments tries to convert parameters based on the value of NlsLang before sending those parameters to the payment system servlet.

To do so, Oracle Payments first checks its database for the list of preferred and optional languages for that payment system. The information in the database reflects what the Oracle Payments administrator entered using the Oracle Payments administration user interface.

Second, Oracle Payments does one of the following, depending on what it finds in the database:

- If the database lists a language that matches the value of NlsLang, Oracle Payments keeps the value of NlsLang and passes it to the payment system servlet.
- If the database does not list a language matching the value of NlsLang, Oracle

Payments uses the language specified as the preferred language for that payment system, thus changing the value of NlsLang before sending it to the payment system servlet.

Finally, Oracle Payments converts the values of other parameters so they are sent to the payment system servlet in the language specified by NlsLang.

This conversion process works only in one direction; from the source product to the payment system servlet. If the payment system sets up NlsLang when it sends the data back, Oracle Payments uses that information to store the value of OapfVendErrmsg in its database. Oracle Payments does not convert data sent from the payment system servlet back to the source product.

Format of the NLS_LANG Parameter

The value of this parameter follows the same format as Oracle Server's NLS_LANG environment variable:

`language_territory.charset`

For example, JAPANESE_JAPAN.JA16EUC is a valid value for NlsLang.

Format of the Response Body Data From Payment System Servlets

Oracle Payments does not convert the response received from the payment system servlet in the response body. It only treats the data as binary and sends it directly to the source product.

However, if any binary information is sent, such as wallet data, then Oracle Payments converts the character set of the binary data to that specified by the value of NlsLang.

Funds Capture Planning

This section presents planning questions relevant to funds capture.

How do You Need to Organize Your Settlement Batches?

For funds capture functionality, you must decide how settlements are to be grouped into settlement batches. Parameters for grouping can be specified in the funds capture process profile. For information on funds capture process profiles, see Step 21. Setting Up Funds Capture Process Profiles, page 6-4.

What Credit Card Brands do You Want to Support?

For funds capture functionality, you must decide which credit card brands your company wishes to accept for payment, as well as their associated authorization validity periods, in days.

Related Topics

For information on setting up credit card brands, see Step 23. Setting Up Credit Card Brands, page 6-12.

Which APIs do You Need to Use for Funds Capture?

Oracle Payments provides the following APIs:

- Payment Instrument Registration APIs to store payment instruments such as credit cards, bank accounts, PINless debit cards, and purchase cards.
- Payment Processing APIs to perform credit card, PINless debit card, and purchase card operations, such as authorization, capture, and bank account transfer operations
- Risk Management APIs to perform risk analysis

For funds capture functionality, you must decide, based on your requirements, what functionality your source products need, which determines which APIs to use.

Note: Each preintegrated Oracle application already implements the Oracle Payments API relevant to its operation. If you use a preintegrated Oracle application, then no further integration is needed.

Payment Instrument Registration APIs

Payment instrument registration is required. Non-Oracle products can implement payment instruments registration using Payment Instrument Registration APIs and instrument identifiers that are generated using payment requests with Oracle Payments.

Payment Processing APIs

You must decide whether you want to:

- implement online and/or offline payment processing
- accept credit card, PINless debit card payments, purchase cards, or bank account transfers, or some combination
- implement the risk functionality to detect fraudulent transactions

Once you decide on the functionality you wish to implement, you can then review the corresponding APIs on the Oracle Integration Repository at <http://irep.oracle.com/>.

Risk Management APIs

Oracle Payments can run credit card risk management during the authorization phase. If you want to independently perform risk evaluation, then separate risk management APIs can be called from your source products.

Related Topics

For additional information on the preceding APIs, see the Oracle Integration Repository at <http://irep.oracle.com/>.

Which Bank Account Transfer Operations do You Want to Implement for Funds Capture?

Oracle Payments supports bank account transfers as settlements. It also supports EFT online validations for bank account transfers. EFT validations help you verify that the payer's bank account is valid, but they do not place a hold on any funds and they are not offered by all payment systems or in all countries. The validations are online and real time, similar to credit card authorizations, whereas the actual funds transfers are performed offline as settlements. Funds transfers are not performed online since these transaction require one or two business days to complete.

The Source Product Reference number must not be the same as that of the Bank Account Transfer transaction number. It is usually the bank account transfer number plus one.

Related Topics

For information on implementing bank account transfers, see the Oracle Integration Repository at <http://irep.oracle.com/>.

Which Risk Factors do You Want to Implement?

Oracle Payments provides risk management functionality for credit card and purchase card transactions in E-Business applications. Oracle Payments includes a number of seeded risk factors and provides the option to payees or suppliers of running or not running the risk evaluation functionality for each authorization.

A risk factor includes any information which a payee wants to use to evaluate the risk of the customer wanting to buy goods or services from the payee. Examples of risk factors include address verification, time of purchase, and payment amount. These risk factors can be configured for each payee.

Risk management functionality enables payees to manage the risk involved in processing transactions online. It enables businesses to specify any number of predefined risk factors to verify the identity of their customers and assess their customer credit rating and risk rating in a secure environment.

Related Topics

Using Risk Management, page D-1

Funds Disbursement Planning

This section presents planning questions relevant to funds disbursement.

What Disbursement Payment Methods do You Want to Support?

Funds disbursement involves making payments from the first party payer, the deploying company, to third party payees, such as suppliers. A payment can be in electronic form, such as EFT or wire, or in printed form, such as a check.

For funds disbursement functionality, you must decide whether to setup more or less granular disbursement payment methods. The least granular payment methods are those seeded in Oracle Payments, such as Check or Electronic. They describe methods of making payments at the highest level. Under this kind of setup, you can associate many payment process profiles and payment formats with each method. This requires less knowledge from source product users such as invoice entry clerks, but possibly more work later in the payment process.

Alternately, you can define more granular payment methods and associate each to a single payment format and payment process profile. You can even choose to associate specific validations to each payment method. An example of a very granular payment method is Italian Electronic Funds Transfer. With this kind of setup, the source product user needs more knowledge. This approach has advantages, however. For example, payment method validations are run earlier at invoice entry time and thus, errors can be fixed more quickly.

Creating a funds disbursement payment method includes creating usage rules and setting validations. Usage rules specify when a disbursement payment method is available for use by source products for documents payable. By creating usage rules, you enable or disable payment methods for each source product integrated with Oracle Payments. You can provide different usage rules for different source products and change whether and when the payment method is available. Additionally, you can set validations so they run early in the payment process.

Related Topics

For information on funds disbursement payment methods, see Step 12. Setting Up Funds Disbursement Payment Methods, page 5-2.

Understanding Oracle Payments

Functionality Common to Both Funds Capture and Funds Disbursement

Payments functionality that is common to both the Funds Capture and the Funds Disbursements sides of Oracle Payments is the following:

- access control
- APIs
- servlets
- transmission
- security

Understanding Access Control

Access control is an Oracle Payments feature that enables you to control user access in viewing and/or updating data. It is primarily used to secure transaction data and is controlled by the user's security profile. Access control in Oracle Payments that is controlled by permissions on multiple organizations is known as Multiple Organization Access Control (MOAC). MOAC is setup using profiles in Oracle HRMS (Human Resource Management System), where you specify an organization or a hierarchy of organizations to which the user has access. The profile is then assigned to the user, responsibility, or other level.

Overview

Companies model their business units in various ways to maximize performance and cost savings. Oracle Payments can be configured to support a variety of payment models. It can work in a completely decentralized model, where each organization within the company handles its own financial activities. Alternatively, if a company

decides to centralize its financial activities in a shared service center, Oracle Payments can efficiently support the shared service center model also.

Additionally, Oracle Payments provides support to companies that wish to use the payment factory model. The payment factory model enables operating units to maintain their own accounts payable and other payment administrative functions. The payment factory handles communication and transactions with the deploying company's banking partners. Invoice selection occurs in Oracle Payables within a single operating unit. Then a payment factory administrator uses Oracle Payments to consolidate payments from different operating units into a single payment file for transmission and settlement, thereby reducing transaction costs.

Payment Function

A payment function is the type of payment made or captured by a source product. Each document payable has a payment function. For example, Oracle Payables supports the payment functions of making payments to suppliers and others. Different applications have different payment functions. In Oracle Payments, you can limit users to certain payment functions.

Multi-Organization Access Control (MOAC)

Oracle Payments supports Multi-Organization Access Control (MOAC). MOAC is a subfeature of Access Control that enables you to define multiple organizations and the relationships among them within a single installation of Oracle Applications. These organizations can be operating units, business groups, legal entities, sets of books, or inventory organizations.

The primary advantage of MOAC is that you can take action on documents in different operating units without logging into different responsibilities. Data security is maintained using security profiles that are defined for a list of operating units for which specific users are given data access privileges.

Access Control in Read-Only Pages

In read-only pages, you can only view documents or payments for which you have access. That is, the data you see displayed is that to which you have access.

Conversely, you are unable to view documents or payments that belong to an organization or payment function to which you do not have access. However, payment process requests, payment instructions, and funds settlement batches can contain data from multiple organizations and are not organization-specific. In these cases, if you have access to an organization that owns data in the request, instruction, or settlement batch, you will be able to see them. But within that request, instruction, or settlement batch, you will only be able to see the transaction data owned by the organizations to which you have access.

Access Control in Action Pages

Action pages are those pages where you take any kind of action on documents payable, payments, payment process requests, payment instructions, funds settlement transactions, or settlement batches. This includes printing checks, correcting validation errors, or taking other actions in the Funds Capture and Funds Disbursement Dashboards. You can only act on an entity if you have access to *everything* within the entity. For example, if a payment instruction includes payments for Organization 1 and Organization 2, and you only have access to Organization 1, you will be unable to go to any action page for the payment instruction. That is, if you only have partial access to the data in an object, you are unable to access that object. In this event, a message indicating that you do not have full access to the object is shown.

Access Control in Disbursement System Options Setup Page

In the Disbursement System Options setup page, you can view and update organization-level system options only for those organizations to which you have access. Oracle Payments restricts what you can see and update, in that you only see that for which you have access.

Access Control in all Other Setup Pages

In all other setup pages, Oracle Payments does not restrict you from what you can see and update, regardless of access control.

Access Control in Payment Process Concurrent Programs

The following concurrent programs are restricted by your access to organizations:

- Create Payment Instructions: Funds Disbursement. The program will only pick up those payments for which you have permission to take action.
- Create Settlement Batches: Funds Capture. The program will only pick up those transactions for which you have permission.

Understanding Oracle Payments APIs

Oracle Payments provides two types of APIs to simplify the integration of existing or new applications with Oracle Payments for payment processing.

- Oracle Applications APIs: Oracle Applications use these APIs to integrate with Oracle Payments for funds capture and funds disbursement. These are internal to Oracle Applications.
- Electronic Commerce APIs: Third-party applications can use these APIs to integrate their applications with Oracle Payments for funds settlement transactions. The third-party application can be a stand-alone application that communicates with

Oracle Payments via Java APIs or PL/SQL APIs.

- **Payment System Integration:** You can integrate with payment systems by creating or updating XML Publisher templates, transmission configurations, and payment system servlets. The templates are used to translate payment data into a payment system's format. Transmission configurations contain the details of transmission to payment systems and servlets manage the actual transmission process. Oracle Payments provides the Payment System Integration Model to interface with payment gateways and payment processors.

Understanding Oracle Payments Servlets

Oracle Payments consists of the following servlets:

- **ECServlet**

The ECServlet provides an interface to the Oracle Payments engine to process payment-related funds capture operations such as authorization. This servlet is primarily used for the PL/SQL APIs provided by Oracle Payments.

- **Payment system servlets**

Payment system servlets take payment files, as formatted by Oracle Payments, and transmit them to payment systems according to transmission configurations set up in Oracle Payments. Oracle Payments bundles payment system servlets developed by Oracle and/or interfaces with servlets developed by its payment system partners. The payment systems communicate with the payment acquirers or banks to process payment transactions. Oracle Payments includes payment system servlets for Paymentech, First Data (North), and Concord EFSnet. Some payment systems, such as VeriSign, have built their own Oracle Payment servlets.

- **Field-installable servlets**

Oracle Payment supports field-installable servlets. These payment system servlets are not bundled with Oracle Payments. This feature allows a payee to acquire a new, additional, or upgraded payment system servlet and configure it in the same way as the payment system servlets bundled with Oracle Payments.

The ability to add field-installable servlets provides payment flexibility and allows new releases of Oracle Payments and the payment systems to be independent of each other. It also enables users of source products to customize the payment system for their specific needs and regions.

Field-installable payment system servlets for Oracle Payments are available from Oracle's payment system partners, or can be custom built.

Understanding Transmission

To transmit data to or from a payment system, you need a transmission protocol and a

transmission configuration. First, you must have a transmission protocol, which defines the method of transmission. Transmission configurations, which specify concrete transmission details, must be associated with one transmission protocol.

To illustrate, Oracle Payments supports a generic transmission protocol for flat files called File Transfer Protocol for Static File Names. This protocol is a generic method to electronically transmit data, whereas FTP to Paymentech is a specific transmission configuration that specifies how to transmit data to a specific location using FTP as the method.

On the funds capture side, the transmission protocol and configuration are associated with the funds capture process profile, whereas on the funds disbursement side, the transmission configuration is associated with the payment process profile.

Transmission Protocol

Oracle Payments seeds transmission protocols, which are used by all funds capture process profiles and may be used by payment process profiles. These common seeded protocols, such as FTP, are comprised of the following:

- a code entry point, which the payment system servlet uses to accomplish transmission
- a list of parameters, such as network address and port, for which the transmission configuration must supply values
- transmission protocol entry points, which are independent of payment servlets and may be invoked from the Oracle Payments engine

You can view the seeded transmission protocols in Oracle Payments setup pages.

Transmission Configuration

Each transmission protocol has parameters that require values. The values defined for the parameters comprise the transmission configuration for the transmission protocol. For example, the payment system, Paymentech, may require a Socket IP Address of X and a Socket Port number of Y, among other values. Together XY, and the other values, represent Transmission Configuration A for a given transmission protocol.

Understanding Oracle Payments Security

Oracle Payments stores and processes highly sensitive financial data. To ensure proper security of this data, the Oracle Payments has advanced security features. Oracle Payments has several features to ensure the security and privacy of your data.

This section explains the security features of Oracle Payments and describes the setup required to properly utilize these features.

Multiple Organization Access Control

MOAC (Multi-Organization Access Control) is part of the Oracle Payments security feature. For information on MOAC, see Multi-Organization Access Control (MOAC), page 3-2.

Payment Instrument Encryption

Payment Instrument Encryption is an advanced Oracle Payments security feature that enables Oracle Applications to encrypt credit card data. This feature assists with your compliance with the cardholder data protection requirements of the Payment Card Industry (PCI) Data Security Standard and with Visa's Cardholder Information Security Program (CISP). The Visa program is based on the PCI Data Security Standard. When the feature is enabled, credit card and bank account numbers for external third parties, such as customers, suppliers, or students are encrypted.

Note: Other products such as iExpenses do store internal credit card numbers in Oracle Payments' tables.

Adoption of the Payment Instrument Encryption feature should be part of the implementation of a complete security policy, which is specific to your organization. For example, your security policy should include a regular schedule to rotate keys to secure your payment instrument data. For general guidelines on securing Oracle E-Business Suite applications products, see Best Practices for Securing Oracle E-Business Suite, Oracle MetaLink Document 189367.1.

Oracle Payments Engine to Oracle Payments Servlet Communication

Oracle Payments architecture lets you install the payment system servlet in a machine outside the firewall. If you have installed either Oracle Payments (or its components) or the source product in a distributed environment, Oracle recommends configuring SSL between Oracle Payments and the payment system components. You can create an Oracle Wallet to store certificates and credential information to support authentication of the engine, in this case a client of the servlet, by the server running the servlet. You can specify the wallet location and password using FND profiles. You can configure the server where the servlet is running to request client certificates (on the engine side). Oracle Payments retrieves the certificates from the Oracle Wallet and sends the certificates to the server for authentication. Oracle Payments also supports basic authentication of the payment system by the servlet.

These security features are recommended to guard against unauthorized access to data and Oracle Payments services. Oracle iAS web server (Apache Server) provides several types of authentication that you can use to secure servers, listeners, and servlets.

Firewall Protection

Oracle strongly recommends that you install Oracle Payments and the payment system

servlets on a machine inside the firewall.

Secure Socket Layer

If either Oracle Payments (or its components) or the source product is installed in a distributed environment, Oracle recommends enabling SSL communication between Oracle Payments and the payment system components.

Basic Authentication for Payment Systems

For setting up security for basic authentication between Oracle Payments and payment system servlets, you must perform some tasks both in the Oracle Payments setup user interface and in the Apache Server administration tool. While configuring Oracle Payments for a particular payment system, you must assign the payment system user name and password in the payment system configuration screens. You must assign the same user name and password in the Apache Server that runs the payment system servlets.

For details on setting up basic authentication in Apache Server, see the Apache Server documentation.

IP Address Restriction

In addition to using the merchant user name and password, you can restrict access to Oracle Payments and payment systems through IP address restriction. By using IP address restriction, a feature of the Apache Server, you can specify one or both of the following parameters:

- The IP addresses of all trusted hosts (machines from which the web server should accept transaction requests for Oracle Payments)
- The IP addresses of some non-trusted hosts (machines from which the web server should refuse transaction requests for Oracle Payments)

If a request is from a machine on the trusted list, Oracle Payments processes the requested transaction. If the request is from a machine on the non-trusted list, Apache Server denies the request and prevents Oracle Payments from processing it.

Through IP address restriction, you can limit access to all operations from non-trusted machines.

For more information about IP address restriction, including how to specify trusted hosts, see Apache Server documentation.

Other Security Related Information

- Separate HTTP ports for site administration and Oracle Payments administration increases security.

Related Topics

For information on Oracle Wallet, see *Oracle E-Business Suite System Administrator's Security Guide*.

Understanding Funds Capture

This section presents functionality and terms that are relevant to the funds capture process.

Understanding Funds Capture Process Profile

On the funds capture side, the funds capture process profile is a blueprint, which contains information that tells Oracle Payments how to communicate with the payment system. The funds capture process profile contains information for each of the following steps of the funds capture control process:

- online payer verification for a bank account transfer, online authorization for a credit card payment, or online debit for a debit card payment
- notification
- acknowledgement
- rules for building settlements into batches

The funds capture process profile on the funds capture side plays the same role as the payment process profile on the funds disbursement side. The funds capture process profile is defined to control the behavior of all funds capture operations, as listed above. Unlike the payment process profile, however, the funds capture process profile is always derived from routing rules, which are created during Payee setup.

Funds capture process profiles also specify the payment system account to be used for processing transactions.

Understanding Payment Methods (Funds Capture)

On the funds capture side of Oracle Payments, a payment method is the medium by which a third party payer chooses to remit payment to the first party payee. For example, a customer remits payment for a product or service to the deploying company. Oracle Payments supports the following payment methods for automated funds capture processing:

- bank account transfers
- credit cards

- PINless debit cards
- bills receivable remittances

Oracle Payments enables flexible setup of funds capture payment methods as follows:

- You can define unique payment methods for bank account transfers.
- Oracle Payments seeds a single payment method for credit card and PINless debit cards.

Understanding Payees

In Oracle Payments, the payee represents a first party entity that is collecting money from customers (in the case of funds capture payments). A payee also has a business relationship with a payment system, in which the payment system processes transactions for the payee.

You can have more than one payee in Oracle Payments for funds capture. You can have multiple payees, for example, because different business units or legal entities in one organization want to process transactions through different payment systems, or use separate relationships with a payment system.

When you create a payee in Oracle Payments, you must specify several pieces of information.

- organizations for which this payee processes transactions
- a list of payment instrument types that the payee accepts
- payment system accounts and funds capture process profiles that this payee uses to process transactions
- risk formulas
- routing rules

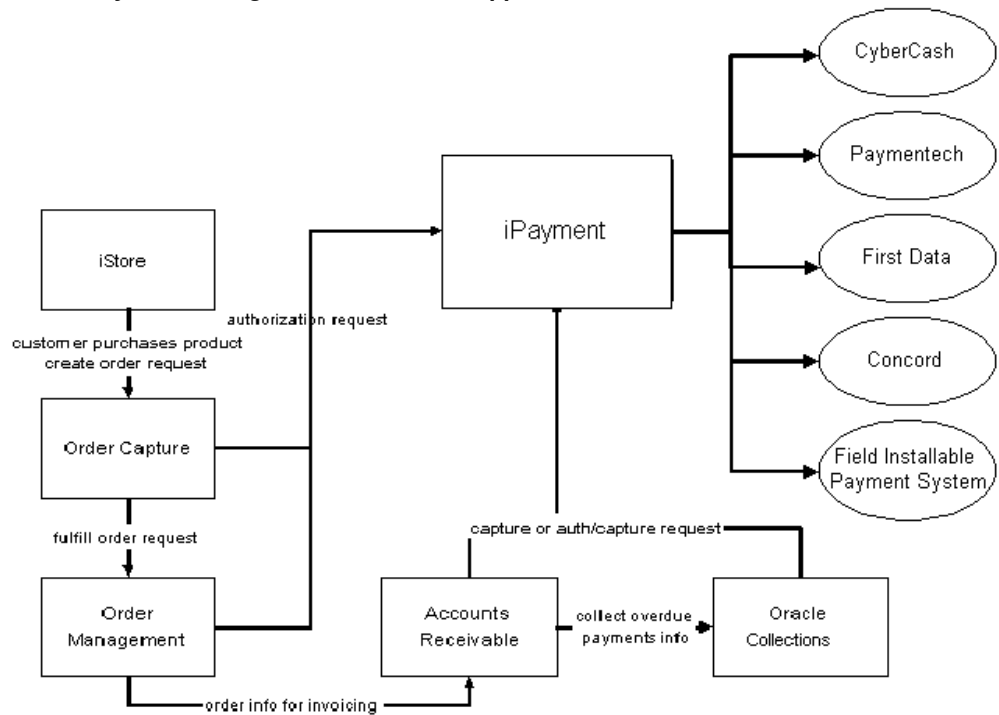
Integration with Other Oracle Applications

Oracle Payments integrates with other Oracle Applications to provide payment processing across your enterprise. Various source products send transaction requests to Oracle Payments for processing. Without Oracle Payment, each of these applications would need to build its own integrations to payment systems. Oracle Payment saves integration effort by providing a single source to payment systems such as Concord EFSnet, First Data North, and other country-specific or region-specific payment systems.

Example of a Payment Processing Flow Using Oracle Payments and Other Oracle Applications:

1. Sales application (for example, iStore or TeleSales): A customer purchases a product and decides to pay by credit card. The sales application submits the order.
2. Order Capture or Order Management: Order Capture and Order Management process the order. Both use Oracle Payments to verify if the credit card number is valid and authorize the order amount. They can also perform some risk evaluation as part of the authorization.
3. Oracle Receivables: When the order is shipped, the transaction information is passed to Oracle Receivables and the billing and takes place.
4. Oracle Collections: When the payment is overdue and your call center begins funds disbursement collection attempts, Oracle Collections uses Oracle Payments to authorize and settle credit card transactions.

Oracle Payments Integration with Oracle Applications



Understanding Credit Card Transactions

Among funds capture payments, Oracle Payments handles credit card, PINless debit card, purchase card, and EFT transactions. This section explains the process flow for a typical credit card transaction.

Traditional Credit Card Transactions

Traditional credit card transaction processing involves a third party payer, a first party payee, an issuing bank, an acquiring bank, a payment processor, and, optionally, a payment gateway.

A credit card transaction consists of two phases: authorization and settlement. A typical flow might occur as follows:

- Authorization

The third party payer purchases goods or services and sends credit card information as part of an order or invoice to the first party payee.

The first party payee accepts the order and sends an authorization request to the payment processor through Oracle Payments and optionally a payment gateway.

The payment processor matches the information with a database maintained by the issuing bank to determine if the customer has enough available credit to cover the transaction. If so, then the payment processor reserves the funds and sends an authorization code back to Oracle Payments.

- **Settlement**

The first party payee delivers goods to the customer and needs to settle the transaction, that is, capture the funds reserved in the authorization. Settlement may occur at the same time as authorization. Settling transactions includes batch administration.

The first party payee issues capture, void, return, credit, and settlement batch functions to the payment processor through Oracle Payments and optionally, the payment gateway.

The payment processor settles the payment with the issuing bank and causes the funds to transfer to the acquiring bank.

Voice Authorization

Sometimes credit card processing networks decline transactions with a referral message indicating that the merchant must call the cardholder's issuing bank to complete the transaction. The payment information in such cases is submitted over the phone. If the transaction is approved, the merchant is provided with an authorization code for the transaction. To facilitate follow-on transactions through Oracle Payments for this voice authorization (for example, capture or void), Oracle Payments provides voice authorization support for gateway-model and processor-model payment systems.

Understanding Purchase Cards

A purchase card is a type of credit card that is issued by an organization to its employees. The card is generally used by the employees for purchasing corporate supplies and services. Payments are made by the corporate buyer to the card issuer.

Benefits of Purchase Cards to the Merchant (First Party Payee)

- Businesses use purchase cards to cut costs and streamline labor intensive processes to procure goods and services. As a result, many buyers prefer merchants that accept purchase cards.
- Merchants generally receive better rates from card associations or issuing banks with purchase cards than with credit cards.

Benefits of Purchase Cards to the Buyer (Third Party Payer)

- A better reconciliation stream by providing purchase order number and additional information.

- Aggregation of purchases when companies receive one invoice for multiple purchase cards.
- Streamlining the purchase order process. Lower processing costs by simplifying the purchasing process, reducing paperwork, and automating controls on the spending limits.
- Merchants accepting purchase cards help the buyer by making purchase information available electronically. This may help companies (buyers) comply with tax regulations, reporting requirements, and expense reconciliation.

Purchase Card Data Levels

For a purchase card, three levels of data can be captured and sent by a merchant to the buyer organization through the payment system. They are:

Level I:

Level I transaction data consists of only basic data. A standard credit card transaction provides level I data to the processor. The buyer cannot derive any special benefits from purchase card usage if the merchant passes only level I data.

Level II:

Level II transaction data consists of data such as tax amount and order number in addition to level I data.

Level III:

Level III line item detail provides specific purchase information such as item description, quantity, unit of measure and price. This information is very useful to the buyer to help streamline accounting and business practices and to merge payment data with electronic procurement systems. Data in the table below is only indicative. The actual fields are payment system-dependent.

Note: Oracle Payments supports Level III data for both payment processors and gateway model payment systems.

The table below lists information on data that is passed by Oracle Payments in each level.

Data	Level 1	Level 2	Level 3
Card Number	X	X	X
Card Holder Name	X	X	X

Data	Level 1	Level 2	Level 3
Card Expiration Date	X	X	X
Card Holder Billing Address	X	X	X
Currency Code	X	X	X
Tax Amount		X	X
Transaction/Order Number		X	X
Ship from Postal Code		X	X
Destination Postal Code		X	X
Discount Amount			X
Freight Amount			X
Duty Amount			X
Line Item Information			X

Processing Purchase Card Transactions

The transaction phases in a purchase card transaction are the same as in a credit card transaction. The phases are authorization and settlement. For more information about transaction phases, see *Understanding Credit Card Transactions*, page 3-11.

Oracle Payments automatically recognizes purchase cards based on a set of seeded card number ranges. Oracle Payments passes additional information to the payment system during the settlement or settlement batch operation and through Gateway online requests. Authorization and other settlement operations carry the same information for purchase cards as they do for credit cards.

Understanding PINless Debit Card Transactions

PINless debit card transactions are a type of payment method offered by some payment systems to first party payees in selected industries that are traditionally viewed as

recurring billers. The consumer initiates the debit card payment process without providing a PIN. The merchant authenticates the consumers and assumes 100% liability for the transaction and any subsequent adjustments.

Process Flow for Gateway-Model Payment System

Most gateway type payment systems handle PINless debit card transactions in a single step. After receiving the authorization request, the payment system will send the transaction to the debit network. The consumer's account (payer) is debited after the authorization request is approved. The merchant's account (payee) is credited sometime later. PINless debit card transactions are different from the process flow of credit card transactions where the authorization and fund capture are separated into two steps.

To ease integrations for source products, Payments requires an authorization, but allows settlement to be optional.

The authorization step routes and sends the transaction to the payment system, saves the transaction into Oracle Payments and returns the authorization response to the source product (similar to credit cards).

If the source product invokes settlement, the original transaction is retrieved from Oracle Payments schema. If the transaction was authorized, a Success message is returned to the source product.

Process Flow for Processor-Model Payment System

For most payment processors, the flow for PINless debit cards is identical as that for payment gateways. However, some processor type payment systems (such as Paymentech) require an additional settlement step to complete the transaction. The first party payer's account will be credited after settlement and the fund transfer is said to be completed.

To ease integrations for source products, Payments requires an authorization, but allows settlement to be optional. If a payment system requires a settlement step, Payments automatically performs that step without input from the source product.

As in the gateway case, if the source product does invoke settlement, the original transaction is retrieved from Oracle Payments. If the transaction was authorized, a Success message is returned to the source product.

Understanding Funds Capture Bank Account Transfers

Oracle Payments supports funds capture bank account transfers for both business-to-consumer and business-to-business models. The funds capture bank transfer functionality facilitates electronic transfer of payment amounts from a customer's bank account to the payee's bank account.

Electronic Funds Transfer (EFT) Online Validations

EFT online validations are a real time service provided by some payment systems to validate the third party payer bank account to be used in an EFT transaction. EFT online

validations service ensures that the third party payer's bank account instrument exists and that there is no fraud alert. Electronic funds transfer transactions are not real time, since one or two business days are needed to complete the transaction. It is not possible, therefore to ensure that the bank account is still open and has sufficient funds. EFT online validation assists with validity checking as follows:

- The validation step is an optional step for EFT transactions. It can be performed any number of times.
- The validation is performed in real time.
- The EFT online validation response message shares the same message standard with the credit card authorization response message for processor type payment systems.

Note: EFT online validation is only offered for United States ACH and not for all payment systems. EFT online validation checks whether a bank account exists and that the account is not flagged as fraudulent. EFT online validation does not reserve funds or check if the account has sufficient funds.

Understanding Gateway-Model and Processor-Model Payment Systems

Oracle Payments supports both gateway-model and processor-model payment systems. The processor model describes the interface between Oracle Payments and a payment processor. A payment processor is a service that interacts directly with banks and card institutions such as Visa, Mastercard, and American Express, to process financial transactions. The gateway model describes the interface between Oracle Payments and a payment gateway. A payment gateway is a service provider that acts as an intermediary between a first party payee and a payment processor.

The choice of integrating to a gateway-model or processor-model payment system is generally determined by the business type, number of transactions per day, and the acquiring bank. Processors typically have more rigorous security, connectivity, and testing requirements. Gateways provide ease-of-use, often using SSL based internet connectivity. Gateways charge additional fees (including per-transaction fees), beyond what the processor charges. Typically, pricing varies by payment system and the processor model payment systems often favors higher-volume merchants who are willing to put forth the effort and cost of processor connectivity. The Gateway model favors lower-volume merchants, or merchants who are willing to pay a per-transaction premium for easier set up and connectivity.

A gateway-based system takes all transactions online. A processor-based system allows authorizations in real-time and follow-up transactions such as settlements and credits offline. Offline transactions must be batched together and sent as a single request to the payment system. All transactions other than authorizations are, by default, performed offline. Offline transactions are sent to the processor when the next settlement batch

operation is attempted.

You can create settlement batches manually or automatically according to a schedule, using the Funds Capture Process Home or the concurrent request manager. You can retrieve the final statuses of transactions in a settlement batch by retrieving clearing information. This is also done through the Funds Capture Process Home or the concurrent request manager.

Credit cards cannot be clubbed with other settlement batch. The settlement batch for refund is always a follow on transaction.

Understanding Terminal-Based and Host-Based Merchants

For gateway-model payment systems, Oracle Payments supports these processing models that the financial industry uses for credit card transactions:

- **Terminal-Based Merchant**

The first party payee determines when the payment gateway should create settlement batches.

- **Host-Based Merchant**

In this model, the payment gateway's host machine maintains all the transactions and is usually responsible for settlement batch operations at a predetermined frequency.

The choice of being a terminal-based or host-based merchant is generally determined by the business type, number of transactions per day, and the model supported by the payment gateway. The processing model that you choose affects how you perform the settlement operations. For a terminal-based merchant model, you must periodically perform settlement batch operations. Consult your acquiring bank for more information when you sign up.

Understanding Offline and Online Payments

Oracle Payments supports two models of payment processing for credit, purchase, and PINless debit cards as follows:

- Online Payment Processing, page 3-18
- Offline Payment Processing, page 3-18

Funds capture bank account transfers are always offline.

The types of operations that you can process online depends on the payment system type you have chosen: gateway or processor model and your operation model: host-based or terminal-based.

For processor model payment systems, authorization operations must be online and settlement operations are offline. For gateway payment systems, both authorization and

capture operations can be online.

Online Payment Processing

Online payment processing is the model in which a transaction is immediately forwarded to the payment system processor. The results from the payment system are immediately returned to the source product. Online transactions are supported for credit, purchase, and PINless debit cards. Online validation transactions are supported for Electronic Funds Transfer.

Offline Payment Processing

Offline payment processing is the model in which transactions are not immediately forwarded to payment systems. When a source product submits a transaction in a scheduled mode, or if the payment is predated, the payment information is saved in the Oracle Payments database and is sent to the payment system at a later time.

The offline method uses a concurrent program that submits offline transactions at regular intervals. The program browses the stored transactions and sends transactions to the payment systems and updates to the source products.

Understanding Risk Management

Fraudulent credit card payments continue to grow in number, as do subsequent losses by banks and merchants. Improved fraud detection has, therefore, become essential.

Oracle Payments provides risk management functionality for credit card and purchase card transactions. Oracle Payments includes a number of built-in risk factors and provides the ability to adjust risk calculation formulas and risk thresholds.

A risk factor includes any information which a first party payee wants to use to evaluate the risk of fraud when a third party payer attempts to make a payment. Examples of risk factors are: address verification, time of purchase and payment amount. These risk factors can be configured for each first party payee.

Risk management functionality enables first party payees to manage the risk involved in processing transactions online. It allows businesses to have several predefined risk factors to verify the identity of their customers and to assess their customers' credit rating and risk rating in a secure environment.

First party payees can associate the risk factors with different weights as a formula and define any number of risk formulas in Oracle Payments based on their business model. When a source product requests an authorization, it can specify the formula to be used to assess risk. When the source product requests an authorization with a risk formula specified, Oracle Payments evaluates the risk and, in parallel, sends the authorization request to the payment system. After getting a response from the payment system, Oracle Payments returns both the authorization code and the risk score to the source product. The source product now decides whether to continue with the transaction and make a settlement or discontinue the transaction.

Alternatively, the Risk API can be called independently of the authorization APIs. Using the Risk API separately allows merchants to evaluate risk first. Depending on the risk score, first party payees may not want to submit the authorization to a payment system and incur a transaction charge. Please note that when the source product calls the Risk API separately, Oracle Payments cannot evaluate the risk scores associated with AVS (Address Verification System). Oracle Payments gets the AVS codes directly from the payment system during an authorization request. As no authorization request is sent in this scenario, Oracle Payments cannot get AVS codes from the payment system and hence cannot evaluate risks scores associated with AVS.

Risk management helps businesses in reducing manual operational overheads to handle bad transactions and in avoiding costly penalties such as chargebacks from banks.

Note: Oracle Payments does not support risk management for PINless debit card or bank account transfer transactions.

Risk Factors Shipped with Oracle Payments

The following is a list of basic risk factors shipped with the Risk Management component. These risk factors can be configured per payee.

- Payment amount limit is the amount involved in a payment request. It varies from business to business and the risk factor score can be configured for different amount ranges based on the business model.
- Time of purchase is the time that a payment request is made by the customer. Site administrators can define the time duration during which the payment requests are high risk and assign the risk factor scores for each duration.
- Ship to/bill to address is used to match the ship to address to the bill to address in a payment request. A payment request is considered high risk if these two addresses do not match.
- Risky payment instruments are a list of payment instruments (e.g, credit cards, bank accounts) that are considered risky by each payee. These include the instruments that were used by customers earlier and had resulted in fraud or chargebacks. Such a list can be generated internally by the payee or obtained from other sources. If these instruments are reused in a payment request, then the payee may again face fraud or chargeback. Risk management functionality can detect whether risky payment instruments are being used during processing by looking at the risky instrument repository. If the instrument being used for the payment is found in the repository, then the payment is considered a high risk payment. The Risky Instruments Upload Utility adds and deletes a list of risky instruments from the database.

- Transaction amount is the total amount of payments made by a customer using the same instrument in a specified duration of time. The duration of time is set up by the user. This is related to the payment amount limit risk factor. A customer can make payments in smaller amounts, which are not captured by the payment amount limit risk factor but can be captured by the transaction amount risk factor. Transaction amount risk factor sums up the total amount of payments in a specific duration of time and captures the risk on that amount. The total number of payments made during a specific time period can be checked by looking at the payment history. The site administrator can set up a time duration and a transaction amount. In evaluating this risk factor, if the total payment amount exceeds the transaction amount within the specified time duration, then the payment is considered a high risk payment.
- Payment history tracks the reliability of the payer involved in a payment request. If a payer has a good history of payments over a long duration, then payments requested by this payer are considered to be low risk payments.
- Address verification service (AVS) check is the risk involved on the AVS code that is returned by the credit card network. Address verification service is provided by MasterCard and Visa credit card networks to match the billing or shipping address with the address that is maintained for the cardholder by the issuing bank. Oracle Payments does address verification during an authorization request, by calling the payment system with the address and zip code information along with the payment transaction information. The payment system then does the authorization and also returns various AVS codes to Oracle Payments. Various AVS codes are returned based on the complete address match, zip code match, street address match, etc. A site administrator can configure all AVS codes returned by the payment systems and their corresponding risk factor scores. This service is only provided in the United States of America.
- Frequency of purchase tracks the sudden surge in the use of a payment instrument in a short duration. For a particular payment instrument in a payment request, if the frequency of use in a duration configured is more than the setup value, then the payment request is considered to be a high risk payment.

Oracle Receivables Risk Factors

For customers who have both Oracle Payments and Oracle Receivables installed and registered, more risk factors are available. These risk factors are set up in Oracle Payments and the values of these risk factors are set up in Oracle Receivables. Oracle Receivables stores credit management information about customer accounts such as credit rating and risk rating. The following are risk factors used in risk analysis:

- Credit limit is an overall credit limit associated with a customer's account. If a customer has an outstanding balance and the total amount of payment made by the customer exceeds the overall credit limit, then the payment becomes a high risk payment. Overall credit limit varies from business to business. It can be set up as an

overall credit limit at the customer or site level through Oracle Receivables.

- Transaction credit limit is the credit limit per transaction associated with a customer's account. When a payment request exceeds the transaction credit limit, it becomes a risky payment. The transaction credit limit varies from business to business. It can be set up at the customer or site level through Oracle Receivables.
- Credit rating is the information that enables payees to effectively manage financial terms with their customers. It is useful for online financing or in evaluating purchases of a large amount by a new customer. Credit Rating is a user defined field and the information can be taken from Oracle Receivables. A payee associates risk scores to credit rating. A higher risk score implies that selling goods or services to the customer is risky.
- Risk code is a user defined risk assessment field in Oracle Receivables. It is useful for online financing or for evaluating purchases of a large amount for a new customer. The information is available from Oracle Receivables. A payee associates risk scores to all the risk codes. A higher risk score implies that selling goods or services to the customer is risky.

Oracle Payments Routing and Operation

Oracle Payments accepts funds capture payment transactions from source products and routes them to the appropriate payment systems using the appropriate payment system account and funds capture payment profile. Each payee can have its own set of routing rules with its own set of priorities.

What Constitutes a Routing Rule?

Every routing rule is made up of the following components:

- Basic Rule Information - This information is used to select and rank all the rules that may be applicable to a payment transaction. The basic rule information consists of Rule Name and Payment Method.
- Destination Information - The destination information specifies the payment system to which the payment transaction should be routed and how it should be sent. The destination information consists of Payment System Account and Funds Capture Process Profile.
- Routing Rule Conditions - This specifies the conditions under which a rule becomes applicable to a payment transaction. A rule condition is comprised of a criterion for the condition (such as Amount, Currency, Organization ID, Card Type, Card Number and Bank Routing Number), the type of operation related to the criterion, and the value of the criterion. Multiple rule conditions can be defined for a routing rule.

How Routing Works

Routing of a payment transaction is based on a set of routing rules set up in the Oracle Payments user interface by the Oracle Payments administrator. The routing engine finds the appropriate Payment System Account and Funds Capture Process Profile, and through them, the Payment System in the following sequence:

1. The routing engine retrieves the rules associated with the Payee and Payment Method specified in the payment request.
2. The routing rule with the highest priority is evaluated first. If the values in the transaction match the conditions specified in the routing rule, the request is routed to the corresponding Payment System using the specified Payment System Account and Funds Capture Process Profile.
3. If the values in the request do not match the conditions specified, the routing rule with the next highest priority is evaluated.
4. In case the payment request values do not match any of the conditions specified, the transaction is routed to the default Payment System using the default Payment System Account and Funds Capture Process Profile.

Routing rules are prioritized by an administrator. During processing, the rules are evaluated in the order in which they are prioritized.

Oracle Payments supports credit cards, purchase cards and bank account transfers. The payment methods available depend on the payment system that you decide to use.

Payees and businesses can customize how Oracle Payments routes transactions to the payment systems using routing rules based on their business rules and the available payment methods. For example:

- A business sends all electronic payment transactions to a single payment system: Payment System A.
- A business sends all small or micropayment transactions to Payment System A and all credit card transactions to Payment System B.
- A business sends all bank account transfers under \$10 to Payment System A, and all other transactions to Payment system B.
- A business sends all transactions using US dollars to Payment System A and all transactions using other currencies to Payment System B.

Routing Rule Conditions

Routing rule conditions determine whether the rule is applicable to a payment transaction. A rule can have multiple rule conditions. A rule is applicable to a payment

transaction only if the payment transaction can meet all the conditions for the rule. For example, a payee can route all Visa credit card transactions when the order amount is greater than 500 US dollars to Payment System C.

The following table lists the values in the Operation and Value fields for a selected payment instrument type and criterion.

Payment Instrument Type	Criterion	Operation	Value
Credit Card	Card Issuer	Equal, Not Equal To	Select a card issuer from the list of values.
Credit Card	Currency	Equal, Not Equal To	Select a currency from the list of values.
Credit Card	Amount	Greater than, Greater than or equal to, Less than, Less than or equal to	Specify a value.
Credit Card	Card Number	Equal, Not Equal To	* Specify a value.
PINless Debit Card	Card Issuer	Equal, Not Equal To	Select a card issuer from the list of values.
PINless Debit Card	Currency	Equal, Not Equal To	Select a currency from the list of values.
PINless Debit Card	Amount	Greater than, Greater than or equal to, Less than, Less than or equal to	Specify a value.
PINless Debit Card	Card Number	Equal, Not Equal To	* Specify a value.
Bank Account Transfer	Payer Bank Routing Number	Equal, Not Equal To	Specify a value.
Bank Account Transfer	Currency	Equal, Not Equal To	Select a currency.

Payment Instrument Type	Criterion	Operation	Value
Bank Account Transfer	Amount	Greater than, Greater than or equal to, Less than, Less than or equal to	Specify a value.

* - Value can be digits, spaces, dashes and wild card character (%). For example, if the value is 4111%, then the routing rule applies to all card numbers that begin with 4111.

Understanding Transaction Reporting

Oracle provides management summaries directly from the transaction data. Transaction reporting (TR) users can view indicators on a daily, weekly, or monthly basis, targeted to their particular lines of business and summarized across all processors, types of cards, and transaction types.

Transaction reporting lets every manager in an organization of any size to know the state of business transacted on credit and purchase cards on a daily basis, and to make mid-course corrections that drive the business towards achieving its goals. Transaction reporting helps enterprises achieve consistent, high-integrity information, corporate wide alignment, and collaborative decision making. A proactive e-mail notification system relieves the burden of constantly monitoring critical measures. Through transaction reporting, Oracle Payments provides an environment that supports mixed workloads, such as processing transactions versus running queries, without compromising on performance or scalability, providing the simplest, and therefore the least costly approach.

Functioning of the E-mail Push System

The Oracle Payments e-mail push system provides the ability to send e-mail notifications to specified users which frees mail from the need to monitor critical measures.

Any user with the Oracle Payments Daily Business Close User responsibility can run the e-mail push system by submitting a concurrent request. You can configure the Oracle Payments e-mail push system to send e-mails on a pre-defined schedule. The reports provide a daily summary of transactions. For best results, schedule the process to run at the close of business everyday. Specify the recipients for the e-mail notification by providing the e-mail ID or user names as parameters for the concurrent task. The user names must have valid e-mail IDs associated with them. Specify multiple recipients for a notification by separating the e-mail IDs or user names by a comma (',').

The e-mail push system sends the following information to the receiver after summarizing the transactions for that day.

- Login URL
- Date that the report was generated
- Total number of transactions
- Total transaction amount
- Total number of authorization transactions
- Total authorization amount
- Total number of settled transactions
- Total captured amount
- Total number of credit/return transactions
- Total credit/return amount
- Total number of credit card transactions
- Total credit card transactions amount
- Total number of purchase card transactions
- Total purchase card transactions amount

Scheduling E-mail Push Programs

To schedule concurrent requests for e-mail push programs:

1. Log in to Self Service Applications as any user with the Oracle Payments Daily Business Close User responsibility.

Choose the Oracle Payments Daily Business Close User if the user has multiple responsibilities linked to the user name.
2. Navigate to the Submit a New Request window.
3. Select the Single Request option.
4. From the Name choice list, select IBY Push E-mail Report.
5. Specify the recipient e-mail address. For multiple addresses, use the comma (',') as a separator.
6. To define a schedule, click Schedule.

The Schedule window appears.

7. Define the schedule.

Defining a schedule can be as simple as submitting as soon as possible or using a more complex schedule.

8. Click Submit.

Understanding Funds Disbursement

This section presents functionality and terms that are relevant to the funds disbursement process.

Understanding Documents Payable

A document payable is a transaction in a source product that is sent to Oracle Payments for payment. An example of a document payable in Oracle Payables is an invoice. During the payment process, documents payable are grouped together into actual payments.

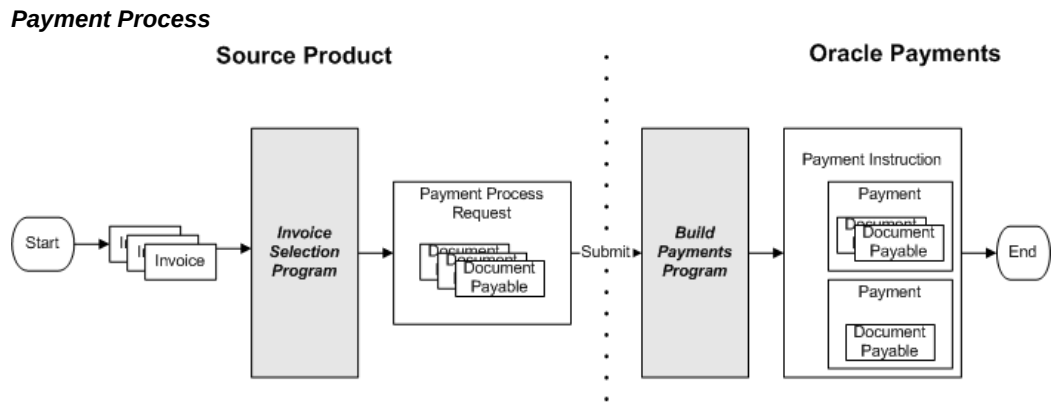
Understanding Payments

A payment is a single transfer of funds from one person or organization to another via printed payment document or electronic transmission.

Payment Process

The payment process starts when a source product, such as Payables, needs to pay documents payable, such as invoices. The source product user groups the documents payable into a payment process request and submits the request to Oracle Payments. Within Oracle Payments, the Build Payments program takes the submitted documents payable, validates them, groups them into payments, and then validates the payments. Each payment consists of one or more documents payable. Next, the Create Payment Instructions program groups the payments into payment instructions, and then validates the payment instructions. Each payment represents a check that will be printed or a single electronic funds transfer transaction, depending on the type of processing selected. Each payment instruction results in one payment file that contains pertinent information on one or more payments, such as payment amount and an account to be credited or debited. In the case of electronic payments, the payment file includes the payment instructions in a format required by the financial institution. To actually make payments, you print checks or electronically transmit the payment instruction to an external payment system or to a financial institution.

The figure below depicts the high-level payment process.



Understanding Payment Methods (Funds Disbursement)

On the funds disbursement side of Oracle Payments, a payment method is a payment attribute on a document payable. The payment method indicates the medium by which a first party payer makes a payment to a third party payee. Payment methods also include other information used during the early stages of payment processing, such as validations and rules that determine how payment methods can be assigned to documents payable. Examples of funds disbursement payment methods include the following:

- checks printed in-house by the payer
- checks outsourced to the bank for printing
- electronic funds transfers, through ACH in the United States or BACS in the United Kingdom
- wires

Oracle Payments seeds some payment methods, but also enables you to define your own as well.

A source product user, such as an Oracle Payables associate, must select a payment method when entering a document payable, such as an invoice. The source product uses Oracle Payments setup to default payment methods onto each document payable and to restrict the user's choice of payment methods to encourage efficient payment processing.

Understanding Payment Process Profiles

A payment process profile is a payment attribute assigned to documents payable, which specifies handling of the documents payable, payments, and payment instructions by Oracle Payments. Payment process profiles include several types of information,

including specifications for payment instruction formatting and transmission.

Payment process profiles can be assigned to documents payable either by the source product user or by the Oracle Payments payment administrator. The selection of valid payment process profiles is determined by the payment process profile's usage rules, which are created in Oracle Payments setup. Usage rules for payment process profiles can be based on payment method, payment currency, first party organization, or internal bank account.

Payments are built from documents payable that have the same payment process profile, among other attributes. Payment instructions are built from payments that have the same payment process profile, among other attributes. Therefore, the payment process profile is available to specify Oracle Payments behavior at every step of the payment process.

When you create a payment process profile, you must specify whether the profile governs payment processing for printed or electronic payments. Accordingly, the payment process profile allows you to assign appropriate payment document printing values or payment system communication configurations. The payment process profile also includes a payment instruction format, which is in turn associated with an XML Publisher template, as well as rules for grouping documents payable into payments and payments into payment instructions.

The relationship between payment process profiles and payment methods is many-to-many. For example, you can define one generic payment method like Electronic, and map it to any number of specific payment process profiles like: US Automated Clearing House and German EFT. Alternatively, you can set up specific payment methods that map one-to-one with payment process profiles. Additionally, you can define one payment process profile for multiple payment methods. This may be desirable when multiple payment methods can be handled by the same payment format. For example, the Germany Domestic profile can handle Outsourced Checks and EFT Payments payment methods.

For positive pay and electronic payment processing, payment process profiles are mapped to payment systems, payment system accounts, and transmission configurations.

Oracle Payments seeds a limited number of payment process profiles.

Understanding Payment Process Requests

A payment process request is a request created by a source product for Oracle Payments payment services. The payment process request, which originates in the source product during the documents payable selection process, contains one or more documents payable to be paid, along with information that allows Oracle Payments and the source product to identify the request and optional payment processing instructions. The source product may submit payment process requests to Oracle Payments via user action or concurrent program.

Once the payment process request is submitted to Oracle Payments, the Build Payments

program validates the documents payable, groups the documents payable into payments according to document attributes and payment process profile grouping rules, and then validates the payments.

Understanding Payment Grouping

Payment grouping is set up using check boxes in the Payment Grouping region of the Create Payment Process Profile page. Selection of payment grouping attributes specifies that only payments with like payment grouping attributes will be included in a payment instruction when that Payment Process Profile is used during the Create Payment Instructions program.

Understanding Payment Instructions

A payment instruction contains one or more payments, along with aggregate payment information, and is created by running the Create Payment Instructions program. Depending on your setup, a payment instruction can be converted into a payment file to be printed onto payment documents, such as checks, or into a payment file that is transmitted to a payment system or financial institution for further processing and disbursement. To see the relationship between payment instructions and payments, see Payment Process, page 3-27.

Each payment instruction that is electronically transmitted to a payment system or financial institution is associated with a payment file. This payment file contains data that instructs the payment system or financial institution how to make the payment. The following information is typically included in electronically transmitted payment files:

- number of payments to be made
- amount of each payment
- first party payer and third party payee bank account information
- name of payees

The following information is typically included in printed payment files:

- amount of each payment
- numbering of checks
- name of payees

Understanding Validations

Overview

In payment processing, it is critical to ensure that payment files sent to payment systems and financial institutions are valid, in addition to being correctly formatted. If this is not done, the payment process is slowed, which results in additional time and cost due to problem resolution. To help achieve straight-through processing, Oracle Payments enables you to ensure that payment-related validations are present.

Oracle Payments provides seeded validations that are associated with the supported payment formats by default. Validations are implemented using a flexible framework that enables you to assign new validation rules. You can choose between using the seeded library of validations, using your newly added validations, or using some combination of these rules.

Some of the system validations are:

- Payee bank account must be active.
- Checks whether the payment is resulting in international payment. If it is an international payment, then checks whether the "Allow international Payment" is selected at the Payee bank account level.
- Checks whether the payment currency is same as the payee bank account currency. If it is not, then checks if it supports multi currency.

The default validation rules are business rules that are followed in the business. If you look at the validation rules mentioned, most of them are general standard practices followed in the business. There is no specific document on the default validations and we cannot document all the general business rules also.

Additionally, you have choices with respect to the timing of validation rule execution. Validations can be assigned toward the beginning or toward the end of the payment process. A combination of rule assignments can also be used. You can adapt validation rule assignment to support your business model. For example, assigning and executing validations toward the beginning of the payment process may be best for a decentralized payment environment, whereas assigning and executing validations toward the end of the payment process may be better in a shared service environment, where payment specialists can resolve validation failures.

Seeded validations can be assigned in one the following places:

- Create Payment Method page--funds capture side
- Create Payment Method page--funds disbursement side
- Create/Update Format page

Validations are comprised of the following categories:

- Pre-defined Validations
- User-defined Validations

Country-Specific Validations

A country-specific validation is a validation related to a specific country. It is comprised of a code and multiple sub-validations that are required for the specific country, a validation point, and additional specifications regarding the payment formats and/or payment methods to which the validations are applicable. Country-specific validations are seeded by Oracle Payments and cannot be modified. For a listing of Oracle Payments seeded, country-specific validations, see Country-Specific Validations, page B-3.

The table below describes the elements of a country-specific validation.

Elements of a Country-Specific Validation

Element	Description
Code	The internal code for the validation.
Sub-Validations	All the validations that are executed in a certain validation.

Element	Description
Validation Point	<p>The point at which the validation is executed in the payment process. Validations occur at the following points in the payment process:</p> <ul style="list-style-type: none"> Document Level in the Source Product: The validation is executed early in the process while the documents are created in the source product. The validation is run by the source product user while he/she, for example, is entering invoices in Payables. Example of a validation run at the document level in the source product: Assign validations to the payment method on the invoice Document Level in Oracle Payments: The validation is automatically executed late in the process by the Build Payments program as the documents are grouped into payments. Example of a validation run at the document level in Oracle Payments: Assign validations to the payment format. Payment Level in Oracle Payments: The validation is automatically executed late in the process by the Build Payments program as the documents are grouped into payments. This occurs when the validation is set on a field, such as the payment amount. Example of a validation run at the payment level in Oracle Payments: Assign validations to the payment amount on the payment. Payment Instruction Level in Oracle Payments: The validation is automatically executed late in the process by the Create Payment Instruction program as the payments are grouped into payment instructions. This occurs when the validation is set on a field, such

Element	Description
	as the payment instruction total amount. Example of a validation run at the payment instruction level in Oracle Payments: Assign validations to the total payment amount on the payment instruction.
Entity to which the Validation is Applicable	The entity to which the validation is linked. Oracle Payments enables you to control, through the user interface, which payment method or payment format will execute which validations. Certain applicability links are seeded by default. You can, however, change these applicabilities. For example, you can specify that you want a certain validation that is linked to a certain payment format to be applicable to another payment format as well.

User-Defined Validations

User-defined validations are basic validations that correspond to simple operations. For example, validating that the length of a certain field does not exceed a specific character limit is an example of a user-defined validation. These validations can be used as components, or building blocks, to build more complex validations. They enable you to validate, for example, the following conditions:

- length of a field
- whether a field is populated
- whether a field contains non-numerical characters

To illustrate the application of user-defined validations, you can create a new payment format, for example, in the Create Format setup page and then assign validations to the newly created format.

Note: The user-defined validations are executed at the same points as the country-specific validations. For information on validation points, see the Validation Point, page 3-32, in the Elements of a Country-Specific Validation table.

Shared Setup Tasks for Funds Capture and Funds Disbursement

Overview

Both the funds capture and funds disbursement processes have setup tasks in common. The setup tasks described in this section are shared by both funds capture and funds disbursement and represent the first six setup tasks for Payments. Since both funds capture and funds disbursement use these same setup tasks, these shared setup steps must always be performed, whether you are using only the funds capture features of Payments, only the funds disbursement features, or both.

Recommended Setup Steps

Oracle Payments recommends that you perform the following optional setup steps before setting up other E-Business Suite products or Oracle Payments:

- Setting Up Payment Function Access
- Setting Up Function Security

Setting Up Payment Function Access

A payment function is an attribute on a document payable that indicates the purpose of payment. Examples of payment functions include supplier payments, employee expense payments, and loan payments.

In addition to identifying the purpose of documents payable, payment functions also serve to control the user's ability to process certain documents payable. That is, you can only take actions on payment process requests and payment instructions when all the documents payable and payments within have payment functions for which you have permission.

Payment functions are enabled or disabled in the Oracle Payments Funds Disbursement

Process Home page menu. For details on how to add or remove functions from menus, see the *Oracle E-Business Suite System Administrator's Guide*.

Setting Up Function Security

Oracle Payments enables you to perform a variety of sensitive tasks. You can use function security to allow users to perform or disallow them from performing the following tasks:

- update Oracle Payments setup
- retry funds capture transactions
- void funds disbursement payments
- stop funds disbursement payments
- view sensitive employee data
- submit separate remittance advice

Setup Checklist for E-Business Suite Products

To use Oracle Payments, you must perform the setup steps indicated in the table below in other E-Business Suite products. For each application installed, consult the guides for that application to determine the sequence of setup steps.

Setup Checklist for E-Business Suite Products

Setup Step	Step Type	Oracle Application and Applicable Guide
Set up internal bank accounts and related payment documents.	required	Oracle Cash Management, Bank and Account Administration: Define Bank Accounts, <i>Oracle Cash Management User Guide</i> .

Setup Step	Step Type	Oracle Application and Applicable Guide
Set up bill-to location and operating units.	required	Legal Entity Configurator, <i>Oracle Financials Implementation Guide</i> . See Legal Associations, <i>Oracle Financials Implementation Guide</i> , Creating Establishments, <i>Oracle Financials Implementation Guide</i> , and Updating Establishments, <i>Oracle Financials Implementation Guide</i> .
Set up external bank accounts.	conditionally required Note: External bank accounts are supplier bank accounts. This step is required if the Payments user and/or the Payables user wants to transfer funds electronically to supplier bank accounts. This step is not required if the Payments user and/or the Payables user only wants to print checks.	Oracle Cash Management, Creating Banking Details, <i>Oracle iSupplier Portal User's Guide</i> , <i>Oracle iSupplier Portal User's Guide</i> .
Set up a tax reporting entity.	required	Oracle Payables, <i>Oracle Payables User Guide</i> . See Reporting Entities, <i>Oracle Payables Implementation Guide</i> .
Define conversion rates.	conditionally required when using multiple currencies	Oracle General Ledger, <i>Oracle General Ledger User's Guide</i> . See Defining Conversion RateTypes, <i>Oracle General Ledger User's Guide</i> .

Prerequisites

Before you can setup Oracle Payments, you must perform following prerequisite setup

steps:

- Set Up Taxes, *Oracle E-Business Tax User Guide*. See Setting Up Taxes, *Oracle E-Business Tax User Guide*.
- Set Up a Tax Registration, *Oracle E-Business Tax User Guide*. See Setting Up a Tax Registration, *Oracle E-Business Tax User Guide*.
- Set Up Legal Entities, Legal Entity Configurator, *Oracle Financials Implementation Guide*. See Creating a Legal Entity, *Oracle Financials Implementation Guide*.

Setup Checklist for Shared Oracle Payments Setup Tasks

The table below shows the Oracle Payments setup checklist for features used by both funds capture and funds disbursement. These shared setup steps should be setup in the sequence indicated in the table below.

Oracle Payments Setup Checklist for Features Used by Both Funds Capture and Funds Disbursement

Step Number	Setup Step	Step Type	Oracle Application
1.	Creating Oracle Payments Users	required	Application Object Library
2.	Setting Up System Security Options	optional	Oracle Payments
3.	Setting Up or Updating Oracle XML Publisher Templates for Payment Formats or Reporting Formats. Note: Many such formats are seeded in Oracle XML Publisher.	conditionally required	Oracle XML Publisher
4.	Setting Up Formats	conditionally required	Oracle Payments

Step Number	Setup Step	Step Type	Oracle Application
5.	Setting Up Validations	optional	Oracle Payments
6.	Setting Up Transmission Configurations	conditionally required	Oracle Payments
7.	Configuring Tunneling	conditionally required	Oracle Payments
8.	Setting Up Payment Systems	conditionally required	Oracle Payments
9.	Configuring Oracle Payments Sample Servlet	optional	Oracle Payments
10.	Setting Up SSL Security for Payment System Servlet Communication	optional	Oracle Payments
11.	Configuring the XML Framework	required	Oracle Payments

Step 1. Creating Oracle Payments Users

The table below shows the main user interfaces that Oracle Payments provides that serve different purposes, along with their corresponding responsibilities.

Oracle Payments User Interfaces with Corresponding Responsibilities

Main User Interfaces	Corresponding Responsibilities
Funds Capture Setup	Funds Capture Setup Administrator
Funds Disbursement Setup	Funds Disbursement Setup Administrator
Funds Capture Process Dashboard	Funds Capture Process Manager

Main User Interfaces	Corresponding Responsibilities
Funds Disbursement Process Dashboard	Funds Disbursement Process Manager
Credit Card Processing Analytics	Credit Card Processing Analytics

A user can be assigned multiple responsibilities and roles.

Note: To access Credit Card Processing Analytics, login using the JTF (Java Technology Foundation) user interface.

Assigning Responsibilities and Roles to an Oracle Payments User

You can assign multiple responsibilities and roles to an existing user or to a new user. To create a user, see Step 1. Creating Oracle Payments Users, page 4-6.

The table below lists seeded Oracle Payments responsibilities and their description.

Seeded Oracle Payments Responsibilities and their Description

Responsibility	Description
Payments Setup Administrator	Setup for Shared, Funds Capture, and Funds Disbursement Note: The Payments Setup Administrator responsibility is used only if one user sets up the three functionality pieces listed above.
Funds Capture Setup Administrator	Setup for Shared and Funds Capture
Funds Disbursement Setup Administrator	Setup for Shared and Funds Disbursement
Funds Capture Process Manager	Funds Capture Process Dashboard
Funds Disbursement Process Manager	Funds Disbursement Process Dashboard
Credit Card Processing Analytics	Credit card Processing Analytics

If a user is assigned the Credit Card Processing Analytics responsibility, you must

assign the role indicated in the table below to the user with its corresponding permissions.

Credit Card Processing Analytics Responsibility and its Corresponding Required Role and Permissions

Responsibility	Role	Permissions
Credit Card Processing Analytics	IBY_DBC_ROLE (specific to JTF technology stack)	IBY_DBC_ROLE IBY_DBC_VIEW_PERMISSIO N

Step 2. Setting Up System Security Options

System security options enable you to set security options for payment instrument encryption, masking, and credit card control. These options are used for both funds capture and funds disbursement processes. Payments uses the settings to handle security issues, such as encrypting payment instrument sensitive data, payment instrument masking, and credit card owner verification controls.

For payment instrument encryption, Payments uses a chained key approach. To simplify, the chained key approach is where A encrypts B and B encrypts C. In Oracle Payments, the system key encrypts the subkeys and the subkeys encrypt the payment instrument data. This approach allows easier rotation of the system key. The system key is the encryption master key for the entire installation. It is stored in a wallet file and is used to encrypt Oracle Payments subkeys.

Prerequisite

Before you can set up security options, you must set up a wallet. To set up a wallet, see *Setting Up the Wallet*, page 4-7.

Setting Up the Wallet

Payments performs system key management using features from Oracle Wallet Manager.

The wallet is a file, which stores the system key. The contents of the wallet file are managed by Oracle Wallet Manager. The wallet file has two functions:

- to perform HTTP client authentication of your middle-tier server for payment systems that require this level of security

When used for client authentication, the wallet contains the private key of the entity authorized to send transactions to the payment system (usually the counterpart to the middle-tier server's public certificate). This is sensitive data and, depending on

how much trust is placed in the ability to authenticate as the certificate's subject, potentially damaging if compromised.

- to store the system (master) security key used to encrypt sensitive data
- Storing the system key in the wallet file provides greater security for the encrypted payment instrument data since the system key resides outside the Oracle Payments database. As this key is used to secure such data as credit card numbers, compromise of the wallet is *highly damaging*.

The purpose of setting up the wallet in the Wallet Setup page is to:

- specify the location of the wallet file
- define the password for the wallet file
- specify whether to generate the system key yourself or let the system do it

Creating a Wallet File

To create a wallet file, you must start the Oracle Wallet Manager program. On UNIX systems this is done with the following command:

```
owm
```

If the wallet will contain only the system security key, it is sufficient to create an empty wallet file. If the wallet is to contain a private key for client authentication, it must be imported here. Once the wallet file is accessible to the middle-tier server, it is initialized with the system security key using the following Oracle Payments navigation: Oracle Payments Setup > System Security Options. You have the option of importing your own 24-bit system security key (stored in a binary file whose location is specified through the user interface) or you can generate a random one. Once the wallet setup process is complete, a system security key exists in the wallet, and a passwordless version of the wallet named cwallet.sso is created in the same directory as the original wallet file.

Defining the Wallet File Password

To define the password for the wallet file in the Wallet Setup page, enter any string. This password is used to encrypt the wallet file.

Specifying or Generating the System Key File Location

In the Wallet Setup page, you can provide the system key by specifying the location of the system key file or you can let the system generate the system key for you. In either case, the specified or generated key is put into the wallet file and encrypted with the password you provide.

Encrypting Payment Instruments

In the System Security Options setup page, you specify whether you want to enable or

disable encryption of payment instruments and whether you wish the encryption to occur immediately when new payment instruments are registered or be performed on a regularly scheduled basis for performance reasons.

Masking Payment Instruments

In the System Security Options setup page, credit card numbers and external bank account numbers can be masked by selecting the number of digits to mask and display. For example, a credit card number of XXXXXXXX represents a display of the last four digits and a masking of all the rest. These settings specify masking for payment instrument numbers in the user interfaces of many applications.

Verifying Credit Card Owners

This option enables you to require users to enter the credit card security code and/or credit card statement billing address. This information is passed to the payment system, which in turn, checks with the credit card issuer to confirm the credit card owner's security code and/or statement billing address.

Step 3. Setting Up Oracle XML Publisher Templates

Payments uses Oracle XML Publisher templates to correctly format funds capture and funds disbursement transactions and to enable you to easily manage the formats. These payment templates can be created new or modified with minimal effort by using a standard text editor, such as Microsoft Word. Consequently, when a payment system or financial institution requires a change to its payment instruction format, the change can be made quickly by modifying the appropriate Oracle XML Publisher template.

Special consideration has been given by Payments to the complexity of creating fixed position and delimited formats. Oracle XML Publisher's eText feature is used for these format types. eText allows the format layout to be presented in an understandable tabular structure.

Several payment-related templates are provided out of the box. You may want to use or modify an existing template to meet your format requirements.

Purpose

The purpose of setting up Oracle XML Publisher's templates is to create and register templates in Oracle XML Publisher. These templates are required by Oracle Payments to format payment instructions.

Related Topics

For detailed information on creating, registering, and using Oracle XML Publisher's formatting templates, see *Oracle XML Publisher User's Guide*.

Step 4. Setting Up Formats

Financial institutions, payment systems, and/or countries have specific formatting requirements for funds capture transactions, funds disbursement transactions, payment documents, and payment-related reporting. Formats are created within Oracle Payments to represent these requirements. Each format in Oracle Payments corresponds to one Oracle XML Publisher template. Oracle Payments uses Oracle XML Publisher templates to format funds capture and funds disbursement transactions according to the formatting requirements of specific financial institutions or payment documents.

In the Create Format setup page, formats are associated with specific Oracle XML Publisher templates and can also be assigned validation sets to validate transactions that use that format. Multiple formats can be used for a single payment system.

One format is provided out of the box for each of the payment-related templates in Oracle XML Publisher.

Prerequisites

Before you can set up formats, you must perform the following setup step:

- Oracle XML Publisher templates

Purpose

The purpose of setting up formats is to define formats within Oracle Payments, associate them with your XML Publisher templates, and assign validation sets.

Note: Before you can create a format in Payments, the corresponding XML Publisher template must be available. To see if the corresponding XML Publisher template is available, you can search for it in the Search region of the XML Publisher templates setup page.

Step 5. Setting Up Validations

Validations ensure that funds capture and funds disbursement transactions are valid, in addition to being correctly formatted before they are printed or submitted to payment systems.

In the Validations page, you can view seeded validations. For each validation, you can view the parameters of the validation and the formats and payment methods to which it is assigned.

You can assign pre-defined or user-defined validations to a format of type Disbursement Payment Instruction or to a funds disbursement payment method. You

can assign pre-defined validations to a format of type Funds Capture Settlement Batch.

Related Topics

For detailed information on validations, see *Understanding Validations*, page 3-30.

For information on formats, see *Step 4. Setting Up Formats*, page 4-10.

For information on funds disbursement payment methods, see *Step 12. Setting Up Funds Disbursement Payment Methods*, page 5-2.

Step 6. Setting Up Transmission Configurations

A transmission configuration implements a specific transmission protocol, which allows the delivery of a transaction to a specific payment system or financial institution.

Each transmission protocol has parameters that require values. The values defined for the parameters comprise the transmission configuration for that transmission protocol. For example, the payment system, PaymentTech, may require a Socket IP Address of X and a Socket Port Number of Y. Together, X and Y represent Transmission Configuration A for a given transmission protocol.

Purpose

The purpose of setting up transmission configurations in the Create Transmission Configuration setup page is to enable electronic connectivity with payment systems by specifying parameter values.

Related Topics

For additional information on transmission protocols, see *Understanding Transmission*, page 3-4.

Step 7. Configuring Tunneling

Tunneling is a transmission feature in Oracle Payments whereby data, such as a payment instruction, is delivered using two protocols, one of which encapsulates the other. Tunneling is also referred to as delegated transmission, since the initial transmission from Oracle Payments is a request to an external module, that is, the Transmission Servlet, to deliver data using an independent transmission protocol. The name of the transmission protocol, its parameters, and the actual data which must be delivered by it are encapsulated within the body of the tunneling transmission protocol.

The purpose of tunneling is to allow connectivity between Oracle Payments and external payment systems without compromising network security. Processor payment systems, for example, often require protocols, such as FTP or raw IP socket connectivity to receive payment instruction files. Instead of creating breaches in their firewall to

accommodate these connectivity requirements, users can instead deploy the Payments Transmission servlet on a host outside their firewall and then tunnel, or delegate, requests to it from the Oracle Payments engine. The Oracle Payments Transmission Servlet, by design, makes no use of the applications database, and can be completely isolated from the deployer's intranet.

Tunneling Protocol

Oracle Payments uses a customized tunneling protocol called the Oracle Payments Tunneling Protocol (code= IBY_DELIVERY_ENVELOPE). This protocol uses HTTP POST as its underlying transmission mechanism (HTTPS is supported as well) and sends within the body of the request an XML message header identifying the tunneled or encapsulated protocol, as well as the parameters to use when invoking it, such as host name, user name, and password for FTP. After the XML message header is the data to be delivered.

As a transmission protocol code-point, the Oracle Payments Tunneling Protocol implements the `oracle.apps.iby.net.TunnelingFunction` interface.

The table below presents the parameters and descriptions of the Oracle Payments Tunneling Protocol.

Parameters and Descriptions of the Oracle Payments Tunneling Protocol

Parameter	Description
WEB_URL	the HTTP/HTTPS URL of the Transmission Servlet executing the protocol
USERNAME/PASSWORD	the username and password used to access the servlet if its URL is secured by HTTP authentication

Transmission Servlet

The Oracle Payments Transmission Servlet is the module which executes tunneled or delegated transmission requests sent from the Oracle Payments engine. The servlet receives Payments HTTP XML Delivery Envelope requests and parses them into XML message header and transmission data components. The format of the XML message header is defined by an XML DTD file named `DeliveryEnvelope.dtd`. This message header specifies both the parameters to pass to the tunneled/encapsulated transmission protocol, as well as the transmission protocol, itself, by means of its Java code-point class name and entry function name. The Transmission Servlet then attempts to dynamically load the Java class implementing the tunneled protocol and invokes it by passing to it the transmission parameters parsed from the XML message header, as well

as the transmission data. This behavior is identical to that of the Oracle Payments engine. Any protocol can be tunneled, as long as it implements the `oracle.apps.iby.net.TransmitFunction` interface. Therefore, any custom-defined protocol can be tunneled or encapsulated to the servlet, provided the Java class which implements its code-point is in the CLASSPATH of the servlet's application container.

Class `oracle.apps.iby.bep.TransmitServlet` implements the Oracle Payments Transmission Servlet and is aliased to URL `/OA_HTML/ibytransmit` on the middle-tier iAS application server. To relocate the servlet to a different host, such as the one in a DMZ network zone, the user must copy all the Applications-specific Java files from the E-Business Suite installation to a class repository accessible by the Transmission Servlet's new servlet container. Any new transmission protocols that you develop must have their Java code copied to this repository if you want the servlet to support that protocol.

Configuring Tunneling

Tunneling is configured through the Oracle Payments transmission configuration user interface. A tunneling transmission configuration is specified as any other transmission configuration, but the protocol is always the Payments HTTP XML Delivery Envelope protocol. Once the tunneling protocol is configured, any regular, non-tunneling transmission configuration can use or be encapsulated by it, by specifying a value for the tunneling configuration field in the transmission configuration user interface.

Important: Oracle Payments does not support the tunneling or encapsulation of a tunneling protocol.

Step 8. Setting Up Payment Systems

A payment system is an organization that provides financial settlement services. Companies that deploy Payments typically choose payment systems to process their funds captures and, sometimes, their electronic funds disbursements payment instructions. A payment system can be the bank at which the deploying company has its bank accounts or it can be a third party processor that connects deploying companies with financial institutions. The latter is commonly the case for credit card processing. Payment systems are not required for printed funds disbursement payments, but may be required for related services, such as positive pay or other reporting.

Prerequisites

Before you can set up payment systems, you must perform the following setup steps:

- formats
- transmission configurations

Purpose

The purpose of setting up payment systems is to:

- define the external organizations that Oracle Payments collaborates with to process your funds capture and funds disbursement transactions
- define the deploying company's relationships with its payment systems

Specifying Supported Capabilities

When creating a payment system on the Create Payment System setup page, you:

- specify payment instruments the payment system will support on the funds capture side and/or the funds disbursement side
- assign payment instruction formats to the payment system
- assign transmission protocols to the payment

Settings Required by Payment System

In this region, you can define parameters that the payment system requires for each transaction or settlement batch. When you define payment system accounts, you provide the actual payment system-provided values for these parameters.

Payment Systems on the Funds Capture Side

On the funds capture side, payment systems play a central role in the creation of funds capture process profiles, since the creation of a funds capture process profile is payment system-specific. Funds capture process profiles specify how Payments processes settlements, including how the settlement is to be formatted and how it is to be transmitted.

After setting up and choosing the Update Account, you must then update the Payer notification values in the Funds Capture Process Profiles, the update happens successfully.

Payment Systems on the Funds Disbursement Side

On the funds disbursement side, the payment system plays a smaller role in the creation of payment process profiles, which are blueprints for payments that contain payment instruction formatting and transmission information, as well as payment grouping, payment limits, and payment sorting details.

Step 9. Configuring Oracle Payments Sample Servlet

The Oracle Payments sample servlet is a gateway model servlet that you can use to test your Oracle Payments implementation without having to register with a real payment system. The sample servlet only supports core Oracle Payments operations, such as authorization, capture, and return for credit cards.

You can use the sample servlet to test the integration between your source product and Oracle Payments. All transactions sent to the sample servlet should succeed, unless the transaction amount matches certain pre-set values, in which case an error is induced. You can use the integration to simulate error scenarios and test error handling in the calling source product.

The table below lists the pre-defined transaction amounts and their associated error codes.

Pre-defined Transaction Amounts and their Associated Error Codes

Transaction Amounts	Error Messages
1001	Communication error when contacting the gateway. Please try again.
1002	Given order id used for a previous transaction.
1004	A parameter to this transaction is either malformed or missing.
1005	Generic payment system error occurred. Please check error code.
1008	Transaction type is not valid or not supported for this merchant.
1016	Internal payment system failure. Please check error code.
1017	Account does not have sufficient funds to complete this transaction.
1019	Invalid credit card number/expiration date.
1020	Authorization declined.

Transaction Amounts	Error Messages
1021	Voice authorization code incorrect.

Installing the Sample Servlet

To configure the sample servlet, perform the following steps.

1. Add the alias statement to the configuration file of the servlet zone you wish the sample servlet to run as specified in the orion-web.xml file which is in the following directory:

```
$ORA_CONFIG_HOME/10.1.3/j2ee/oacore/application-  
deployments/oacore/html/
```

2. In the same configuration file, provide the following servlet zone-wide parameters listed in the table below, which are set by a statement of the form `servlet.default.initArgs=`.

Zone-Wide Parameters

Parameter	Example Value	Description
errorfile	tmp/error.log	Debug file used to write errors and stack traces.
debugfile	tmp/debug.log	Log file used to write debugging messages.
debug	true, false	Turns debugging on or off.

Configuring Sample Servlet as a Payment System

Once the sample servlet is installed and configured, the servlet must be added as a payment system so it can be used. Login to the Oracle Payments Setup Administrator responsibility and create a payment system for the sample servlet with the following values:

- Name: Sample Servlet
- Suffix: lop
- Payment System Type: Gateway

- Transmission Servlet Base URL: Example - `http://localhost:<port>/OA_HTML`, where <port> is the port of your installation
- Supported Payment Instrument: Credit Card

Note: The sample payment system already exists in the Oracle Payments setup. You only need to enter the Base URL

Adding a Payment System Account

For each payee that uses the sample servlet, enter any value for the payment system account name.

Testing the Sample Payment System

To test the sample payment system, create a transaction through the Transaction Testing pages, or from a source product. Verify that you have a routing rule that routes the transaction to the sample payment system and that your transaction matches the routing rule.

The \$0 authorization cannot be executed using the sample payment system.

Related Topics

For more information on setting up a routing rule for first party payees, see Step 17. Setting Up First Party Payees, page 6-9.

For more information on configuring Paymentech, see Configuring Paymentech on *MetaLink*, Note 405996.1.

For more information on configuring FDC North, see Configuring FDC North on *MetaLink*, Note 405999.1.

For more information on configuring Concord EFSnet, see Configuring Concord EFSnet on *MetaLink*, Note 406000.1.

For more information on configuring Verisign, see Configuring PayPal Payflow Gateway (formerly Verisign) on *MetaLink*, Note 406002.1.

For more information on configuring CyberSource, see Configuring CyberSource on *MetaLink*, Note 406003.1.

Step 10. Setting Up SSL Security for Payment System Servlet Communication

When Oracle Payments communicates with the payment system servlets, the information exchanged may be sensitive information such as credit card numbers. If the

communication is not secure, it poses a security risk.

The security risk increases under the following circumstances if:

- Oracle Payments and the payment system servlets are installed on separate machines.
- Oracle Payments is running outside your firewall.

Steps

1. To set up a payment system servlet with secured sockets layer, enable HTTPS on the middle-tier server where the servlet resides.
2. If there are no funds capture profiles defined yet for the payment system, change the BASE URL parameter of the payment system to use the https: protocol. Otherwise, change the URLs on any transmission configurations set up to be used with that payment system to contain https:.

Step 11. Configuring the XML Framework

Oracle Payments incorporates an XML framework, which enables it to communicate with payment systems using XML. The IBY: XML_BASE property and, optionally, the IBY: JAVA_XML_LOG property must have valid values. Both properties can be set by running AutoConfig.

Setup Tasks for Funds Disbursement

Overview

Funds disbursement is a payment sent from the first party payer, the deploying company, to the third party payee, such as suppliers. A payment can take an electronic form; such as EFT or wire, or a printed form such as a check.

To use Oracle Payments, you must perform the setup steps indicated in the table below in other E-Business Suite products. For each application installed, consult the guides for that application to determine the sequence of setup steps.

Setup Checklist for E-Business Suite Products

Setup Step	Step Type	Oracle Application and Applicable Guide
Set up third party payees and suppliers and supplier sites.	required	iSupplier Portal, <i>Oracle iSupplier Portal Implementation Guide</i> . See Supplier Definition and Maintenance and Defining Supplier Sites.
Set up VAT reporting.	conditionally required	Oracle Financials for countries in the EMEA region, <i>Oracle Financials for Europe User Guide</i> . See Setting Up VAT Reporting.

The Oracle Payments setup checklist for the funds disbursement process is shown in the table below. These setup steps must be performed in Oracle Payments in the sequence indicated.

Funds Disbursement Setup Checklist

Step Number	Setup Step	Step Type	Oracle Application
12.	Setting Up Funds Disbursement Payment Methods	conditionally required	Oracle Payments
13.	Setting Up Payment Method Defaulting Rules	optional	Oracle Payments
14.	Setting Up Bank Instruction Codes	optional	Oracle Payments
15.	Setting Up Delivery Channel Codes	optional	Oracle Payments
16.	Setting Up Payment Reason Codes	optional	Oracle Payments
17.	Setting Up Payment Process Profiles	required	Oracle Payments
18.	Setting Up Disbursement System Options	required	Oracle Payments

Step 12. Setting Up Funds Disbursement Payment Methods

Note: If you intend to use funds capture functionality, you must complete Steps 13 to 18 as described below. If you do not intend to use funds capture functionality, you do not need to perform Steps 13 to 18.

A funds disbursement payment method is a medium by which the first party payer, or deploying company, makes a payment to a third party payee, such as a supplier. You can use a payment method to pay one or more suppliers. Oracle Payments supports several payment methods for funds disbursement, including the following:

- Checks
- Electronic Funds Transfer (EFT)

- Bills Payable
- Wire

Purpose

The purpose of creating funds disbursement payment methods is to:

- define the payment methods you want to use to make payments
- define rules to default payment methods onto documents payable
- assign validations to be run on documents payable, payments, and payment instructions

Creating Funds Disbursement Payment Methods

Creating a funds disbursement payment method in Oracle Payments is comprised of the following tasks:

- entering general information
- creating usage rules
- assigning validations

Entering General Information--First Node

If, after completing the Create Payment Method: General page, you click the Review button, the system navigates to the Create Payment Method: Review page and skips the remaining tasks in creating a payment method. The system inserts, however, default values for usage rules and payment field validations. No validations are assigned for this payment method. The Create Payment Method: Review page reflects the default information when you navigate to it by clicking the Review button.

Entering Header Information

In the Code field of the Create Payment Method: General page, enter a user-defined code, which represents a shortname for the payment method.

Note: For some formats, the payment method field or equivalent must be populated from a list of possible values, defined by the payment system or country. Those lists may not correspond, one-to-one, with payment methods that are seeded in Oracle Payments, or with new payment methods that you create.

To correctly populate the payment method or equivalent field in the format, enter the

value of a newly created payment method, as it should appear in the payment file, in the Format Value Mapping field of the Create Payment Method: General page or the Update Payment Method page.

For example, if a formatting template uses a format value of WIRE, but you want to create several new wire payment methods that each have different validations, you can enter WIRE in the format mapping value for each, and that is the value that will appear in the payment file for all payment instructions that have those payment methods.

In the Anticipated Disbursement Float field, enter a value that represents the number of days after submission of the payment instruction until the funds leave the first party payer's account.

Specifying Bills Payable Payment Methods

Bills payable is a payment method involving the transfer of funds between bank accounts, where one party promises to pay another a specified amount on a specified date. If you wish to create a payment method that is used only for bills payable, you must enter a value in the Maturity Date Calculation field. This value represents the number of days to add to today's date to determine the bills payable due date.

Creating Usage Rules--Second Node

Usage rules specify when a payment method is available for use by source products for documents payable. When you create usage rules, you enable or disable payment methods for each product integrated with Oracle Payments. By defining or not defining conditions for each product with an enabled or disabled payment method, you can effectively provide different usage rules for different source products and change whether and when the payment method is available.

The Availability column on the Create Payment Method: Usage Rules page defaults an availability of Always, which means that the payment method for the applicable product is always available and it also implies that no conditions have been set. The value in the Availability column is read-only and reflects whether conditions have been defined for the applicable product on the Usage Rules: Update Conditions for <product name> page.

The payment method that the source product user sees in the source product, depends on the usage rules specified in the Create Payment Method: Usage Rules page.

The following table shows the relationship between the definition of payment method conditions and the availability of payment methods for source products on the Create Payment Method: Usage Rules page.

Relationship Between the Definition of Payment Method Conditions and the Availability of Payment Methods for Source Products

Availability Column Displays...	Conditions for the Payment Method	Enabled Check box
Always	No Conditions Set	Selected
Conditional	Conditions Exist	Selected
Never	No Conditions Set	Selected Enabled Check Box is Deselected

In the Usage Rules: Update Conditions for <Product Name> page, the First Party Organization subregion uses access control security so that when you click the Add button, you only see the first party organizations to which you have access. That is, you can only add first party organizations to which you have access.

If, after completing the Create Payment Method: Usage Rules page, you click the Review button, the system navigates to the Create Payment Method: Review page and skips the remaining tasks in creating a payment method. The system inserts, however, default values for payment field validations. No validations are assigned for this payment method. The Create Payment Method: Review page reflects the default information when you navigate to it by clicking the Review button.

Defining Usage Conditions

When the source product user is entering an invoice in Oracle Payables, for example, the source product user enters data for all the variables indicated on the Usage Rules: Update Conditions for <product name> page. This data includes the organization and the legal entity to which the invoice belongs, as well as the payment processing transaction types available for use on the payment method, such as customer refunds, loan funding, payables documents, employee expense reports, bank account transfers, and adhoc payments. All of the data you specify on the Usage Rules: Update Conditions for <product name> page is entered on the invoice in Oracle Payables before the Oracle Payables user selects the payment method.

Assigning Validations--Third Node

Assign the validations you want performed for each document, payment, or payment instruction that uses this payment method. For detailed information on validations, see Document Validation Flow (F5), *Oracle Payments User's Guide* or Payment Instruction Validation Failure Handling Flow (F9), *Oracle Payments User's Guide*.

Updating Payment Methods

To update payment methods, you can:

- update usage rules by removing or adding conditions
- assign or remove validations

Duplicating Funds Disbursement Payment Methods

To duplicate a payment method, click the applicable Duplicate icon in the Payment Methods page. This action copies the entire original payment method. You can then make changes to the duplicate and save it as a new funds disbursement payment method.

Step 13. Setting Up Payment Method Defaulting Rules

Payment method defaulting rules determine when payment methods default onto a document payable, such as an invoice. Various products are shown in the Payment Method Defaulting Rules page because you can have different defaulting rules for different products. A payment method defaults onto a document payable when all specified conditions are met. That is, values on the document payable, such as legal entity, organization, and payment type, must match the values for the defaulting rules' conditions for the applicable payment method to default onto the document payable.

Oracle Payments applies the rules in the user-specified priority. For example, if the first rule is a match, Oracle Payments stops and defaults that rule's corresponding payment method onto the invoice. Further, suppose you specify that the payment method for all documents processed by Oracle Payables is first, Check and second, EFT. In this case, if the conditions for Payment Method Check match those on the invoice, then Payment Method Check defaults onto the invoice. If the conditions for Payment Method Check do not match those on the invoice, then Oracle Payments determines whether the conditions for Payment Method EFT matches. If the conditions for Payment Method EFT match those on the invoice, then Payment Method EFT defaults onto the invoice.

Generally, the source product allows the user to override the default payment system manually.

Purpose

The purpose of setting up payment method defaulting rules is to create and maintain defaulting rules for when payment methods are to default on documents to be paid.

Creating Defaulting Rules

If the First Party Legal Entity, the First Party Organization, the Payment Processing

Transaction Type, the Currency, and the Payee Location on the defaulting rule all match the same values for those attributes on the invoice, then that payment method defaults onto the invoice.

Specifying First Party Legal Entities

The first party legal entity is the legal entity to which the invoice belongs.

Specifying First Party Organization Entities

The first party organization is the organization to which the invoice belongs.

First party organization uses access control security, so that when you click the Add button, you only see the first party organizations to which you have access. This means that you can only add first party organizations to which you have access.

Specifying Payment Processing Transaction Types

The Payment Processing Transaction Type list of values only displays the payment processing transaction types assigned to the source product for which you are updating or creating a rule. Payment processing transaction types includes the following payment types:

- customer refunds
- loan funding
- payable documents
- employee expense reports
- bank accounts transfers
- ad hoc payments

Reordering Defaulting Rules Priority

Typically, defaulting rules are reordered when:

- you create a new defaulting rule
- the deploying company's business process has changed

Step 14. Setting Up Bank Instruction Codes

Bank instruction codes are generally country-specific identifiers provided by a country's government or central bank. These codes provide the payment system or bank with additional details about how the country-specific payment is to be processed.

Purpose

The purpose of entering country-specific bank instruction codes required by a particular country's payment system or bank is to enable you to process payment instructions to the country-specific payment system or bank.

Seeded Bank Instruction Codes

Oracle Payments provides many seeded bank instruction codes. The Author column of the Bank Instruction Codes page displays Oracle Payments for seeded bank instruction codes and the applicable user name for user-defined codes.

Creating Bank Instruction Codes

To create a bank instruction code, enter the following:

- code obtained from the country's government or central bank
- format value obtained from the country's government or central bank
- country for which you are creating the instruction code

Step 15. Setting Up Delivery Channel Codes

Delivery channel codes are generally country-specific identifiers provided by a country's government or central bank. These codes provide the payment system or bank with additional details about how the country-specific payment is to be delivered to a payee.

Purpose

The purpose of entering country-specific delivery channel codes required by a particular country's payment system or central bank is to enable you to specify how payments are to be delivered to payees by the country-specific payment system or bank.

Seeded Delivery Channel Codes

Oracle Payments provides many seeded delivery channel codes. The Author column of the Delivery Channel Codes page displays Oracle Payments for seeded delivery channel codes and the applicable user name for user-defined codes.

Creating Delivery Channel Codes

To create a delivery channel code, enter the following:

- code obtained from the country's government or central bank

- format value obtained from the country's government or central bank
- country for which you are creating the delivery channel code

Step 16. Setting Up Payment Reason Codes

Payment reason codes are generally country-specific identifiers provided by a country's government or central bank. These codes provide the payment system or bank with additional details about the reason for the payment for regulatory reporting purposes.

Purpose

The purpose of entering country-specific payment reason codes required by a particular country's payment system or central bank is to enable you to specify the reason for the payment to the country-specific payment system or bank.

Seeded Payment Reason Codes

Oracle Payments provides many seeded payment reason codes. The Author column of the Payment Reason Codes page displays Oracle Payments for seeded payment reason codes and the applicable user name for user-defined codes.

Creating Payment Reason Codes

To create a payment reason code, enter the following:

- code obtained from the country's government or central bank
- format value obtained from the country's government or central bank
- country for which you are creating the payment reason code

Step 17. Setting Up Payment Process Profiles

A payment process profile is a payment attribute assigned to documents payable, which specifies how Oracle Payments performs processing. Payment process profiles are comprised of several types of payment processing information, including specifications for formatting and transmission.

Prerequisites

Before you can set up payment process profiles, you must perform the following setup steps:

- funds disbursement payment methods

- formats
- payment systems and transmission configurations, if you plan to use them

Purpose

The purpose of setting up payment process profiles is to specify the details of the payment process. Payment process profiles are blueprints that contain all the rules for creating and disbursing payments.

Creating Payment Process Profiles

The payment process profile provides a number of features that vary in complexity, some of which may not be needed for basic payment processing. Therefore, once you create a payment process profile by entering data in the Create Payment Process Profile page and applying it, you can search for the newly create profile and set up any additional features by using the update functionality.

After you finish entering data in the Create Payment Process Profile page, you can click the Save and Add Details button to navigate to the Update Payment Process Profile page. The initial data entered is saved to the database and the newly created Payment Process Profile is ready for use, if appropriate or you can add additional features later by using the update functionality.

Entering Header Information

Selecting a processing type of Electronic or Printed determines what fields you see in the header of the Create Payment Process Profile page.

The list of values for the Payment System field is based on the value selected from the Payment Instruction Format list of values. If you do not select an option from the Payment Instruction Format list of values, then the list of values for the Payment System field is blank.

If a payment system is not selected from the Payment System list of values, then the list of values for the Transmission Configuration field is empty. When a payment system is selected, the Transmission Configuration list of values displays the configurations that are linked to the payment system.

Specifying the Electronic Processing Type

If you select Electronic from the Processing Type drop-down box in the Create Payment Process Profile page, the following fields display in the header that are pertinent to electronic payment processing:

- Payment Completion Point drop-down box. This option indicates when a payment is marked complete. Payments can automatically be marked complete anytime from the time the payment instruction is formatted in Oracle Payments to when the

payment is transmitted to the payment system.

- **Allow Manual Setting of Payment Completion check box.** This setting enables payment administrators to mark payments complete manually in the Payment Processes region of the Funds Disbursement Process Home page.
- **Automatically Transmit Payment File after Formatting check box.** If selected, the payment file is automatically transmitted to the payment system or bank after it is formatted. If it is not checked, a payment administrator will have to initiate transmission from the Funds Disbursement Dashboard.

Specifying the Printed Processing Type

If you select Printed from the Processing Type drop-down box, the following fields display in the header that are pertinent to printed payment processing:

- **Default Payment Document list of values**

The list of values displayed for the Default Payment Document field are based on the value selected from the Payment Instruction Format list of values. If you do not make a selection from the Payment Instruction Format list of values, the list of values in the Default Payment Document field is blank.

- **Send to File radio button**
- **Send to Printer radio button**
- **Automatically Print after Formatting check box**

If selected, the Default Printer field is populated with the default printer assigned in Oracle Applications. You can change this value if you wish, but you must select a default printer from the Default Printer list of values.

Specifying Payment Process Profile Usage Rules

Payment process profile usage rules determine when payment process profiles can be assigned for use on documents payable. You can select the All radio button to indicate that all payment methods, first party organizations, internal bank accounts, and/or currencies apply to this payment process profile or you can select the Specify radio button to specify specific values for any of these categorizations that apply to this payment process profile.

First Party Organization uses access control security so that when you click the Specify radio button and then click the Add button, you will only see the first party organizations to which you have access. This means you can only add first party organizations for which you have access.

Specifying Payment Instruction Creation Rules

This region enables you to specify how payments are grouped into payment instructions and how those payments are sorted within the payment instructions.

Internal Bank Account and Payment Currency payment grouping options are only displayed under the Payment Instruction Creation Rules region when the Processing Type selected is Electronic.

When the Payment Process Request grouping option is specified, payments that were submitted to Oracle Payments in different payment process requests are not mixed together in the payment instructions. However, there is no assurance that only a single payment instruction is created with payments from within a payment process request because payments within the same payment process request may contain different payment process profiles, and therefore need to be grouped into separate payment instructions.

Specifying Payment Grouping

Descriptions of selected payment grouping options are indicated in the table below.

Descriptions of Payment Grouping Options

Field	Feature	Description
RFC Identifier	check box	Identifier of the Regional Finance Center. This is relevant if you are deploying Oracle Payments for use by a United States federal agency.
Payment Function	check box	Function of the payment, that is, a type of payment. Examples of payment functions are Supplier Payment and Employee Reimbursement.

Specifying Payment Limits

You can specify a maximum payment amount and/or a maximum number of payments for a payment instruction. If you specify a maximum payment amount, you must specify an exchange rate type from the Exchange Rate Type drop-down box. Examples of the exchange rate type options are indicated in the table below.

Descriptions of Exchange Rate Type Options

Field	Description
Corporate	An exchange rate that is optionally used to perform foreign currency conversion. The corporate exchange rate is usually a standard market rate determined by senior financial management for use throughout the organization. This rate is defined in Oracle General Ledger.
Spot	A daily exchange rate used to perform foreign currency conversions. The spot exchange rate is usually a quoted market rate that applies to the immediate delivery of one currency for another.
User	A user-defined exchange rate.

Specifying Payment Sorting

Oracle Payments applies user-specified payment grouping rules to a pool of payments, thereby grouping individual payments into various payment instructions. The system then applies user-specified sorts, in sequence, so that payments within a payment instruction are sorted as specified.

Specifying Separate Remittance Advice

Remittance advice is a report sent to a payee that lists the documents payable paid as part of each payment. You can specify the format for the remittance advice document and the delivery method.

When the separate remittance advice Format field is specified, the Condition field under the Separate Remittance Advice region in the Reporting subtab is set to All Payments. You can change this value in the Update Payment Process Profile page if you wish.

If you select the Override Payee Delivery Method Preference check box, you can override the supplier/payee's delivery method preference that was set in the supplier/payee setup.

If the Override Payee Delivery Method Preference check box is deselected, the delivery method preference set at the supplier/payee level is used, overriding the delivery method set in the Create Payment Process Profile page or in the Update Payment Process Profile page.

Adding or Updating Usage Rules

The Usage Rules subtab displays the same regions and fields that appear in the Create Payment Process Profile page.

Adding or Updating the Payment System

The Payment System subtab displays the accounts associated with the payment system. If you wish, you can enable one or more payment system accounts.

Select a payment transmission protocol from the Payment Transmission Protocol drop-down list. The drop-down list displays seeded protocols that are linked to the payment system you selected from the Payment System list of values on the Create Payment Process Profile page.

Specifying the Payment System

The Payment System field is enterable in the Update Payment Process Profile page only if you do not select a payment system from the Payment System list of values on the Create Payment Process Profile page. If you select a payment system from the Payment System list of values in the Create Payment Process Profile page, you cannot update the Payment System field in the Update Payment Process Profile page because it is read-only.

If you select or enter a payment system in the Payment System field of the Update Payment Process Profile page, you must select at least one payment system account by selecting the corresponding Enabled check box.

Enabling Payment System Accounts

Payment system account names are user-defined identifiers that identify the deploying company's accounts with its payment system. All payment system account names are enabled by default and populate the Name field under the Payment System Accounts region when you navigate to the Update Payment Process Profile page for the first time from the Create Payment Process Profile page. The populated payment system account names are the same as the user-defined identifiers originally entered in the Name field of the Update Payment System Accounts page during shared setup for payment systems.

The value in the Account Payment Process Profile Name field defaults the concatenated string of the account payment process profile name and the payment system account name. If you wish, you can change this defaulted account payment process profile name.

The required value in the Account Payment Process Profile Name field is only valid for the account payment process profile names that are enabled. Users typically enable one payment system account name at a time by selecting the corresponding Enabled check box. You can, however, enable more than one payment system account if you want to

use the same payment process profile for different payment system accounts.

If you did not navigate directly from the Create Payment Process Profile page, but instead clicked the Update icon on the Payment Process Profiles page, and you deselect the Enabled check box in the Payment System subtab of the Update Payment Process Profile page, the system end dates the payment system account but does not delete it.

Adding or Updating Payment Creation

The information entered in the Payment Creation subtab is used to define payment creation rules for grouping documents payable into payments.

Specifying Document Grouping

Specifying document grouping options defines grouping rules used to group documents payable into payments.

Specifying Document Limits

In this region you can specify a maximum payment detail size limit and a maximum payment amount for payments. The payment detail is built using the Payment Detail Form field. This field should contain a SQL expression written by a database administrator. This SQL expression, which can reference columns of the documents payable table, is used by Oracle Payments to generate payment detail in the form of text that becomes part of the payment. An example of payment detail that would create the payment detail by concatenating the purchase order number and payment date of the documents in the payment might look as follows: PO_NUMBER || PAYMENT_DATE.

In the Maximum Payment Detail Length field you enter the maximum number of characters allowed for payment detail. This limit is imposed by the deploying company's payment system or a regulatory body.

Adding or Updating Payment Instruction Creation

The information entered in the Payment Instruction Creation subtab is used to group payments into payment instructions. The Payment Instruction Creation subtab includes the Bank Instructions region, which does not appear on the Create Payment Process Profile page.

Specifying Payment Grouping

The Internal Bank Account payment grouping option check box is only displayed when the selection for the Processing Type field is Electronic.

Specifying Bank Instructions

In this region, you specify bank instruction codes and other fields of text that are added to all payment instructions that are created using this payment process request. The

specified codes and text are usually used to provide additional payment processing instructions for the intended payment system or additional payment information for the intended payee. The table below describes the fields in the Bank Instructions region.

Description of Fields Under the Bank Instructions Region, Payment Instruction Creation Subtab of the Update Payment Process Profile Page

Field	Feature	Description
Bank Instruction 1 and 2	List of Values	Bank Instruction Code. For information on setting up bank instruction codes, see Step 14. Setting Up Bank Instruction Codes, page 5-7.
Bank Instruction Details	Enterable Field	Text that appears in the electronic payment instruction.
Payment Text Message 1 and 2	Enterable Fields	Messages that are added to electronic payment instructions and may be passed to payees by the payment system.

Adding or Updating Payment Instruction Format

The Payment Instruction Format subtab displays information about the format of the payment instruction that is submitted to the payment system or bank, along with the payment.

Specifying Payment File Information

The enterable fields in the Payment File Information region contain user-defined information that is requested by the payment system or bank to submit payment files. The table below describes the payment file information fields.

Description of Fields Under the Payment File Information Region, Payment Instruction Format Subtab of the Update Payment Process Profile Page

Field	Feature	Description
Outbound Payment File Prefix	Enterable field	Prefix on the file name that is submitted to the payment system or bank.
Outbound Payment File Extension	Enterable field	Extension on the file name that is submitted to the payment system or bank
Outbound Payment File Directory	Enterable field	Location on the deploying company's computer from which the payment file is submitted to the payment system or bank

Entering Sequence Information

Sequences are used in Oracle Payment to number sequential items. An example of a sequence is the sequential numbering of invoices. Database Administrators define sequences in the database. Payment systems or central banks may require that some of these sequences be reset on a periodic basis.

The table below describes the fields under the Periodic Sequences in Format region of the Payment Instruction Format subtab.

Description of Fields under the Periodic Sequences in Format Region, Payment Instruction Format Subtab of the Update Payment Process Profile Page

Field	Feature	Description
Sequence Name	Enterable field	Name of a sequence that a Database Administrator defines in the database. This value is linked to an entry in an Oracle XML Publisher template, which you select when you create a format during shared setup.

Field	Feature	Description
Payment System Account Name	Automatically populated by any enabled payment system accounts	Any enabled payment system account is displayed in the Payment System Account Name field.
Reset Sequence Value	Enterable field	Restarts the sequence at the value specified in the Reset Sequence Value field.
Last Used Number	Enterable field	A number, after which the sequence restarts.
Set Schedule	Icon	Opens the Concurrent Request program page, where you need to run the program with name Reset Periodic Sequence Value by specifying when and how frequently you want Oracle Payments to reset the sequence value.

Note: If no payment system is selected or entered for the Payment System field in the Payment System subtab of the Update Payment Process Profile page, then the Periodic Sequences in Format region is not displayed.

The Periodic Sequences in Format region always displays three rows, which enables you to enter up to three sequence names. When you enter a sequence name in the Periodic Sequences in Format region, the Sequence Settings table displays all the payment system account names that were enabled in the Payment System subtab of the Update Payment Process Profile page.

Adding or Updating Payment Reporting

The Reporting subtab displays enterable fields to select options for running various reports.

Specifying Payment Instruction Register Information

A payment instruction register is a report that is created for each payment instruction. The register indicates what payments are contained in that payment instruction.

Specifying Positive Pay Information

A positive pay file is a security measure in the form of a document that the deploying company sends to its payment system or bank to inform it of payments made by check. When you print checks, then you can electronically transmit a list of payments to the bank or payment system that indicates the checks you printed, so the bank or payment system knows what checks to pay. This list prevents the payment system or bank from paying fraudulent checks, since such checks are not listed on the positive pay file.

The table below describes selected fields under the Positive Pay region of the Reporting subtab.

Description of Fields Under the Positive Pay Region, Reporting Subtab of the Update Payment Process Profile Page

Field	Feature	Description
Outbound Payment File Prefix	Enterable field	Prefix entered on the file name of a positive pay file.
Outbound Payment File Extension	Enterable field	Extension entered on the file name of a positive pay file.
Outbound Payment File Directory	Enterable field	Folder on the deploying company's computer from which the positive pay file is submitted to the payment system or bank.
Automatically Transmit File	Check box	If selected, Oracle Payments transmits the positive pay file when payments are issued. That is, when checks are printed, then the positive pay file is generated and transmitted.

Specifying Separate Remittance Advice

Separate remittance advice is a document that lists the invoices paid with a particular payment. You can specify the format for the separate remittance advice document and the delivery method.

The table below describes selected fields under the Separate Remittance Advice region of the Reporting subtab.

Description of Fields Under the Separate Remittance Advice Region, Reporting Subtab of the Update Payment Process Profile Page

Field	Feature	Description
Condition	Drop-down list	<p>Specifies when or for which payments this remittance advice is generated.</p> <p>Number of Documents option indicates the number of payments that must be included in a payment instruction for Oracle Payments to generate separate remittance advice for the included payments. The Payment Detail Length option indicates the minimum payment detail length required to generate separate remittance advice for a payment.</p>

Specifying Regulatory Reporting

Regulatory reporting refers to reports required by a regulatory body, such as a level of government, the central bank, or an individual bank.

The fields in the Regulatory Reporting region of the Update Payment Process Profile page enable you to determine the conditions under which regulatory reporting can be generated. In addition to these conditions, you can use a SQL function if you want to implement more complex criteria. This SQL function overrides the fields in the user interface, such as the reporting threshold amount.

To implement this SQL function, you will need to update the following stub SQL function:

IBY_EXTENSIBILITY_CALLOUTS_PUB (ibyextcs/b.pls) with the following signature:

```
FUNCTION isCentralBankReportingRequired(  
    p_payment_id          IN NUMBER,  
    x_return_status       OUT NOCOPY VARCHAR2  
) RETURN VARCHAR2;
```

The function accepts one parameter, the Payment ID. This function must return either Y or N.

Adding or Updating Additional Information

This subtab enables you to define descriptive flexfields, if applicable.

Step 18. Setting Up Disbursement System Options

Disbursement system options are system-wide payment options that control disbursements made by the first party payer to suppliers. Oracle Payments provides two levels of system options; enterprise-level system options and organization-level system options, by operating unit or legal entity.

Overview

The first party payer, or deploying company, that disburses payments, can set system options for payment features. Oracle Payments seeds one enterprise-level system option on the Disbursement System Options page. When you access this page, you can view the default system options for the entire enterprise. Enterprise-level system options are updateable if you have been assigned security update permission.

You can also view the system options for each organization to which you have access in your security profile. This means that you can only view and update those organizations to which you have security access. For information on security access for multiple organizations, see *Define Multiple Organization Security Profile*, *Oracle Applications Multiple Organizations Implementation Guide*.

Upon initial implementation, the enterprise-level settings display the seeded settings for enterprise-wide options, which are used for the enterprise and all organizations within the enterprise. Once you change a value on the Update Disbursement System Options: Enterprise-wide page, the user interface displays the existing values in the database. If you have update access, you can change the system options at the enterprise-level or the organization-level. Making a change at an organization level creates a record for the organization. After that, any changes made to the enterprise-level do not update the organization-level.

Note: Only some system options, such as default payment method, can be set at the organization-level.

Note: The Disbursement System Options are treated as defaults. Source products can override these settings when a payment process request is submitted.

Purpose

The purpose of setting up disbursement system options is to specify how the payment

process runs at the enterprise and organization-levels.

Updating Default Payment Method System Options

The table below describes the default payment method system options.

Description of Default Payment Method System Options

Default Payment Method Region	Features	Description
Based Only on Payment Method Defaulting Rules Setup.	Radio button	This option uses the payment method defaulting rules set up in Step 12 in Oracle Payments.
Override Defaulting Rules when Default Method Set for Payee.	Radio button	This option uses the default payment method set for each supplier, or payee, in iSupplier Portal.

Updating Payment Processing System Options

The table below describes selected payment processing system options.

Selected Payment Processing System Options

Region/Field Name	Features	Description
Validation Failure Results Region		
Document	Drop-down list	These options either direct Oracle Payments to stop payment processing for review of the applicable documents if validation failures occur or to reject some or all of the documents.

Region/Field Name	Features	Description
Payment	Drop-down list	These options either direct Oracle Payments to stop payment processing for review of the applicable payments if validation failures occur or to reject some or all of the payments.
Proposed Payments Region		
Review Proposed Payments after Creation	Drop-down list	This field determines whether the payment process is stopped after payments are created and validated, to give the Payment Administrator the ability to review and potentially remove payments.
Allow Payee Bank Account Override on Proposed Payments	Check box	<p>If the check box is selected, you can change the bank account to which you are making a payment on the Review Proposed Payments page. If Review Proposed Payments after Creation is set to No, this field is not used.</p> <p>If the check box is deselected, you cannot change the bank account to which you are making a payment.</p>
Payment Process Request Status Report Region		
Format	List of values	Oracle Payments seeds one Status report format, which you can select from the list of values. You can also create your own Status report formats. If you create your own Status report formats, they are available as options from the list of values.

Region/Field Name	Features	Description
Automatically Submit at Payment Process Request Completion	Check box	<p>If the check box is selected, Oracle Payments automatically runs the Status report after the Build Payments program completes.</p> <p>If the check box is deselected, Oracle Payments does not run the Status report after the Build Payments Program completes.</p>
Payment Instructions Region		
Save Formatted Payment File in Database	Check box	If the check box is selected, the formatted payment file created from the payment instruction is stored in the database.

Updating Default Payment Specifications for Payee System Options

The table below describes the default payment specifications for payee system options.

Default Payment Specifications for Payee System Options

Field Name	Features	Description
Bank Charge Bearer	Drop-down list	When funds are sent by EFT, the bank that does the processing charges a fee. By selecting an option from the Bank Charge Bearer drop-down list, you can indicate the party who is responsible for paying the EFT fee. This field may not be used by all banks in all countries.

Field Name	Features	Description
Pay Each Document Alone	Check box	If the check box is selected, while creating a new supplier the Pay each document alone checkbox at the supplier level is automatically selected. That is, the 'Pay each document alone option defaults based on this setup.

Setup Tasks for Funds Capture

Overview

Note: If you intend to use the funds capture functionality, you must complete the funds capture setup steps.

Funds capture is the automated funds capture process through electronic payment methods, such as direct debits of bank accounts, credit cards, and remittance of bills receivable, where payment is retrieved, or captured, from the payer who owes a debt to the payee.

The Oracle Payments setup checklist for the funds capture process is shown in the table below. These setup steps must be performed in the sequence indicated.

Funds Capture Setup Checklist

Step Number	Setup Step	Step Type	Oracle Application
19.	Configuring the ECAApp Servlet	required	Oracle Payments
20.	Setting Up Funds Capture Payment Methods	required	Oracle Payments
21.	Setting Up Funds Capture Process Profiles	required	Oracle Payments

Step Number	Setup Step	Step Type	Oracle Application
22.	Setting Up First Party Payees	required	Oracle Payments
23.	Setting Up Credit Card Brands	conditionally required	Oracle Payments
24.	Loading Risky Instruments	optional	Oracle Payments

Step 19. Configuring the ECApp Servlet

An ECApp servlet is the only front-end servlet in Oracle Payments. You must configure the ECApp servlet to use Oracle Payments PL/SQL API and 3i Backward Compatibility API.

Oracle Payments integrations packaged with E-Business Suite products start functioning after you configure the ECApp servlet. The ECApp servlet provides an interface to the Oracle Payments engine to process payment-related operations such as authorization, capture, and return.

Setting Up SSL Security for the ECApp Servlet

To enable SSL security for the ECApp servlet, perform the following steps:

1. Create a wallet file on the database tier.
2. Import the certificate for the Certifying Authority (CA), which signed the Middle Tier's server certificate into the wallet file.
3. Save the wallet with auto login mode enabled.
4. Specify the location of the wallet for all E-Business Suite applications by running AutoConfig or manually setting the system profile option Database Wallet Directory (FND_DB_WALLET_DIR) to the full path of the directory where the wallet file resides.

Related Topics

For information on the Oracle wallet, see *Oracle Advanced Security Administration Guide*.

Step 20. Setting Up Funds Capture Payment Methods

A funds capture payment method is a payment medium by which a third party payer, such as a customer, chooses to remit payment to the first party payee, which is the deploying company. Oracle Payments supports three funds capture payment methods for automated funds capture processing: credit cards, PINless debit cards, and bank account transfers.

Oracle Payments seeds three updateable funds capture payment methods for credit card, PINless debit card, and bank account transfer instrument types and also seeds five updateable payer-initiated instrument types, such as check or wire transfer. For payer-initiated instrument types, Oracle Payments records the transaction, but does not perform any processing for these payment methods.

From the Funds Capture Payment Methods page, you can view and update the seeded funds capture payment methods or define new funds capture payment methods for bank account transfers only.

How Source Product Users Employ Funds Capture Payment Methods

Once the funds capture payment methods are set up, they are available for selection by the source product user while creating or updating external payers. Additionally, source product users select one of the following seeded funds capture payment methods before submitting a payment transaction to Oracle Payments:

- ACH (Automated Clearing House) Transfer
- EFT (Electronic Funds Transfer) Bank Transfer
- Credit Card
- PINless Debit Card
- Check
- Cash Payment
- Wire Transfer
- Invoice
- Offline Manual Payment

Of those in the preceding list, the first two are bank account transfer methods and the last five are payer-initiated methods.

Purpose

The purpose of setting up funds capture payment methods is to assign validation sets to seeded funds capture payment methods and to assign validation sets and define details of bank account transfers.

Creating Payment Methods

When you create a funds capture payment method for the processing type of bank account transfer, you enter user-defined values in the Code field, which is an identifier, or shortname, for the funds capture payment method you are creating.

Assigning Validations

In the Assign Validation Sets region, you assign applicable validation sets to the funds capture payment methods of processing type Bank Account Transfer that you are creating.

Note: The Assign Validation Sets and Descriptive Flexfields regions only display for non payer-initiated payment instrument types.

Step 21. Setting Up Funds Capture Process Profiles

A funds capture process profile is a setup entity that acts as a blueprint for funds capture processing. It is equivalent to the payment process profile for funds disbursement processing. When you create a funds capture process profile, you specify funds capture processing rules for a specific payment system. These funds capture process profile rules include how:

- online messages, such as authorizations, for bank account transfers, credit cards, and debit cards are formatted and transmitted
- settlements are aggregated into a settlement batch
- settlement batches are formatted and transmitted
- acknowledgements received from the payment system are received and parsed
- payment system account information is linked into the payment process

Prerequisites

Before you can set up funds capture process profiles, you must perform the following set up steps:

- formats
- transmission configurations
- payment systems
- funds capture payment methods

Purpose

The purpose of setting up funds capture process profiles is to specify funds capture processing rules for a specific payment system. The funds capture process profiles contain all the rules necessary for the execution of funds capture orders for specific payment systems.

You must update the Payer notification values in the Funds Capture Process Profiles for the update to happen successfully.

Sending and Receiving Online Payer Verifications, Authorizations, and Debits

The region name that you see in the Create Funds Capture Process Profiles page is dependent upon the option selected in the Processing Type drop-down list.

Sending and Receiving Online Payer Verifications

If you select Bank Account Transfer from the Processing Type drop-down list in the Create Funds Capture Process Profile page, the Online Payer Verification region displays. By selecting outbound and inbound payment system formats, as well as a transmission protocol, the deploying company ensures that it can:

- send an online message to the payment system, asking it to verify that the transmission configuration is set up correctly to capture funds from customer bank accounts
- receive an online response message from the payment system

Note: Online Payer Verification is an optional feature and is not offered by all payment systems.

Sending and Receiving Online Authorization

If you select Credit Card from the Processing Type drop-down list in the Create Funds Capture Process Profile page, the Online Authorization region displays. By selecting outbound and inbound payment system formats, as well as the transmission protocol, the deploying company ensures that it can:

- send an online message to the payment system, asking it to authorize an online

credit card payment from the customer's credit card account with the issuing bank

- receive an online response message from the payment system

Sending and Receiving Online Debits

If you select Debit Card from the Processing Type drop-down list in the Create Funds Capture Process Profile page, the Online Debit region displays. By selecting outbound and inbound payment system formats, as well as the transmission protocol, the deploying company ensures that it can:

- send an online message to the payment system, asking it to debit a customer's checking account for payment via the customer's debit card
- receive an online response message from the payment system

Sending and Receiving Settlements

By selecting outbound and inbound payment system formats, as well as the transmission protocol, the deploying company ensures that it can:

- send a settlement request to the payment system, which includes individual online settlements for gateway-model payment systems and settlement batches for processor-model payment systems
- receive an online response message from gateway-model payment systems

Sending Payer Notifications to Payers

Payer notification is a message sent by the deploying company to each of its customers after the settlement or settlement batch is transmitted, notifying them of a funds capture transaction. This notification is formatted according to the option selected in the Format field of the Payer Notification region and delivered to the payer, that is, the deploying company's customer, by e-mail, fax, or United States mail.

Note: If the payment system is a processor, then the Automatically Submit at Settlement Completion check box appears in the Payer Notification region of the Create Funds Capture Process Profile page. If the payment system is a gateway, then the Automatically Submit at Settlement Completion check box disappears from the Payer Notification region of the Create Funds Capture Process Profile page.

Receiving Acknowledgements from Payment Systems

By selecting outbound and inbound payment system formats, as well as the

transmission protocol, the deploying company ensures that it can:

- Actively retrieve a response from the payment system, acknowledging that a settlement or batch settlement was received or not received from the deploying company.
- Parse the acknowledgement message.

Acknowledgements are sent from the payment system to the deploying company, indicating whether credit card or bank account transfer settlements were made. Acknowledgements can be pushed by the payment system onto the deploying company's system, or the deploying company may need to actively retrieve acknowledgements from the payment system. Acknowledgements can also indicate whether the settlement batch was processed and had successes or failures.

A payment system can reject a settlement batch when the batch is incomplete or missing information.

Specifying Settlement Grouping

Settlement grouping begins when you select check boxes in the Settlement Grouping region of the Create Funds Capture Process Profile page. Selection of settlement grouping attributes specifies that settlements with the same settlement grouping attributes will be included in a unique settlement batch when that Funds Capture Process Profile is used.

The grouping example in the table below illustrates how settlements A to D are grouped into settlement batches, given the three settlement grouping attributes selected for Funds Capture Process Profile Y. The following settlement grouping attributes are used in the example:

- First Party Organization
- First Party Legal Entity
- Settlement Date

Example of Settlement Grouping into Settlement Batches for the Funds Capture Process Profile Y

Settlement	Settlement Attributes on Settlement	Unique Settlement Batches Contains...
A	<ul style="list-style-type: none"> First Party Organization = Organization Unit 1 First Party Legal Entity = Oracle North America Settlement Date = January 1, 2006 	Settlement Batch I contains: <ul style="list-style-type: none"> Settlement A Settlement B
B	<ul style="list-style-type: none"> First Party Organization = Organization Unit 1 First Party Legal Entity = Oracle North America Settlement Date = January 1, 2006 	
C	<ul style="list-style-type: none"> First Party Organization = Organization Unit 2 First Party Legal Entity = Oracle North America Settlement Date = January 1, 2006 	Settlement Batch II contains: <ul style="list-style-type: none"> Settlement C
D	<ul style="list-style-type: none"> First Party Organization = Organization Unit 1 First Party Legal Entity = Oracle North America Settlement Date = January 2, 2006 	Settlement Batch III contains: <ul style="list-style-type: none"> Settlement D

Specifying Settlement Limits

You can specify limits that apply to each settlement batch, including the maximum number of settlements in a batch.

The Exchange Rate Type drop-down list has the following options:

- **Corporate:** An exchange rate you define to standardize rates for your company. This rate is generally a standard market rate determined by senior financial management for use throughout the organization.
- **Spot:** An exchange rate which you enter to perform conversion based on the rate on a specific date. It applies to the immediate delivery of a currency.
- **User:** An exchange rate you specify when you enter a foreign currency journal entry.

Specifying Payment System Accounts

When you specify payment system accounts to be used with the funds capture process profile, you are effectively creating separate profiles at the account-level. By default, these account-level profiles are named by concatenating the funds capture process profile name and the payment system account name, but you can change it.

Once the identifier is entered, you can select the applicable transmission configuration that corresponds to each selected transmission protocol.

The transmission configuration options available in the Payment System Accounts region are dependent on the options selected in the Transmission Protocol fields.

Note: When you change options in the Transmission Protocols fields, you must clear the transmission configuration selections and reselect.

The following conditions apply to the selection of transmission configurations:

- When you select a transmission protocol, the corresponding transmission configuration list of values only displays the transmission configurations for the selected transmission protocol.
- If you do not initially select a transmission protocol and then select a transmission configuration, the corresponding transmission protocol field is populated.

Step 22. Setting Up First Party Payees

On the funds capture side, the first party payee is the deploying company, or organizations within the deploying company, that receives funds from its customers, the payers, by credit card payments, debit card payments, direct debits to bank

accounts, and bills receivable transactions sent to banks.

Prerequisites

Before you can set up first party payees, you must perform the following setup steps:

- payment systems
- payment system accounts
- payment methods
- funds capture process profiles

Purpose

The purpose of setting up first party payees is to tie the payment processing rules of the funds capture process profile to the business entities that need to use them.

Completing the Header

In the Code field of the Create Payee page, enter a shortname for the payee. This value is not updateable in the Update Payee page.

In the Merchant Category Code field of the Create Payee page, enter a four-digit code that is usually supplied by the payment system, which identifies the industry in which the deploying company operates.

Selecting Payment Systems and Payment System Accounts

In the Payment System Accounts region of the Create Payee page, you select payment systems from the Payment System list of values that are used by the deploying company or its organizations. Additionally, you select payment system accounts from the Payment System Account list of values.

Each payment system is associated with one or more payment system accounts. The payment system account is an account identifier used by the deploying company's payment system, which is stored in Oracle Payments.

Note: If you do not select a payment system from the Payment System list of values, then all the payment system accounts are displayed in the Payment System Account list of values and the Payment System field is populated with the option that corresponds to the selected payment system account.

Assigning Operating Units

In the Assign Operating Units region of the Create Payee page, you can assign one or more operating units to the payee. The assignment of one or more operating units to a first party payee occurs because source products attribute each funds capture transaction to an operating unit. This setup allows Oracle Payments to map a transaction from an operating unit to a payee. Each operating unit may only be assigned to one payee.

Selecting Default Payment Systems

Default payment systems are used by Oracle Payments to process transactions only if the routing rules do not specify a payment system account or none of the conditions in the routing rules are met for the transaction in question.

Selecting Purchase Card Processing Transaction Detail

You can specify the level of transaction detail that is sent to the payment system for purchase card processing, along with the settlement. Level II of purchase card processing transaction detail includes the amount to be captured plus additional information, such as tax and shipping charges. Level III of purchase card processing transaction detail includes the amount to be captured plus tax and shipping charges, as well as itemization of the products or services purchased, such as one sweater, two pairs of slacks, and three pairs of shoes.

Creating a Risk Formula

A risk formula is a group of risk factors plus weights, which tells Oracle Payments how to calculate the risk score of credit card transactions. A risk factor is any information, which a first party payee uses to evaluate the risk of a customer who wishes to buy products or services from the payee. Examples of risk factors are address verification, payment history, and transaction amount.

One or more risk formulas, comprised of risk factors, can be configured for each first party payee. First party payees can associate each risk factor with different weights that must total 100. Based on their business model, deploying companies can use any number of the seeded risk factors for their organizations.

The majority of the risk factors have associated updatable fields for which you enter information or select options. For some risk factors, such as address verification code or risk scores, your payment system provides values for you to enter.

Specifying a Risk Threshold

A risk threshold is a number value, or score, between 0 and 100, above which, the deploying company may want to review or reject a credit card or a purchase card as payment for products or services. A zero risk threshold represents no risk and 100

represents the highest possible risk. The risk threshold value is determined and entered after reviewing all the risk factors for the particular payee and their associated weights.

Creating Routing Rules

Oracle Payments routes settlement requests to the appropriate payment systems. Each first party payee may have one or more routing rules with corresponding priorities. Routing rules determine which payment system account and which funds capture process profile are used to process funds capture transactions for specific payment methods.

Routing rules are comprised of destination information and one or more conditions that must be met if the funds capture transaction is to be routed using that rule's destination information. Destination information specifies the payment system account to which the transaction is routed. A condition is comprised of a criterion, (such as Amount, Currency, Organization, Card Type, or Bank Routing Number) an operator related to the criterion, and the value of the criterion. An example of a condition is **Amount** (criterion) **Greater Than** (operator) **\$1,000** (value).

Routing rule are prioritized and the one with the highest priority is evaluated by Oracle Payments first. If the values in the requested funds capture transaction match the conditions in the routing rules, the settlement request is routed to the applicable payment system account for processing.

Step 23. Setting Up Credit Card Brands

Oracle Payments seeds the major credit card brands in the Credit Card Issuers Setup page. If necessary, you can add more brands and update them after implementation.

Purpose

The purpose of setting up credit card brands is to enable the credit card brands that your company accepts for payment, along with their associated authorization validity periods, in days, that your company uses.

Step 24. Loading Risky Instruments

The Risky Instruments upload utility, RiskyInstrUtil, is a Java application used to store risky payment instruments.

Requirements

- Java executable in your application environment
- Oracle Applications Java class Library in the CLASSPATH. The Oracle Applications Java class Library is included in the classpath after you set up the applications

environment.

Java Commands

A command using the syntax below requires an operation and a filename. It modifies the risky instruments table in the database, depending on the entries in the file.

```
java<add one space here>-DJTFDBCFILE=<dbc filelocation>-  
DAFLOG_ENABLED=TRUE -DAFLOG_MODULE=iby%-DAFLOG_LEVEL=STATEMENT -  
DAFLOG_FILENAME=./af.logoracle.apps.iby.irisk.admin.RiskyInstrUtil  
[ADD/DELETE] [filename]  
  
<remove the Or>
```

The command below deletes all the risky instruments in the table.

```
java<add one space here>-DJTFDBCFILE=<dbc filelocation>-  
DAFLOG_ENABLED=TRUE -DAFLOG_MODULE=iby%-DAFLOG_LEVEL=STATEMENT -  
DAFLOG_FILENAME=./af.logoracle.apps.iby.irisk.admin.RiskyInstrUtil  
DELETE all
```

File Format

The file format for risky instruments is as follows:

- Each line corresponds to one risky instrument.
- The fields are comma separated and are in the following order: Payee identifier, instrument type, and credit card number. Instrument type has to be a CREDITCARD. For example: payee1, CREDITCARD, XXXXXXXXXXXXXXXX.
- For the add operation, each risky instrument in the file that has a valid payee identifier, instrument type, and a new credit card number is added to the table.
- For the delete operation, each risky instrument that matches the payee identifier, instrument type, and the credit card fields is deleted from the table.
- The command prints the results of the operation on each risky instrument in the file.

Transaction Testing

Testing Transactions

In a live instance of Oracle Payments, all transaction requests are submitted from a source product. While implementing Oracle Payments, however, it is possible that the Payment Administrator may wish to initiate transactions without source products to test Oracle Payments setup and the payment system connectivity. To allow the Payment Administrator to perform such basic testing, the Funds Capture Process Home page provides a Transaction Testing tab that enables him to initiate authorizations and settlements using credit cards or debit cards.

By default, the Payment Administrator does not have the Transaction Testing function available. It should only be enabled, through menu functions, for basic connectivity testing, and then disabled again before going live. Transactions initiated from Oracle Payments, rather than the source product, are not recorded in any source product, nor will they be accounted for. This creates potential for inconsistent data between applications at best, and opportunity for fraud or theft at the worst.

Warning: On a live instance of Oracle Payments, it is strongly recommended that the Transaction Testing function be disabled, thereby unassigning it from the Payment Administrator.

Creating and Testing Authorizations

The Transaction Testing tab is comprised of a Transaction Testing Search page, a three-page train for testing the initiation and submission of authorizations, and a Create Settlement page, which enables the Payment Administrator to provide Oracle Payments with the settlement amount for testing purposes.

Using Oracle Payments with External Payment Systems

Overview of Payment System Integration

This chapter explains how to integrate Oracle Payments with an external payment system.

Oracle Payments architecture allows for integration with third party payment systems for both inbound (funds capture) and outbound (disbursements) transaction processing.

In the funds capture flow, the external payment system can be one of two kinds: gateway or processor. For gateway-model payment systems, every transaction is online and involves real-time communication with the payment system. Processor model payment systems support real-time communication for authorizations and batch operation for settlement processing.

In the disbursements flow, there is no concept of real-time transaction processing. Payment processing is performed in batch mode only (usually by FTPing a payment file to the payment system).

Payment System Integration Components

When implementing an integration between Oracle Payments and a custom payment system, you will need to develop a number of integration components. Integration components are building blocks that are invoked by Oracle Payments at different stages of the payment processing cycle.

The list of components to be implemented depends on various factors such as the type of payment flow (whether funds capture or disbursements), the type of the payment system (whether gateway or processor), and the type of the payment instrument, whether credit card, debit card, or bank account.

The table below lists the various integration component types that may be implemented for interfacing with a custom payment system. Your custom implementation may only

require a subset of the listed integration components.

Integration Point Type Component Types

Integration Component	Description
Format	<p>Used in the construction of a payment system-specific message or payment file.</p> <p>Processor model payment systems typically support two formats: one for online transactions and the other for batch transactions.</p> <p>Note: Gateway model payment systems typically support online formats only.</p>
Format Validation	<p>Contains the set of payment system-specific validations.</p>
Transmission Function	<p>Define how to communicate with the payment system.</p>
Acknowledgement Parser	<p>Define how Oracle Payments handles a response message from the payment system to a request message, whether online or batch.</p>
Payment System	<p>Define attributes for the payment system, which you provide in the setup user interface.</p>
Process Profile	<p>Define the attributes of a payment system, which is used to control the payment processing behavior within Oracle Payments.</p> <p>Note: The process profile can be a funds capture process profile for funds capture or a payment process profile for disbursements.</p>

Note: Several of the integration component types are developed in Java.

Oracle Payments supports both inbound (funds capture) and outbound (disbursement) flows. Therefore, separate sections outline how you can implement a payment system

integration for funds capture and for funds disbursement. The sections below explain in detail the tasks you must perform to achieve a custom payment system integration.

For both funds capture and funds disbursement, you must perform some basic setups using the Oracle Payments Setup page. These basic setups are similar for funds capture and funds disbursement and involve the creation of entities, such as payment systems, payment system accounts, and process profiles.

The rest of the integration involves creating integration components, such as format, transmission, validation and acknowledgement. Creation of these components depends on the characteristics of your payment system and are discussed in detail below.

Developing a Custom Payment System Integration for Funds Capture

The list of tasks required to be performed for integration with a custom payment system depends on the following factors:

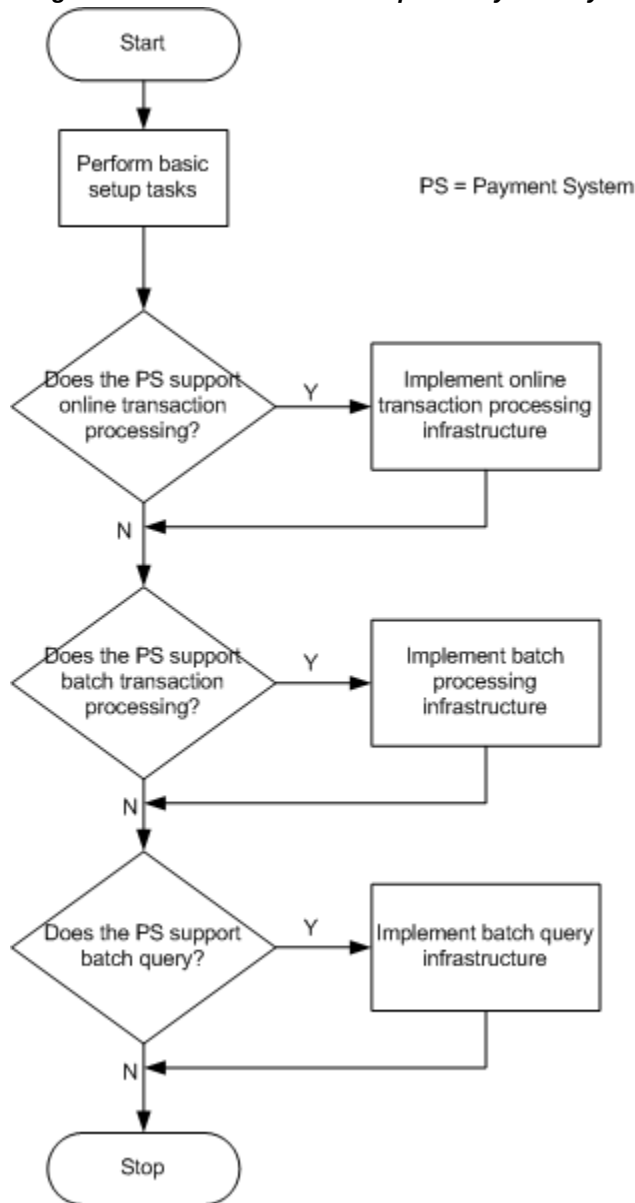
- the type of payment system, whether gateway or processor model
- the type of payment instrument, whether credit card, debit card, or bank account

Implementation guidelines for integrating with a custom payment system include the following:

- For credit card and debit card processing, an online authorization transaction is usually mandatory, regardless of whether the payment system is a gateway or a processor. For bank account transactions, there is no authorization step. Some payment systems support the concept of online verification, where some basic validations on the bank account are performed.
- In the gateway model, only individual transactions are sent to the payment system. There is no concept of a batch. In the processor model payment systems, authorizations are sent as online transactions (processed real-time), whereas settlements are processed in batches offline.
- Processor model payment systems may or may not support acknowledgments. If acknowledgements are supported, a batch query may be performed to retrieve the settlement status from the payment system and set the settlement transactions to their final status. If the processor model payment system does not support batch queries, then the submission of the settlement is the final step in Oracle Payments.

To determine what tasks you need to perform for your custom integration, see the flowchart below.

Integration Task List for Funds Capture Payment Systems



The table below lists the tasks that must be completed for integrating a payment system for funds capture.

Basic Setup Tasks for Integrating a Custom Payment System

Task	Mandatory	Description
Define the payment system.	Yes	<p>Set up the new payment system.</p> <p>For information on setting up payment systems, see Step 8. Setting Up Payment Systems, page 4-13.</p>
Define payment system account options.	No	<p>Set up the options for all accounts with the payment system.</p>
Create funds capture process profile.	Yes	<p>Define a funds capture process profile specific to the payment instrument (credit card, debit card, or bank account).</p> <p>The funds capture process profile controls the behavior of all funds capture operations.</p> <p>For information on setting up payment systems, see Step 21. Setting Up Funds Capture Process Profiles, page 6-4.</p>
Create transaction validation.	No	<p>Required only if you want to implement payment system-specific validations.</p>
Create batch validation.	No	<p>Never for gateway model payment system.</p> <p>For processor model payment system, required only if you want to implement payment system-specific validations.</p>
Seed validation assignments.	No	<p>Required only if you want to implement payment system-specific validations.</p>

Online Transaction Processing Infrastructure Tasks

Task	Mandatory	Description
Develop an online authorization/verification format template.	No	<p>Define a payment system-specific authorization format and develop a template for XML Publisher.</p> <p>For credit card and debit card processing, it is necessary to implement online authorization formats.</p> <p>For processing bank account transactions, implementation of online verification formats is payment system dependant.</p> <p>Create XML Publisher templates using the elements of the funds capture extract.</p> <p>For information on funds capture extract, see Funds Capture Extract, page J-2.</p>
Set up the online authorization/verification format template.	No	<p>Define system data attributes for the new format. Link the newly created XML Publisher template to the format.</p> <p>For information on setting up formats, see Step 4. Setting Up Formats, page 4-10.</p>
Develop an online authorization/verification transmission function.	No	<p>Implement a transmission function protocol.</p>

Task	Mandatory	Description
Setup the online authorization/verification transmission protocol.	No	<p>Define a transmission protocol and associate it with the transmission function code-point that implements it. Also define the protocol parameters.</p> <p>For information on setting up transmission configurations, see Step 6. Setting Up Transmission Configurations, page 4-11.</p>
Develop an online authorization/verification acknowledgement parser.	No	Implement an acknowledgement parser.
Setup the online authorization/verification acknowledgement parser.	No	Define system data attributes for the parser.
Develop a settlement format template.	No	<p>For gateway model payment system, the format may be identical to the authorization format template.</p> <p>This task is not relevant for processors.</p> <p>Create XML Publisher templates using the elements of the funds capture extract.</p> <p>For information on funds capture extract, see Funds Capture Extract, page J-2.</p>
Setup the settlement format template.	No	<p>This task is not relevant for processors.</p> <p>Link the newly created XML Publisher template to the format.</p> <p>For information on setting up formats, see Step 4. Setting Up Formats, page 4-10.</p>

Task	Mandatory	Description
Develop a settlement transmission function.	No	<p>For gateway model payment systems, the transmission function may be identical to the authorization transmission function.</p> <p>This task is not relevant for processors.</p>
Setup the settlement transmission protocol.	No	<p>This task is not relevant for processors.</p> <p>For information on setting up transmission configurations, see Step 6. Setting Up Transmission Configurations, page 4-11.</p>
Develop a settlement acknowledgement parser.	No	<p>For gateway model payment systems, the acknowledgement parser may be identical to the authorization parser.</p> <p>This task is not relevant for processors.</p>
Develop a settlement query format template.	No	<p>Query support is optional for gateway model payment systems.</p> <p>For most gateway model payment systems, an acknowledgement request message is not defined.</p> <p>This task is not relevant for processors.</p> <p>Create XML Publisher templates using the elements of the funds capture extract.</p> <p>For information on funds capture extract, see Funds Capture Extract, page J-2.</p>

Task	Mandatory	Description
Setup the settlement query format template.	No	Optional for gateway model payment systems. Link the newly created XML Publisher template to the format. This task is not relevant for processors.
Develop a settlement query transmission function.	No	Optional for gateway model payment systems. This task is not relevant for processors.
Setup the settlement query transmission protocol.	No	Optional for gateway model payment systems. This task is not relevant for processors.
Develop a settlement query acknowledgement parser.	No	Optional for gateway model payment systems. This task is not relevant for processors.

The following note is relevant to the Batch Processing Infrastructure Tasks table shown below.

Note: Batch processing tasks are not relevant for gateways.

Batch Processing Infrastructure Tasks

Task	Mandatory	Description
Develop a settlement format template.	Yes	<p>Define a payment system-specific settlement format message.</p> <p>Create XML Publisher templates using the elements of the funds capture extract.</p> <p>For information on funds capture extract, see Funds Capture Extract, page J-2.</p>
Setup the settlement format template.	Yes	<p>Link the newly created XML Publisher template to the format.</p> <p>For information on setting up formats, see Step 4. Setting Up Formats, page 4-10.</p>
Develop a settlement transmission function.	Yes	<p>Implement the transmission function to communicate with the payment system for settlements.</p>
Setup the settlement transmission protocol.	Yes	<p>Define a transmission protocol and associate it with the transmission function code-point that implements it. Also define the protocol parameters.</p> <p>For information on setting up transmission configurations, see Step 6. Setting Up Transmission Configurations, page 4-11.</p>
Develop a settlement acknowledgement parser.	Yes	<p>Implement the settlement acknowledgement parser.</p>
Setup the settlement acknowledgement parser.	Yes	<p>Define the system data attributes for the acknowledgement parser.</p>

The following note is relevant to the Batch Query Infrastructure Tasks table shown below.

Note: Batch query tasks are not relevant for gateways.

Batch Query Infrastructure Tasks

Task	Mandatory	Description
Develop a settlement query format template.	No	<p>Define a payment system-specific query format message.</p> <p>Create XML Publisher templates using the elements of the funds capture extract.</p> <p>For information on funds capture extract, see Funds Capture Extract, page J-2.</p>
Setup the settlement query format template.	No	<p>Link the newly created XML Publisher template to the format.</p> <p>For information on setting up formats, see Step 4. Setting Up Formats, page 4-10.</p>
Develop a settlement query transmission function.	No	<p>Implement the transmission function to communicate with the payment system for queries.</p>
Setup the settlement query transmission protocol.	No	<p>Define a transmission protocol and associate it with the transmission function code-point that implements it. Also define the protocol parameters.</p> <p>For information on setting up transmission configurations, see Step 6. Setting Up Transmission Configurations, page 4-11.</p>

Task	Mandatory	Description
Develop a settlement query acknowledgement parser.	No	Implement the query acknowledgement parser.
Setup the settlement query acknowledgement parser.	No	Define the system data attributes for the acknowledgement parser.

Developing a Custom Payment System Integration for Funds Disbursement

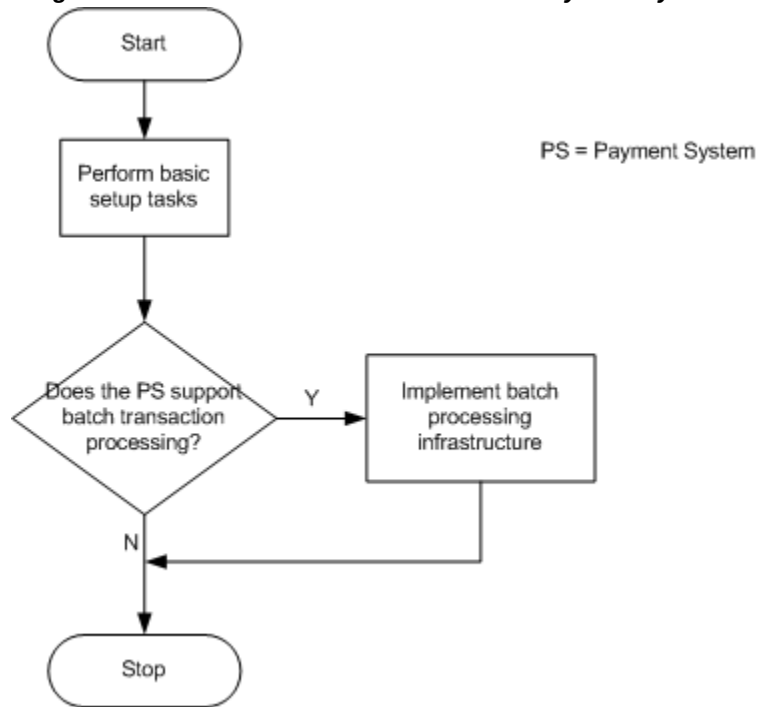
Unlike funds capture, funds disbursement integrations are always offline using payment files. There is no concept of online/real-time transaction processing in the funds disbursement flow.

In the funds disbursement flow, your implementation will typically involve the creation of a payment file using data from a payment instruction (also known as a payment batch). The payment file is created in a payment system-specific format, and this file is transferred to the payment system, usually via FTP.

In case the payment system supports query functionality, you will need to implement a query message and an acknowledgment parser.

Use the flowchart below to determine what tasks you need to perform for your custom integration.

Integration Task List for Funds Disbursement Payment Systems



Below is the Basic Setup Tasks table, which shows the tasks for integrating a custom payment system.

Basic Setup Tasks

Task	Mandatory	Description
Define the payment system.	Yes	Set up the new payment system.
Define payment system account options.	No	Set up the options for all accounts with the payment system.
Create payment process profile.	Yes	<p>The payment process profiles define how payments are created and disbursed.</p> <p>The profile controls the behavior of the funds disbursement operations.</p>

Task	Mandatory	Description
Create document validations.	No	<p>Implement business rules to validate documents payable.</p> <p>You can validate individual attributes of the document, such as internal bank account, document payment amount, or payee bank account.</p>
Create payment validations.	No	<p>Implement business rules to validate payments.</p> <p>You can validate individual attributes of the payment, such as payment amount or payment currency.</p>
Create payment instruction validations.	No	<p>Implement business rules to validate payment instructions.</p> <p>You may validate the number of transactions in the payment instruction.</p>
Seed validation assignments.	No	<p>Link the validation sets to payment methods and payment formats.</p>

The following note is relevant to the Batch Processing Infrastructure Tasks table shown below.

Note: Batch processing tasks are not relevant for gateways.

Batch Processing Infrastructure Tasks

Task	Mandatory	Description
Develop a settlement format template.	Yes	<p>Define a payment system-specific settlement format message.</p> <p>Create XML Publisher templates using the elements of the funds disbursement extract.</p> <p>For information on funds disbursement extract, see Extract Structure Overview, page K-1.</p>
Setup the settlement format template.	Yes	<p>Link the newly created XML Publisher template to the format.</p> <p>For information on setting up formats, see Step 4. Setting Up Formats, page 4-10.</p>
Develop a settlement transmission function.	No	<p>Implement the transmission function to communicate with the payment system for settlements.</p>
Configure the settlement transmission protocol.	No	<p>Define a transmission protocol and associate it with the transmission function code-point that implements it. Also define the protocol parameters.</p> <p>For information on setting up transmission configurations, see Step 6. Setting Up Transmission Configurations, page 4-11.</p>
Develop a settlement acknowledgement parser.	Yes	<p>Implement the settlement acknowledgement parser.</p>

Task	Mandatory	Description
Setup the settlement acknowledgement parser.	Yes	Define the system data attributes for the acknowledgement parser.

Defining a Payment System

The first task in integrating any payment system is defining a payment system. The definition must include both the payment system's attributes as well as the account options, if any, defined for the payment system accounts the users will establish.

Payment System Attributes

The table lists key attributes of the payment system.

Key Attributes of the Payment System

Attribute	Description	Constraints
Payment System	The name of the payment system.	Required.
Supported Instruments	The instrument types supported by the payment system.	Must be one of the following instrument types: <ul style="list-style-type: none"> • Credit Card • Purchase Card Level II/Level III • PINless debit card • Bank Account

Attribute	Description	Constraints
Type	Payment system model type. See Understanding Gateway-Model and Processor-Model Payment Systems, page 3-16 for a discussion of the differences between the processor and gateway payment system models.	Must be processor or gateway model payment system.
Suffix	Unique three-letter identifier for the payment system.	<p>Must not be any of the following reserved suffixes seeded in Oracle Payments:</p> <ul style="list-style-type: none"> • lop (sample gateway system) • efs (Concord EFSnet) • ptk (Paymentech) • fdb (FDC North)
Servlet Base URL	The URL of the payment system servlet.	Must be an URL in this form: http://<host>:<port>/<servlet zone>

All of these attributes can be set in the Create/Update Payment System page.

Related Topics

For information on setting up payment systems, see Step 6. Setting Up Payment Systems, page 4-13.

Account Options

Account options are attributes defined by the payment system for its user accounts. First, you must define the parameters in the Payment System page. Then, you enter the actual values when you create a payment system account in the Payment System Account page. Account options are used for payment instruction file generation and are represented in the funds capture extract.

Note: To implement the custom payment system integrations, ensure

that you document the values.

Related Topics

To enable payment system accounts, see [Enabling Payment System Accounts](#), page 5-14

Formats

A payment system uses different formats during transaction processing. For example, there may be one format for authorization and another for settlement batch. Before creating a format in Oracle Payments, a corresponding Oracle XML Publisher template entity must be available.

Developing a Format Template

The table below describes the Oracle XML Publisher template attributes fixed by Oracle Payments:

Oracle XML Publisher Template Attributes Fixed by Oracle Payments

Attribute	Description	Constraint
Data Definition	The template's data definition determines the structure of the data to which the template is applied.	Always equal to: IBY_FNDcpt_INSTRUCTIO N_1_0 for funds capture. Always equal to: IBY_FD_INSTRUCTION_1_0 for funds disbursement.

For implementing the custom payment system integrations, a payment instruction template must be developed for XML Publisher using one of the supported XML Publisher template types such as eText (RTF) or XSL. You will need to seed your XML Publisher template and link your format to the seeded template.

Setting Up a Format Template

Once an Oracle XML Publisher template is created, the corresponding formats entity must be created in Oracle Payments, using the [Create Format](#) page.

Extract Generator

The extract generator produces the payment file extract document, a superset of all data pertaining to the transaction. A format template is applied to the extract to produce a final payment file.

The payment file extract is an XML document whose structure conforms to an XML schema. The funds capture XML schema is defined in file \$IBY_TOP/patch115/publisher/defs/IBY_FCI_1_0.xsd. The funds disbursement XML schema is defined in file \$IBY_TOP/patch115/publisher/defs/IBY_PPIOUT_1_0.xsd.

The funds capture XML schema supports transactions for all funds capture instrument types, such as credit card, bank account, PINless debit card and all funds capture transaction operation types, such as authorization, online settlement, and settlement batch.

The funds disbursement XML schema contains all disbursement-related payment information, including payment instruction information, a payment process profile, a payment format, and constituent payments.

For implementing the custom payment system integrations, the structure of the extract must be thoroughly understood to create the payment instruction file templates.

Extract Formatter

The extract formatter takes the extract document produced by the extract generator and applies a format template to produce the final payment file. Oracle Payments uses the Oracle E-Business Suite application Oracle XML Publisher (XDO) as its formatting engine.

For implementing the custom payment system integrations, templates must be created for every transaction operation type supported by the payment system. If the integration model for the payment system does not conform to one of formatted payment file delivery, you can use an ordinary formatting template that produces the unchanged extract documents as its output, deliver the extract to a servlet using HTTP, and then extract document to the payment system's native payment request mechanism in the servlet map. An ordinary template is already seeded by Oracle Payments with a template code IBY_IDENTITY. IBY_IDENTITY is a special case of a template that does no formatting and, instead, returns the extract as is.

Transmission Functions

The transmission function is responsible for delivering the payment file to the payment system and implements a particular transmission protocol.

For implementing the custom payment system integrations, transmission function code-points must be created for each transmission protocol for communicating with the

payment system.

One component of a payment system specification is the transmission protocol used to deliver payment files to the payment system. A transmission protocol has transmission parameters associated with it that define the required system data when making a communication attempt. A transmission protocol also has a defined transmission function code-point, which is a self-contained unit of code implementing the protocol and conforming to the interface: `oracle.apps.iby.net.TransmitFunction`.

Developing a Transmission Function

A transmission protocol is implemented through a Java class which must implement the interface `oracle.apps.iby.net.TransmitFunction`. To implement the interface, the class must define function `transmit`.

The table explains the signature of the function:

Argument Type	Type	Description
params	<code>java.util.Dictionary</code>	The map of protocol parameter names and values representing the transmission configuration. The names are taken from the parameter name definitions for the protocol.
payload	<code>java.io.InputStream</code>	The message payload being delivered.
<return>	<code>java.io.InputStream</code>	The response to the transmission; maybe null.

On implementing the protocol in function `transmit`, the function should handle any system exception that occurs by the exception of type `oracle.apps.iby.exception.PSEException` such as:

```
throw new PSEException(PSEException.COMMUNICATION_ERROR);
```

If a mandatory transmission protocol parameter is not set, an exception using the same class type should be thrown such as:

```
throw new PSEException(PSEException.CODEPOINT_ARG_ERR,  
PSEException.CODEPOINT_ARG_ERR_TOKEN_ARG,  
    <parameter code>);
```

Because the transmission function is invoked through Java reflection APIs, special action must be taken to preserve any exceptions generated by the transmission function; or else the Oracle Payments engine only sees a generic `java.lang.reflect`.

InvocationTargetException when the function fails. To preserve the original exception for the engine to display, the following function should be called:

```
oracle.apps.iby.engine.CodePoint.storeException()
```

Setting Up a Transmission Protocol

When configuring a payment system account for the payment system in the user interface, you will automatically see all transmission protocol parameters defined for the payment system's system payment profile.

The table lists the attributes of a transmission protocol:

Attributes of a Transmission Protocol

Attribute Name	Description	Constraint
TRANSMIT_PROTOCOL_CODE	The unique identifier for the protocol.	Must be unique and less than or equal to 30 characters.
TRANSMIT_PROTOCOL_NAME	The description of the protocol.	
TRANSMIT_CODE_LANGUAGE	The language in which the transmission function code-point for the protocol is implemented.	Must be JAVA.
TRANSMIT_CODE_PACKAGE	The code-point package. The fully-qualified class name of the transmission function code-point.	A fully qualified class name.
TRANSMIT_CODE_ENTRY_POINT	The code-point entry-point: the programming language function name that is called.	Must equal: transmit.

The table defines the attributes of a transmission protocol's parameters, sub-entities of the protocol.

Attributes of a Transmission Protocol's Parameters

Attribute Name	Description	Constraints
TRANSMIT_PARAMETER_CODE	The identifier for the parameter.	Must be unique and less than or equal to 30 characters among all parameters defined for the protocol.
TRANSMIT_PARAMETER_TYPE	The datatype of the parameter.	Supported values are: <ul style="list-style-type: none">• VARCHAR2• NUMBER
MANDATORY_FLAG	Whether the parameter is mandatory; used for user interface validation during transmission configuration.	Possible values: <ul style="list-style-type: none">• Y (Yes)• N (No)
TRANSMIT_PROTOCOL_CODE	The protocol to which this parameter belongs.	
TRANSMIT_PARAMETER_NAME	The name of the parameter.	

A transmission protocol and its parameters can be seeded in Oracle Payments by creating a SQL script to insert them.

Note: Insert all but translatable protocol attribute
TRANSMIT_PROTOCOL_NAME name into tables
IBY_TRANSMIT_PROTOCOLS_B. Insert the translatable attribute
TRANSMIT_PROTOCOL_NAME into
IBY_TRANSMIT_PROTOCOLS_TL and call the procedure
IBY_PP_MLSUTL_PVT.TRANS_PROT_ADD_LANGUAGE.

Insert all but translatable protocol parameter attribute
TRANSMIT_PARAMETER_NAME name into tables
IBY_TRANSMIT_PARAMETERS_B. Insert the translatable attribute
TRANSMIT_PARAMETER_NAME into
IBY_TRANSMIT_PARAMETERS_TL and call the procedure
IBY_PP_MLSUTL_PVT.TRANS_PARAM_ADD_LANGUAGE.

Acknowledgment Parser

A payment system sends an acknowledgement upon receiving a funds capture instruction delivery. The task of the acknowledgment parser is to parse the response understandable to Oracle Payments.

For implementing the custom payment system integrations, acknowledgement parsers must be created for every transmission function. If the payment system does not support acknowledgement for a protocol, a default acknowledgement parser must still be created which returns default or trivial values.

Acknowledgement parsers are self-contained code-points that parse responses from the payment system into a form that can be processed by the Oracle Payments engine.

Seeding an Acknowledgement Parser

The table defines the attributes when seeding an acknowledgement parser:

Attributes when Seeding an Acknowledgement Parser

Attribute Name	Description	Constraint
ACK_READER_CODE	The unique identifier for the parser.	Must be unique and less than or equal to 30 characters.
READER_CODE_LANGUAGE	The language that the transmission function code-point for the parser is implemented.	Must be Java.
READER_CODE_PACKAGE	The code-point package- the fully-qualified class name of the acknowledgement parser code-point.	A fully qualified class name.
READER_CODE_ENTRY_POINT	The code-point entry-point: the programming language function name that is called.	The method name must be <i>parse</i> .

After the attributes for an acknowledgement parser are defined, they can be seeded in Oracle Payments by creating a SQL script to insert them into the table `IBY_ACK_READERS`.

For implementing the custom payment system integrations, the Java class implementing an acknowledgment parser must be distributed to the user and then place it in the CLASSPATH of the application server hosting the Oracle Payments

engine.

Developing an Acknowledgement Parser

All acknowledgement parsers must sub-class the interface: `oracle.apps.iby.bep.ACKParser`. The interface has a single function, `parse`, with these interface:

Argument Name	Type	Description
<code>ackFile</code>	<code>java.io.InputStream</code>	The acknowledgment message or file.
<code>hints</code>	<code>java.util.Dictionary</code>	Collection of name-values providing information about the transaction the acknowledgement is for. For example, the instrument type used, or credit card issuer.
<code><return></code>	<code>bep.ACK</code>	A corresponding object for the acknowledgment.

The `hints` argument to the acknowledgement parser is a collection of name-value pairs providing information about the transaction the response was created for.

The table below lists the possible values in this collection:

Possible Values in a Collection of Name-Value Pairs

Hint Key	Description	Value
ACKParser. CARD_ISSUER_HINT	The card issuer; for acknowledgments where the structure of the response varies based upon the card issuer.	One of the following card issuer codes: <ul style="list-style-type: none">• AMEX• DINERS• DISCOVER• ENROUTE• JCB• MASTERCARD• UNKNOWN• VISA
ACKParser. INSTR_TYPE_HINT	The transaction instrument type.	One of the following values: <ul style="list-style-type: none">• BANKACCOUNT• CREDITCARD• PINLESSDEBITCARD• PURCHASECARD

Hint Key	Description	Value
ACKParser. TRXN_TYPE_ID_HINT	The type of transaction.	One of the following values: <ul style="list-style-type: none"> • 2 (Auth only) • 3 (Auth capture) • 5 (Return) • 8/9 (Capture/Settlement) • 11 (Credit) Value data-source is IBY_TRXN_SUMMARIE S_ALL.TRXNTYPEID

Note: The hints are populated only for certain transaction types. For example, the hints will be not be populated for a batch close operation.

The result of an acknowledgement parser returns is an object derived from class: `oracle.apps.iby.bep.ACKParser`. This class is a record intended to hold various response fields mapped from the payment system response.

ACK

The abstract class `oracle.apps.iby.bep.ACK` defines the most basic acknowledgement attributes inherited by all sub-classes. The table describes the structure of this class and all derived sub-classes.

Note: Implicitly, for each attribute listed for a particular class there exists `get<AttributeName>` and `set<AttributeName>` functions for accessing the attributes. The `get<AttributeName>` returns an object of the attribute's type and the `set<AttributeName>` takes an object of the attribute's type as its single argument.

Attribute Name	Type	Description
BEPErrorCode	String	Payment system error code.

Attribute Name	Type	Description
BEPErrorMessage	String	Payment system error message.

TrxnACK

The abstract class `oracle.apps.iby.bep.TrxnACK` defines common attributes for transaction acknowledgements. It is a subclass of `oracle.apps.iby.bep.ACK` and hence inherits its attributes as well.

Attribute Name	Type	Description
OrderId	String	Order identifier for this transaction.
TrxnStatus	int	<p>Status of the transaction. The possible values are:</p> <ul style="list-style-type: none"> • 0 (Success) • 1 (Communication error) • 2 (Duplicate order id) • 3 (Duplicate batch id) • 4 (Required field missing) • 5 (Payment system error) • 8 (Operation not supported) • 11 (Pending) • 20 (Declined)
TrxnDate	java.util.Date	Date the transaction was completed.

Attribute Name	Type	Description
TrxnReqType	String	<p>Transaction request type. The possible values are:</p> <ul style="list-style-type: none"> • ORAPMTCAPTURE (Capture/Settlement) • ORAPMTCLOSEBATCH (Batch close) • ORAPMTCREDIT (Credit) • ORAPMTREQ (Authorization/Auth Capture/Verification/Debit) • ORAPMTRETURN (Return) • ORAPMTVOID (Void)
ExtensibilitySet	oracle.apps.iby.util. NameValuePair[]	The collection of extensibility name-value pairs.

Attribute ExtensibilitySet is typed as an array of `oracle.apps.iby.util.NameValuePair` objects and may be optionally set by the parser if the payment system acknowledgement contains important data that do not correspond to any of the attributes in an appropriate acknowledgements object. Any name values returned by the ExtensibilitySet attributed are stored in table IBY_TRXN_EXTENSIBILITY and are included in the extract document of any follow on transaction using the series of Extend sub-elements which may appear beneath the OriginalCCTransaction or OriginalDCTransaction elements.

The table below explains the attributes of the `oracle.apps.iby.util.NameValuePair` class.

Attribute Name	Type	Description
Name	String	Name/key.
Value	String	Value.

CreditCardTrxnACK

The class `oracle.apps.iby.bep.CreditCardTrxnACK` holds acknowledgment information for a single credit card transaction. The table lists the attributes (not including ones derived from its parent class `oracle.apps.iby.bep.TrxnACK`).

Attribute Name	Type	Description
AuthCode	String	Authorization code.
AVSResponse	String	Address verification system response from the payment system.
SecurityCodeCheck	String	Payment system response for the credit card security code (CVV2) check.
RefCode	String	Reference code.

BankAccountTrxnACK

The class `oracle.apps.iby.bep.BankAccountTrxnACK` holds acknowledgment information for a single bank account transaction. The table below lists the attributes (including inherited ones from `oracle.apps.iby.bep.TrxnACK`).

Attribute Name	Type	Description
RefCode	String	Bank reference code.
TrxnAmount	<code>oracle.apps.iby.ecapp.Price</code>	Actual amount of the transfer.
PostDate	<code>java.util.Date</code>	Date funds will be posted.
FundsCommitted	Boolean	Indicates whether funds were committed.

BatchACK

The abstract class `oracle.apps.iby.bep.BatchACK` defines common attributes of batch acknowledgments. The table below list the attributes, in addition to those

inherited from class `oracle.apps.iby.bep.ACK`.

Attribute Name	Type	Description
BatchId	String	Identifier of the batch.
BatchStatus	int	Status of the batch. The possible values are: <ul style="list-style-type: none">• 0 (Success)• 1 (Communication error)• 3 (Duplicate batch id)• 5 (Payment system error)• 11 (Pending)
BatchDate	java.util.Date	Date of batch submission.
TrxnACKs	bep.TrxnACK[]	Collection of acknowledgments for the individual transactions in this batch.

Attribute Name	Type	Description
TrxnACKType	String	<p>Enumerated value indicating transaction acknowledgments. The values are:</p> <ul style="list-style-type: none"> • BatchACK. TRXN_ACK_ALL (All transactions in the batch have an acknowledgement) • BatchACK. TRXN_ACK_POSITIVE (All transactions missing an acknowledgement assumed failed) • BatchACK. TRXN_ACK_NEGATIVE (All transactions missing an acknowledgement assumed successful)

Note: All batch statuses except pending (11) are final. In case a batch acknowledgement does not exist, for example immediately after a batch close, or during a batch query which occurs before the batch has been processed, a batch acknowledgement object with batch status 0 should be created although there is no existing acknowledgement file.

CreditCardBatchACK

The class `oracle.apps.iby.bep.CreditCardBatchACK` extends `oracle.apps.iby.bep.BatchACK` and contains attributes for credit card batch acknowledgments. The table below explains the attributes.

Attribute Name	Type	Description
AuthTotal	<code>oracle.apps.iby.ecapp.Price</code>	Total authorizations completed.

Attribute Name	Type	Description
CaptureTotal	oracle.apps.iby.ecapp.Price	Total settlements completed.
CreditTotal	oracle.apps.iby.ecapp.Price	Total credits completed.

BankAccountBatchACK

The class `oracle.apps.iby.bep.BankAccountBatchACK` extends `oracle.apps.iby.bep.BatchACK` and contains attributes for credit card batch acknowledgments. The table below describes the attributes.

Attribute Name	Type	Description
CreditTotal	ecapp.Price	Total credits completed.
DebitTotal	ecapp.Price	Total debits completed.

Creating Custom Disbursement Validation Sets

Validation sets are pieces of code that are executed at various processing points by the Build Payments program. Each validation set applies a set of business rules and determines whether the payment entity in consideration passes all the rules or not. Unless all rules are passed, the validation set returns a failure status to the Build Payments Program.

Validation Set Level

Before you create a validation set, you must determine at what level the validation applies. The validation set level, also known as the validation point determines the entity that needs to be validated. The three validation levels are as follows:

- Document
- Payment
- Disbursement Instruction

Validation Code Entry Point

The Build Payments program dynamically invokes the validation set linked to a

particular entity. The validation set entry point is seeded in the `IBY_VALIDATION_SETS_VL` view. The important columns pertaining to validation code entry point are listed in the table below, which stores seeded validation sets.

IBY_VALIDATION_SETS_VL

Column Name	Example	Meaning
VALIDATION_SET_CODE	PT_CHECK_GEN	Unique name of the validation set used for querying.
VALIDATION_LEVEL_CODE	DOCUMENT	This validation is applicable to documents payable (invoices).
VALIDATION_CODE_PACKAGE	IBY_VALIDATIONSETS_CALLS_PUB	This validation set is stored in this package.
VALIDATION_CODE_ENTRY_POINT	PT_CHECK_GEN	This validation set has this method name.
VALIDATION_CODE_LANGUAGE	PL/SQL	This validation set is implemented in PL/SQL.

For the same validation set data shown in the table above, the Build Payments program invokes the PL/SQL validation set `IBY_VALIDATIONSETS_CALLS_PUB`. `PT_CHECK_GEN` dynamically to validate a document payable.

Note: You should create the custom validation set in a custom package. You should not add your custom validation sets into the `IBY_VALIDATIONSETS_CALLS_PUB` package, since this package is meant for Oracle Payments only.

Linking a Validation Set To a Payment Entity

Typically, validation sets are linked to one of two payment entities as follows:

- Method
- Format

Validation sets linked to the payment method can also be applied online, such as validating the invoice in Oracle Payables at the time of invoice creation. Online validations do not fail the document, but present the user with warning messages

before submitting the document to Oracle Payments. This provides you with the ability to correct errors before the document is submitted to Oracle Payments in the document validation cycle. When online validations are performed on an invoice, it is put on hold if there are validation errors. The errors generated during online validation are shown as hold reasons, and the selection program will not pick up the invoice until the hold reasons are corrected. Validation sets linked to payment methods are re-applied when the documents payable are submitted to Oracle Payments.

The vast majority of seeded validation sets are linked to formats. The link between validation sets and payment methods/formats is stored in the IBY_VAL_ASSIGNMENTS table, which stores the link between a validation set and a payment entity.

IBY_VAL_ASSIGNMENTS

Column Name	Example	Meaning
VALIDATION_SET_CODE	PT_CHECK_GEN	Unique name of the validation set used for querying.
VAL_ASSIGNMENT_ENTITY_TYPE	FORMAT	Type of payment entity to which this validation set is linked. This will be FORMAT or METHOD.
ASSIGNMENT_ENTITY_ID	IBY_PAY_CHK_PT	The actual entity to which this validation set is linked. In this example, the entity is the format IBY_PAY_CHK_PT.
TERRITORY_CODE	PT	If null, this validation set is applicable, regardless of the payment country. If not null, this validation is applicable only to the specified country.

The preceding table shows that the validation set PT_CHECK_GEN is linked to the payment format IBY_PAY_CHK_PT. Since the validation set PT_CHECK_GEN is applicable at the document level, the Build Payments program executes this validation whenever a document payable comes in with the format IBY_PAY_CHK_PT.

Validation Set Signature

All validation sets contain the same five-argument signature. The third parameter is the payment entity to be validated. It can be a document payable, a payment, or a payment

instruction, depending on the validation level for which you have coded your validation.

Document Level Validation Set

Document-level validations will be implemented in PL/SQL and must have the following signature:

Name	Data Type	Type	Description
p_validation_assign_id	NUMBER	IN	The validation assignment id. This value is from IBY_VAL_ASSIGNMENTS validation_assignment_id.
p_validation_set_code	VARCHAR2	IN	The validation set that needs to be invoked. This value is from IBY_VALIDATION_SETS_VL validation_set_code.
p_document_id	NUMBER	IN	Unique id of the document to be validated. This value is from IBY_DOCS_PAYABLE_ALL document_payable_id.
p_is_online_val	VARCHAR2	IN	Y/N flag indicating whether this is an online validation call.

Name	Data Type	Type	Description
x_result	NUMBER	OUT	Numeric value indicating the result of the applied validations. Zero (0) indicates that all validations were successful; 1 indicates that at least one validation failed.

```

PROCEDURE MY_VALIDATION_SET(
    p_validation_assign_id IN
        IBY_VAL_ASSIGNMENTS.
            validation_assignment_id%TYPE,
    p_validation_set_code IN
        IBY_VALIDATION_SETS_VL.
            validation_set_code%TYPE,
    p_document_id IN
        IBY_DOCS_PAYABLE_ALL.
            document_payable_id%TYPE,
    p_is_online_val IN VARCHAR2,
    x_result OUT NOCOPY NUMBER
);

```

The preceding example shows a validation set called MY_VALIDATION_SET that is applicable at the document level. When you create your document level validation, you need to use the same signature.

Payment Level Validation Set

Payment level validations will be implemented in PL/SQL and must have the following signature:

Name	Data Type	Type	Description
p_validation_assign_id	NUMBER	IN	The validation assignment id. This value is from IBY_VAL_ASSIGNMENTS validation_assignment_id.

Name	Data Type	Type	Description
p_validation_set_code	VARCHAR2	IN	The validation set that needs to be invoked. This value is from IBY_VALIDATION_SETS_VL validation_set_code.
p_payment_id	NUMBER	IN	Unique id of the document to be validated. This value is from IBY_PAYMENTS_ALL payment_id.
p_is_online_val	VARCHAR2	IN	Y/N flag indicating whether this is an online validation call.
x_result	NUMBER	OUT	Numeric value indicating the result of the applied validations. Zero (0) indicates that all validations were successful; 1 indicates that at least one validation failed.

```

PROCEDURE MY_VALIDATION_SET(
    p_validation_assign_id IN
        IBY_VAL_ASSIGNMENTS.
            validation_assignment_id%TYPE,
    p_validation_set_code IN
        IBY_VALIDATION_SETS_VL.
            validation_set_code%TYPE,
    p_document_id IN
        IBY_DOCS_PAYABLE_ALL.
            document_payable_id%TYPE,
    p_is_online_val IN VARCHAR2,
    x_result OUT NOCOPY NUMBER
);

```

The preceding example shows a validation set called MY_VALIDATION_SET that is applicable at the payment level. When you create your payment level validation set, you need to use the same signature.

Payment Instruction Level Validation Set

Payment instruction-level validations are implemented in PL/SQL and must have the following signature:

Name	Data Type	Type	Description
p_validation_assign_id	NUMBER	IN	The validation assignment id. This value is from IBY_VAL_ASSIGNMENTS validation_assignment_id.
p_validation_set_code	VARCHAR2	IN	The validation set that needs to be invoked. This value is from IBY_VALIDATION_SETS_VL validation_set_code.
p_instruction_id	NUMBER	IN	Unique id of the payment instruction to be validated. This value is from IBY_PAY_INSTRUCTIONS_ALL payment_instruction_id.
p_is_online_val	VARCHAR2	IN	Y/N flag indicating whether this is an online validation call.
x_result	NUMBER	OUT	Numeric value indicating the result of the applied validations. Zero (0) indicates that all validations were successful; 1 indicates that at least one validation failed.

```

PROCEDURE MY_VALIDATION_SET(
    p_validation_assign_id  IN
        IBY_VAL_ASSIGNMENTS.
            validation_assignment_id%TYPE,
    p_validation_set_code    IN
        IBY_VALIDATION_SETS_VL.
            validation_set_code%TYPE,
    p_instruction_id         IN
        IBY_PAY_INSTRUCTIONS_ALL.
            payment_instructionid%TYPE,
    p_is_online_val          IN VARCHAR2,
    x_result                 OUT NOCOPY NUMBER
);

```

The preceding example shows a validation set called MY_VALIDATION_SET that is applicable at the payment instruction level. When you create your payment instruction level validation set, you need to use the same signature.

Important Notes Regarding Validation Sets

Note: The p_is_online_val is only relevant for document level validation sets. It should be ignored when implementing payment and payment instruction level validation sets. When validating an invoice in Oracle Payables, either manually or via invoice interface import program, Payables invokes the validation sets attached to the payment method on the document. Oracle Payables invokes the validation set as an online validation to catch possible errors when the invoice is created. Therefore, Oracle Payables invokes the validation set with the p_is_online_val set to Y. This means that the validation error messages are displayed in the schedule payment of the invoice as warnings and the schedule payment is put on hold. This is applicable only to validations assigned to payment methods as it is known at invoice time and Oracle Cash Management validations attached to bank accounts.

Note: The x_result parameter is the return value of the validation set that is passed back to the Build Payments program. A value of zero (0) indicates the validation was successful. A value of 1 indicates the payment entity failed validation.

Storing Errors Generated by a Validation Set

The error messages generated by applying the business rules encapsulated within the validation set needs to be displayed to the user.

The IBY_TRANSACTION_ERRORS table shown below stores error messages generated by a validation set. Since you can have multiple error messages associated with the payment entity being validated, this table can contain multiple rows for a given payment entity id.

IBY_TRANSACTION_ERRORS

Column Name	Example	Meaning
TRANSACTION_ERROR_ID	21412	Unique sequence number for this error message.
TRANSACTION_TYPE	DOCUMENT_PAYABLE	Type of payment entity for which this error message has been generated. Possible values include DOCUMENT_PAYABLE, PAYMENT, and PAYMENT_INSTRUCTION.
TRANSACTION_ID	631	Actual payment entity to which this error message is linked. In this example, the entity is a document payable with id 631.
ERROR_CODE	IBY_DOC_INVALID_PROFILE	This error code is used to retrieve a seeded FND error message.
VALIDATION_SET_CODE	CORE_DOC_VALIDATION	The validation set that generated this message.

Implementing a Document Level Validation Set

A document level validation set needs to follow the following process:

1. Retrieve all attributes of the provided document. The document id is provided as an input parameter to document level validation sets.
2. Create an error record for inserting into the IBY_TRANSACTION_ERRORS table.
3. Apply business rule using the relevant document attributes.
4. If the document has failed validation, populate an error record to reflect that.
5. Repeat Steps 3 and 4 until all business rules are applied.
6. If all validations have been successfully passed, return 0.

7. If any validation has failed, insert the error records into the `IBY_TRANSACTION_ERRORS` table and return 1.

Helper Functions

IBY has implemented a number of helper functions for some of the preceding steps. You can use these functions to shorten your coding time.

- For Step 1, you can use `IBY_VALIDATIONSETS_PUB.initDocumentData` to retrieve all the document attributes. If you need additional attributes that are not available via this method, you will need to write your own PL/SQL logic to retrieve them from the `IBY_DOCS_PAYABLE_ALL` table.
- For Step 3, you can use the method `IBY_VALIDATIONSETS_PUB.evaluateCondition`. This method takes in some pre-defined conditions like `EQUALSTO`, `NOTEQUALSTO`, `NOTNULL`, or `LENGTH`, `MAXLENGTH`. You can implement simple business rules using the method above. Any complex ones must be implemented as custom code.
- For Step 4, use the method `IBY_VALIDATIONSETS_PUB.insertIntoErrorTable`. This method inserts the given error record into a PL/SQL memory structure that can be used for a bulk insert into the transaction errors table.
- For Step 7, use the method `IBY_VALIDATIONSETS_PUB.insert_transaction_errors` to perform a bulk insert of error messages into the `IBY_TRANSACTION_ERRORS` table. This method should be invoked once at the end of your validation set.

Implementing a Payment Level Validation Set

A payment level validation set needs to follow the following process:

1. Retrieve all attributes of the provided document. The document id is provided as an input parameter to payment level validation sets.
2. Create an error record for inserting into the `IBY_TRANSACTION_ERRORS` table.
3. Apply business rule using the relevant payment attributes.
4. If the document has failed validation, populate an error record to reflect that.
5. Repeat Steps 3 and 4 until all business rules are applied.
6. If all validations have been successfully passed, return 0.
7. If any validation has failed, insert the error records into the `IBY_TRANSACTION_ERRORS` table and return 1.

Helper Functions

Oracle Payments has implemented a number of helper functions for some of the preceding steps. You can use these helper functions to shorten your coding time.

- For Step 1, you can use *IBY_VALIDATIONSETS_PUB.initDocumentData* to retrieve some of the document attributes. If you need additional attributes that are not available via this method, you will need to write your own PL/SQL logic to retrieve them from the *IBY_PAYMENTS_ALL* table.
- For Step 3, you can use the method *IBY_VALIDATIONSETS_PUB.evaluateCondition*. This method takes in some pre-defined conditions like *EQUALSTO*, *NOTEQUALSTO*, *NOTNULL*, *LENGTH*, or *MAXLENGTH*. You can implement simple business rules using the method above. Any complex ones must be implemented as custom code.
- For Step 4, use the method *IBY_VALIDATIONSETS_PUB.insertIntoErrorTable*. This method inserts the given error record into a PL/SQL memory structure that can be used for a bulk insert into the transaction errors table.
- For Step 7, use the method *IBY_VALIDATIONSETS_PUB.insert_transaction_errors* to perform a bulk insert of error messages into the *IBY_TRANSACTION_ERRORS* table. This method should be invoked once at the end of your validation set.

Implementing a Payment Instruction Level Validation Set

A payment instruction level validation set needs to follow the following process:

1. Retrieve all attributes of the provided payment instruction. The payment id is provided as an input parameter to payment instruction level validation sets.
2. Create an error record for inserting into the *IBY_TRANSACTION_ERRORS* table.
3. Apply business rule using the relevant payment instruction attributes.
4. If the payment has failed validation, populate the error record and insert it into memory.
5. Repeat Steps 3 and 4 until all business rules are applied.
6. If all validations have been successfully passed, return 0.
7. If any validation has failed, insert the error records in memory into the *IBY_TRANSACTION_ERRORS* table and return 1.

Helper Functions

Oracle Payments has implemented a number of helper functions for some of the preceding steps. You can use these helper functions to shorten your coding time.

- For Step 1, you can use *IBY_VALIDATIONSETS_PUB.initDocumentData* to retrieve all the document attributes. If you need additional attributes that are not available via this method, you will need to write your own PL/SQL logic to retrieve them from the *IBY_DOCS_PAYABLE_ALL* table.
- For Step 3, you can use the method *IBY_VALIDATIONSETS_PUB.evaluateCondition*. This method takes in some pre-defined conditions like *EQUALSTO*, *NOTEQUALSTO*, *NOTNULL*, *LENGTH*, or *MAXLENGTH*. You can implement simple business rules using the method above. Any complex ones must be implemented as custom code.
- For Step 4, use the method *IBY_VALIDATIONSETS_PUB.insertIntoErrorTable*. This method inserts the given error record into a PL/SQL memory structure that can be used to perform a bulk insert into the transaction errors table.
- For Step 7, use the method *IBY_VALIDATIONSETS_PUB.insert_transaction_errors* to perform a bulk insert of error messages into the *IBY_TRANSACTION_ERRORS* table. This method should be invoked once at the end of your validation set.

Implementing Custom Validation Logic

As much as possible, use the method *IBY_VALIDATIONSETS_PUB.evaluateCondition* for all your basic validations. By passing the appropriate token value to this method, you can cover most of the common validation conditions, such as checking if a field is null or not null and verifying that an amount value is less than or greater than a specified limit.

In case the validation required is complex and cannot be implemented using the normal token operators, then use the special token *CUSTOM* in the call to *evaluateCondition*. When the *CUSTOM* token is used, the *evaluateCondition* dynamically invokes a specified validation method. The method to be invoked should be passed as an argument to *evaluateCondition*.

Example

The following is an example of invoking *evaluateCondition* with a *CUSTOM* token.

```

IBY_VALIDATIONSETS_PUB.evaluateCondition
(
  'PAYEE_BANK_SWIFT_CODE',
  l_payee_bank_swift_code,
  'CUSTOM', /* token (operator) is 'CUSTOM' */
  'DFPML.VAL_SWFT_CD', /* custom method to be invoked */
  null,
  l_valResult,
  l_docErrorRec
);

```

The field name and field value parameters to *evaluateCondition* will be passed to your custom validation method. These are the first two arguments to the *evaluateCondition* method. The custom validation method should specify the success or failure of the validation using the third parameter which is an out parameter. If the out parameter is 0, it means that the custom validation was a success (failure otherwise).

Your custom validation method should follow this signature:

```

PROCEDURE <custom validation name>(
  p_fieldName VARCHAR2,
  p_fieldValue VARCHAR2,
  l_chr OUT NOCOPY VARCHAR2
);

```

For example, your custom validation method could look like:

```

PROCEDURE VAL_SWFT_CD(
  p_fieldName VARCHAR2,
  p_fieldValue VARCHAR2,
  l_chr OUT NOCOPY VARCHAR2
)
IS
BEGIN
  /*
   * Do some validations here on the field value.
   */

  :
  :

  /*
   * Return '0' if everything is Ok. Else
   * return '1'.
   */
  l_chr := '0'; /* success */
END VAL_SWFT_CD;

```

Example Code

For examples of how validation sets are implemented, note the following files:

```

$IBY_TOP/patch/115/sql/ibyvalcs.pls
$IBY_TOP/patch/115/sql/ibyvalcb.pls

```

The above two files are the PL/SQL specification and body for the validation sets seeded by Oracle Payments. The body contains validation sets at document, payment and payment instruction levels. You are encouraged to reuse as much of the code as possible.

Seed Your Validation Sets

After you create a validation set, remember to seed your validation set in the IBY_VALIDATION_SETS_VL table.

You cannot directly insert your newly created validation set into IBY_VALIDATION_SETS_VL because it is a view. You need to perform the insert using the following steps:

1. Insert your validation set into the IBY_VALIDATION_SETS_B table. This is the base table that does not contain any translatable text.

Example:

```
insert into iby_validation_sets_b (validation_set_code,
validation_level_code,validation_code_package,
validation_code_entry_point,validation_code_language,created_by,
creation_date,last_updated_by,last_update_date,last_update_login,
object_version_number) values
('paymul_pmt_validations','PAYMENT','mis_iby_val_pkg','mis_iby_gen_v
al_pmt_pml','plsqli','294110',sysdate,'294110',sysdate,'294110',1);
```

2. Insert your validation set into the IBY_VALIDATION_SETS_TL table. This table contains translatable text. The validation set name is translatable.

Example:

```
insert into iby_validation_sets_tl (validation_set_code,language,
source_lang,validation_set_display_name,created_by,creation_date,
last_updated_by,last_update_date,last_update_login,
object_version_number) values
('paymul_pmt_validations','us','us','paymul eft validation-payment
level','294110',sysdate,'294110',sysdate,'294110',1);
```

Now your validation sets will be visible in the Oracle Payments Setup page. The navigation path is Oracle Payments Setup > Payments Setup > Shared Setup > Validations.

You can link your validation set to appropriate payment entities like payment methods and payment formats. After this point, you should be able to test your validation logic by running the Build Payments Program.

Important Notes for Validation Set Implementers

The following are important notes for validation set implementers:

- Do not include a COMMIT or a ROLLBACK call within your validation set. The validation engine will perform a COMMIT after invoking the validation set.
- You may include DML commands to manipulate or query data from other tables from within your custom validation set. However, note that any changes you make will be committed by the validation engine after the validation set has been invoked.

- Do not include any DDL commands that create, alter, or drop tables or views in your validation set. DDL statements issue implicit commits.
- Note that validation sets will be invoked once per payment entity. A document level validation set will be invoked once per document in a payment process request. A payment level validation set will be invoked once per created payment. Therefore, it is in your interest to make your validation set simple and efficient. Otherwise, your validation set may impose a significant performance cost.
- You cannot directly insert your newly created validation set into IBY_VALIDATION_SETS_VL as this is a view. You need to do this insert in two steps: First, into the base table, IBY_VALIDATION_SETS_B, and then into the translatable table, IBY_VALIDATION_SETS_TL.
- It is recommended that your validation set perform validations by invoking the helper function *IBY_VALIDATIONSETS_PUB.evaluateCondition* as much as possible.

Your validation set should also contain the following two calls.

```
/*
 * Inserts the error generated by a single
 * validation within the validation set into
 * the l_docErrorTab temporary PLSQL table.
 *
 * Similarly, you can have l_pmtErrorTab
 * for payments.
 *
 * This call should be invoked once after each
 * validation to accumulate all the error
 * messages in l_docErrorTab.
 */
IBY_VALIDATIONSETS_PUB.insertIntoErrorTable(l_docErrorRec,
l_docErrorTab);

/*
 * Permanently inserts all the error messages
 * in l_docErrorTab into the
 * IBY_TRANSACTION_ERRORS table.
 *
 * This call should be invoked only once at
 * the end of your validation set.
 */
IBY_VALIDATIONSETS_PUB.insert_transaction_errors(p_is_online_val,
l_docErrorTab);
```

Creating Custom Funds Capture Validation Sets

A validation set is a program unit which performs a set of validations against a payment processing entity, typically a fund capture order (payment transaction) or a payment batch. Each validation sets applies a set of business rules and determines whether the payment entity in consideration passes all the rules or not. Funds capture validation sets can be implemented as Java functions or PL/SQL procedures. They are invoked dynamically by the payments engine.

Validation Set Level

Before you create a validation set, you must determine at what level the validation applies. The validation set level, also known as the validation point, determines the entity that needs to be validated. The two validation levels are as follows:

- Order
- Instruction

Order level validations are transaction level validations. Instruction level validations are batch level validations. Order level validations are implemented in Java and instruction level validations are implemented in PL/SQL.

Validation Code Entry Point

The payments engine dynamically invokes the validation set linked to a particular entity. The validation set entry point is seeded in the `IBY_VALIDATION_SETS_VL` view. Columns that are concerned with the validation code entry point are listed in the table below. The `IBY_VALIDATION_SETS_VL` table stores seeded validation sets.

IBY_VALIDATION_SETS_VL

Column Name	Example	Meaning
VALIDATION_SET_CODE	PTK_ONLINE_7_2_BATCH_2_1_0	Unique name of the validation set used for querying.
VALIDATION_LEVEL_CODE	ORDER	This validation is applicable to orders (transactions).
VALIDATION_CODE_PACKAGE	oracle.apps.iby.bep.proc.paymentech. FundsCaptureValidationSet	This validation set is implemented in this Java class.
VALIDATION_CODE_ENTRY_POINT	validate	This validation set has this function name.
VALIDATION_CODE_LANGUAGE	JAVA	This validation set is implemented in Java.

For the same validation set data shown in the table above, the payments engine invokes the Java validation set `oracle.apps.iby.bep.proc.paymentech.FundsCaptureValidationSet.validate` dynamically to validate a transaction

Note: You can create the validation set in your own class. You should not add your custom validation sets in the `oracle.apps.iby.bep.proc.paymenttech.FundsCaptureValidationSet` class. This class is meant for Oracle Payments only.

The `IBY_VALIDATION_SETS_VL` table below, which stores seeded validation sets, illustrates a seeded validation set entry point for a PL/SQL validation.

IBY_VALIDATION_SETS_VL

Column Name	Example	Meaning
VALIDATION_LEVEL_CODE	PTK_BATCH_2_1_0	Unique name of the validation set used for querying.
VALIDATION_LEVEL_CODE	INSTRUCTION	This validation is applicable to instructions (payment batches).
VALIDATION_CODE_PACKAGE	IBY_FND_CPT_VLD_PUB	This validation set is implemented in this PL/SQL package.
VALIDATION_CODE_ENTRY_POINT	Validate_Paymenttech_Batch	This validation set has this procedure name.
VALIDATION_CODE_LANGUAGE	PL/SQL	This validation set is implemented in PL/SQL.

For the same validation set data shown in the preceding table, the payments engine invokes the PL/SQL validation set `IBY_FND_CPT_VLD_PUB.Validate_Paymenttech_Batch` dynamically to validate a payment instruction (payment batch).

Linking a Validation Set To a Payment Entity

Funds capture validation sets are linked to only one payment entity as follows:

- Format

The linkages between the validation sets and payment entities are stored in the `IBY_VAL_ASSIGNMENTS` table shown below.

IBY_VAL_ASSIGNMENTS

Column Name	Example	Meaning
VALIDATION_SET_CODE	PTK_ONLINE_7_2_BATCH_2_1_0	Unique name of the validation set used for querying.
VAL_ASSIGNMENT_ENTITY_TYPE	FORMAT	Type of payment entity to which this validation set is linked. This will always be FORMAT for funds capture validation sets.
ASSIGNMENT_ENTITY_ID	PTECH_ONLINE_7_2	Actual entity to which this validation set is linked. In this example, the entity is the format PTECH_ONLINE_7_2.
TERRITORY_CODE	null	If null, this validation set is applicable, regardless of the payment country. If not null, this validation is applicable only to the specified country.

The preceding example from the IBY_VAL_ASSIGNMENTS table shows that the validation set PTK_ONLINE_7_2_BATCH_2_1_0 is linked to the payment format PTECH_ONLINE_7_2. Since the validation set PTK_ONLINE_7_2_BATCH_2_1_0 is applicable at the order level, the payments engine executes this validation whenever a transaction comes in with the format PTECH_ONLINE_7_2.

Validation Set Signature

Payment system format-specific validations are performed within the payments engine by validation set code-points implemented in Java and PL/SQL and then associated with particular payment system formats.

Transaction Level Validation Set

Transaction level Java validation set implementations must conform to the following class interface:

```
oracle.apps.iby.payment.FndCptValidationSet.
```

This interface class has a single method called *validate* with the signature shown in the table below.

Method Parameters

Name	Type	Description
ecappId	int	Electronic commerce application id for the current transaction.
payee	Payee	Transaction payee.
pmtInstr	PmtInstr	Payment instrument used.
p_is_online_val	IN	Y/N flag indicating whether this is an online validation call.
order	Tangible	Order.
txn	Transaction	The transaction performed.
<return>	ValidationSetResult	Results of the validation.

The *validate* method returns the result of the validation via the *ValidationSetResult* object. The *ValidationSetResult* object contains the following elements:

ValidationSetResult Elements

Name	Type	Description
Valid	boolean	Results of the validation. If passed then true, otherwise false.
Message	String	An encoded Oracle Payments error message string of the form: MESSAGE_NAME#TOKEN_NAME1=TOKEN_VAL1#....
Code	String	Error code for the validation, if any.

Batch Level Validation Set

Batch level validations are implemented in PL/SQL and must have the following signature:

Batch Level Validation Set Signature

Name	Data Type	Type	Description
p_api_version	NUMBER	IN	Version of the API called; should be ignored.
p_init_msg_list	VARCHAR2	IN	Whether to initialize the message list.
p_mbatchid	NUMBER	IN	The identifier of the batch (in table IBY_BATCHES_ALL)
x_return_status	VARCHAR2	OUT	Status of the call.
x_msg_count	NUMBER	OUT	Number of error messages on the stack.
x_msg_data	VARCHAR2	OUT	Message stack of errors.

Implementing a Transaction Level Validation Set

A transaction level validation set needs to follow the following process:

1. Your validation set should be coded in a class that implements the *oracle.apps.iby.payment.FndCptValidationSet* interface. That is, your validation set class must contain the *validate* method.
2. The *validate* method is invoked with input parameters that encapsulate the payee, payment instrument, tangible, and transaction attributes. Apply business rule on the relevant attributes.
3. If any attribute has failed validation, populate the *oracle.apps.iby.engine.ValidationSetResult* object with the error code and error message and return.

4. If all validations successfully pass, set the result code to SUCCESS in the *ValidationSetResult* object.
5. The following payment system-specific implementations of the *oracle.apps.iby.payment.FndCptValidationSet* interfaces can be used for reference. Each is listed with its associated payment system format in the table below.

Seeded Payment System Validation Sets

Payment System-Specific Implementations of the <i>payment.FndCptValidationSet</i> Interfaces	Associated Payment System Format
Paymentech Online and Batch Specifications	oracle.apps.iby.bep.proc.paymentech.FundsCaptureValidationSet
First Data North Authorization and Batch Specifications	oracle.apps.bep.proc.fdcnorth.FundsCaptureValidationSet

Implementing a Batch Level Validation Set

A batch level validation set needs to follow the following process:

1. Implement your batch level validation set using the signature mentioned in section 2.1.5.2.
2. The parameter *mbatchid* provided as an input parameter to the validation set, is the primary key to the *IBY_BATCHES_ALL* table. You can use this parameter to pull the details of the batch from the *IBY_BATCHES_ALL* table and also join to the *IBY_TRXN_SUMMARIES_ALL* table to pull the transaction details.
3. Apply business rule on the relevant payment attributes.
4. If any attribute fails validation, set the *x_return_status* output parameter to *FND_API.G_RET_STS_ERROR*. Populate the error message on the FND message stack and return.
5. If all validations have been successfully passed, set *x_return_status* to *FND_API.G_RET_STS_SUCCESS* and return.
6. For a reference implementation of batch level validations, you can look at the package *IBY_FND_CPT_VLD_PUB* (\$IBY_TOP/patch/115/sql/ibypfcvb.pls). The method *Validate_Paymentech_Batch* is used for validating credit card batches sent to Paymentech. Similarly, the method *Validate_FDCNorth_Batch* is used for validating credit card batches sent to the FDC North payment system.

Seed Your Validation Sets

After you create a validation set, remember to seed your validation set in the IBY_VALIDATION_SETS_VL table.

You cannot directly insert your newly created validation set into IBY_VALIDATION_SETS_VL because it is a view. You need to perform the insert using the following steps:

1. Insert your validation set into the IBY_VALIDATION_SETS_B table. This is the base table that does not contain any translatable text.

Example:

```
insert into iby_validation_sets_b (validation_set_code,
validation_level_code,validation_code_package,
validation_code_entry_point,validation_code_language,created_by,
creation_date,last_updated_by,last_update_date,last_update_login,
object_version_number) values
('my_cc_validations','ORDER', 'oracle.apps.iby.bep.proc.myproc.
FundsCaptureValidationSet','validate','JAVA','294110',
sysdate,'294110',sysdate,'294110',1);
```

2. Insert your validation set into the IBY_VALIDATION_SETS_TL table. This table contains translatable text. The validation set name is translatable.

Example:

```
insert into iby_validation_sets_tl (validation_set_code,language,
source_lang,validation_set_display_name,created_by,creation_date,
last_updated_by,last_update_date,last_update_login,
object_version_number) values
('my_cc_validations','us','us','My credit card validatons','294110',
sysdate,'294110',sysdate,'294110',1);
```

Now your validation sets will be visible in the Oracle Payments Setup page. The navigation path is Oracle Payments Setup > Payments Setup > Shared Setup > Validations.

You can link your validation set to appropriate payment entities like funds capture payment methods and formats.

Now you can test your validation logic by creating transactions and submitting batches.

Important Notes for Validation Set Implementers

The following are important notes for validation set implementers:

- Do not include a COMMIT or a ROLLBACK call within your validation set. The validation engine performs a COMMIT after invoking the validation set.
- You may include DML commands to manipulate or query data from other tables from within your custom validation set. However, note that any changes you make will be committed by the validation engine after the validation set has been invoked.

- Do not include any DDL commands that create, alter, or drop tables or views in your validation set. DDL statements issue implicit commits.
- Note that validation sets are invoked once per payment entity. An order level validation set is invoked once per transaction; a batch level validation set is invoked once per payment batch. Therefore, it is in your interest to make your validation set simple and efficient. Otherwise, it may impose a significant performance cost.
- You cannot directly insert your newly created validation set into IBY_VALIDATION_SETS_VL as this is a view. You need to do this insert in two steps: First, into the base table, IBY_VALIDATION_SETS_B, and then into the translatable table, IBY_VALIDATION_SETS_TL.

Payment Formats

Viewing Payment Formats

To view the layout of a payment format, perform the following steps:

1. Navigate to the Oracle Payments Setup page.
2. Click the Go To Task icon associated with XML Publisher Format Templates.
The Templates search page appears.
3. Search for the template in which you are interested.
The Templates page appears.
4. Click the name of the template you wish to view.
The General page appears.
5. Click the Download icon for the template you wish to view.
You can simply open the RTF file in MS Word or save it to your PC.

Common Disbursement Payment Formats

The formats in this section can be used for payments in multiple countries.

ANSI X12 820 Format

The table below presents basic information about the ANSI X12 820 Format.

ANSI X12 820 Format Basic Information

Format Name	Format Type	Template Name	Extract Description
ANSI X12 820 Format	Electronic	ANSI X12 820 Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0.

Additional details about the ANSI X12 820 Format are presented in the table below.

ANSI X12 820 Format Details

Payment Process Profile Name	Currency Name	Validation Name
ANSI X12 820	Any	Not Applicable

EDIFACT PAYMUL Format

The table below presents basic information about the EDIFACT PAYMUL Format.

EDIFACT PAYMUL Format Basic Information

Format Name	Format Type	Template Name	Extract Description
EDIFACT PAYMUL Format	Electronic	EDIFACT PAYMUL Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0.

Additional details about the EDIFACT PAYMUL Format are presented in the table below.

EDIFACT PAYMUL Format Details

Payment Process Profile Name	Currency Name	Validation Name
EDIFACT PAYMUL	Any	Not Applicable

SWIFT MT100 Format

The table below presents basic information about the SWIFT MT100 Format.

SWIFT MT100 Format Basic Information

Format Name	Format Type	Template Name	Extract Description
SWIFT MT100 Format	Electronic	SWIFT MT 100 Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0.

Additional details about the SWIFT MT100 Format are presented in the table below.

SWIFT MT100 Format Details

Payment Process Profile Name	Currency Name	Validation Name
SWIFT MT100	Any	Not Applicable

SWIFT MT103 Format

The table below presents basic information about the SWIFT MT103 Format.

SWIFT MT103 Format Basic Information

Format Name	Format Type	Template Name	Extract Description
SWIFT MT103 Format	Electronic	SWIFT MT 103 Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0.

Additional details about the SWIFT MT103 Format are presented in the table below.

SWIFT MT103 Format Details

Payment Process Profile Name	Currency Name	Validation Name
SWIFT MT103	Any	Not Applicable

Argentina

Use the Argentine Check Format to produce documents. The Argentine Check Format shows the amount of the payment in numbers and in words, the date of the payment, and the name of the supplier being paid. All text on the Argentine Check Format is printed in Spanish.

Argentine Check Format

The table below presents basic information about the Argentine Check Format.

Argentine Check Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Argentine Check Format	Printed	Argentine Check Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Argentine Check Format are presented in the table below.

Argentine Check Format Details

Payment Process Profile Name	Currency Name	Validation Name
Argentine Check	Argentine Peso	Not Applicable

Belgium

Oracle Payments lets you format payments according to standards defined by the Association Belge des Banques/Belgische Vereniging der Banken (Belgium Bank Association).

With Oracle Payments, you can make EFT payments:

- In euro from a Belgian bank to suppliers who have a bank account with a Belgian bank
- In euro or foreign currency from a Belgian bank to suppliers who have a bank account with a foreign bank
- In foreign currency (a currency other than euro) from a Belgian bank to suppliers who have a bank account with a Belgian bank

Belgian EFT Format

The table below presents basic information about the Belgian EFT Format.

Belgian EFT Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Belgian EFT Format	Electronic	Belgian Domestic EFT Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Belgian EFT Format are presented in the table below.

Belgian EFT Format Details

Payment Process Profile Name	Currency Name	Validation Name
Belgian EFT	Euro	Belgium Domestic EFT Document Validation

Belgian Foreign EFT Format

The table below presents basic information about the Belgian Foreign EFT Format.

Belgian Foreign EFT Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Belgian Foreign EFT Format	Electronic	Belgian International EFT Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Belgian Foreign EFT Format are presented in the table below.

Belgian Foreign EFT Format Details

Payment Process Profile Name	Currency Name	Validation Name
Belgian Foreign EFT	Any	Belgium International EFT Payment Instruction Validation
Belgian Foreign EFT	Any	Belgium International EFT Payment Validation
Belgian Foreign EFT	Any	Belgium International EFT Payee Validation
Belgian Foreign EFT	Any	Belgium International EFT Document Validation

Brazil

Payments are usually transferred from the customer's bank account to the supplier's bank account. With bank transfer, an invoice is released for payment only after the supplier's bank sends a collection document for the related invoice to a customer. A customer can either pay on the individual collection document or pay on several collection documents listed on a report called a Boldero. The customer sends this report to its bank for release of payment.

Bank formats in Brazil do not include currency information; all amounts are assumed to be in Brazilian Real. If you make transactions with foreign suppliers, use currencies other than the Brazilian Real.

Brazilian Check Format

The table below presents basic information about the Brazilian Check Format.

Brazilian Check Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Brazilian Check Format	Printed	Brazilian Check Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Brazilian Check Format are presented in the table below.

Brazilian Check Format Details

Payment Process Profile Name	Currency Name	Validation Name
Brazilian Check	Brazilian Real	Not Applicable

Brazilian Boldero Format

The table below presents basic information about the Brazilian Boldero Format.

Brazilian Bordero Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Brazilian Bordero Format	Printed	Brazilian Bordero Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Brazilian Bordero Format are presented in the table below.

Brazilian Bordero Format Details

Payment Process Profile Name	Currency Name	Validation Name
Brazilian Bordero	Brazilian Real	Not Applicable

Chile

This section presents information about Chilean payment formats.

Chilean Check Format

Use the Chilean Check Format to produce checks according to Chilean legal requirements for invoices in a payment batch. The Chilean Check Format shows the amount of the payment in numbers and in words, the date of the payment, and the name of the supplier being paid. All text on the Chilean Check Format is printed in Spanish.

The table below presents basic information about the Chilean Check Format.

Chilean Check Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Chilean Check Format	Printed	Chilean Check Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Chilean Check Format are presented in the table below.

Chilean Check Format Details

Payment Process Profile Name	Currency Name	Validation Name
Chilean Check	Chilean Peso	Not Applicable

Colombia

This section presents information on Colombian payment formats.

Colombian Check 1 Format

Use the Colombian Check Formats to produce checks according to Colombian legal requirements for invoices in a payment batch. In Colombia, the bank superintendent has established two common check formats. You can use the Colombian Check 1 Format to create the smaller check format and the Colombian Check 2 Format to create the larger check format.

In Colombia, when you send a check to a supplier, you must also send the supplier a remittance advice that shows details about the documents that are included in the payment. For example, the remittance advice shows any invoices and credit memos included in the payment, together with details on withholding taxes.

The Colombian Check 1 Format generates checks that are 162 millimeters wide by 71 millimeters long. Each check shows the date of the payment, the amount of the payment in numbers and in words, and the name of the supplier or beneficiary being paid. All text on the Colombian Check 1 Format is printed in Spanish.

The table below presents basic information about the Colombian Check 1 Format.

Colombian Check 1 Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Colombian Check 1 Format	Printed	Colombian Check 1 Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Colombian Check 1 Format are presented in the table below.

Colombian Check 1 Format Details

Payment Process Profile Name	Currency Name	Validation Name
Colombian Check 1	Colombian Peso	Not Applicable

Colombian Check 2 Format

Use the Colombian Check Formats to produce checks according to Colombian legal requirements for invoices in a payment batch. In Colombia, the bank superintendent has established two common check formats. You can use the Colombian Check 1 Format to create the smaller check format and the Colombian Check 2 Format to create the larger check format.

In Colombia, when you send a check to a supplier, you must also send the supplier a remittance advice that shows details about the documents that are included in the payment. For example, the remittance advice shows any invoices and credit memos included in the payment, together with details on withholding taxes.

The Colombian Check 2 Format generates checks that are 185 millimeters wide by 80 millimeters long. Each check shows the date of the payment, the amount of the payment in numbers and in words, and the name of the supplier or beneficiary being paid. All text on the Colombian Check 2 Format is printed in Spanish.

The table below presents basic information about the Colombian Check 2 Format.

Colombian Check 2 Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Colombian Check 2 Format	Printed	Colombian Check 2 Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Colombian Check 2 Format are presented in the table below.

Colombian Check 2 Format Details

Payment Process Profile Name	Currency Name	Validation Name
Colombian Check 2	Colombian Peso	Not Applicable

Finland

This section presents information on Finnish payment formats.

Finnish LMP2 EFT Format

The Finnish LMP2 EFT Format supports electronic funds transfer (EFT) payments for both domestic and foreign invoices.

The table below presents basic information about the Finnish LMP2 EFT Format.

Finnish LMP2 EFT Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Finnish LMP2 EFT Format	Electronic	Finnish LMP2 EFT Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Finnish LMP2 EFT Format are presented in the table below.

Finnish LMP2 EFT Format Details

Payment Process Profile Name	Currency Name	Validation Name
Finnish LMP2 EFT	Any	Not Applicable

Finnish LMP3 EFT Format

The Finnish LMP3 EFT Format supports electronic funds transfer (EFT) payments for both domestic and foreign invoices.

The table below presents basic information about the Finnish LMP3 EFT Format.

Finnish LMP3 EFT Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Finnish LMP3 EFT Format	Electronic	Finnish LMP3 EFT Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Finnish LMP3 EFT Format are presented in the table below.

Finnish LMP3 EFT Format Details

Payment Process Profile Name	Currency Name	Validation Name
Finnish LMP3 EFT	Any	Not Applicable

Finnish LUM EFT Format

The Finnish LUM EFT Format supports electronic funds transfer (EFT) payments for both domestic and foreign invoices.

The table below presents basic information about the Finnish LUM EFT Format.

Finnish LUM EFT Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Finnish LUM EFT Format	Electronic	Finnish LUM EFT Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Finnish LUM EFT Format are presented in the table below.

Finnish LUM EFT Format Details

Payment Process Profile Name	Currency Name	Validation Name
Finnish LUM EFT	Any	Not Applicable

France

This section presents information on French payment formats.

French Check Format

The table below presents basic information about the French Check Format.

French Check Format Basic Information

Format Name	Format Type	Template Name	Extract Description
French Check Format	Printed	French Check Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the French Check Format are presented in the table below.

French Check Format Details

Payment Process Profile Name	Currency Name	Validation Name
French Check	Euro	Not Applicable

French Check Format Stub After Payment

The table below presents basic information about the French Check Format (Stub After Payment).

French Check Format (Stub After Payment) Basic Information

Format Name	Format Type	Template Name	Extract Description
French Check Format (Stub After Payment)	Printed	French Check Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the French Check Format (Stub After Payment) are presented in the table below.

French Check Format (Stub After Payment) Details

Payment Process Profile Name	Currency Name	Validation Name
French Check (Stub After Payment)	Euro	Not Applicable

French EFT Format

Use the French EFT Format to make payments using an EFT payment method (*Electronic* or *Wire*).

The table below presents basic information about the French EFT Format.

French EFT Format Basic Information

Format Name	Format Type	Template Name	Extract Description
French EFT Format	Electronic	French EFT Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the French EFT Format are presented in the table below.

French EFT Format Details

Payment Process Profile Name	Currency Name	Validation Name
French EFT	Euro	Not Applicable

French Promissory Note Format

Use the French Promissory Note Format (Billet a Ordre) program to make future dated payments for invoices that have the same due date.

The table below presents basic information about the French Promissory Note Format.

French Promissory Note Format Basic Information

Format Name	Format Type	Template Name	Extract Description
French Promissory Note Format	Printed	French Promissory Note Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the French Promissory Note Format are presented in the table below.

French Promissory Note Format Details

Payment Process Profile Name	Currency Name	Validation Name
French Promissory Note	Euro	Not Applicable

French Bills Receivable Remittance Format

Use the French Bills Receivables Remittance Format (Etat de Remise a la Banque) to transmit one or more bills of exchange bills receivable for bank remittance. The French Bills Receivable Remittance Format creates a magnetic file in a format that conforms to the ETEBAC standards issued by the CFONB (Comite Francais d'Organisation et de Normalisation Bancaire) of the AFB (Association Francaise de Banques).

The table below presents basic information about the French Bills Receivable Remittance Format.

French Bills Receivable Remittance Format Basic Information

Format Name	Format Type	Template Name	Extract Description
French Bills Receivable Remittance Format	Electronic	French Bills Receivable Remittance Format	Data extract for credit card, purchase card, debit card, and bank account funds capture transactions.

Additional details about the French Bills Receivable Remittance Format are presented in the table below.

French Bills Receivable Remittance Format Details

Payment Process Profile Name	Currency Name	Validation Name
French Bills Receivable Remittance	Euro	Not Applicable

Germany

This section presents information on German payment formats.

German Domestic EFT Format

Use the German Domestic EFT Format to make domestic (German Inland) payments from a German bank.

The table below presents basic information about the German Domestic EFT Format.

German Domestic EFT Format Basic Information

Format Name	Format Type	Template Name	Extract Description
German Domestic EFT Format	Electronic	German Domestic EFT Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the German Domestic EFT Format are presented in the table below.

German Domestic EFT Format Details

Payment Process Profile Name	Currency Name	Validation Name
German Domestic EFT	Euro	Germany Domestic EFT Payment Validation
German Domestic EFT	Euro	Germany Domestic EFT Payee Validation
German Domestic EFT	Euro	Germany Domestic EFT Payer Validation
German Domestic EFT	Euro	Germany Domestic EFT Internal Bank Validation

German International EFT Format

Use the German International EFT Format to make German international payments from a German bank.

The table below presents basic information about the German International EFT Format.

German International EFT Format Basic Information

Format Name	Format Type	Template Name	Extract Description
German International EFT Format	Electronic	German International EFT Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the German International EFT Format are presented in the table below.

German International EFT Format Details

Payment Process Profile Name	Currency Name	Validation Name
German International EFT	Euro	Germany International EFT Payment Validation
German International EFT	Euro	Germany International EFT Payee Validation
German International EFT	Euro	Germany International EFT Payer Validation
German International EFT	Euro	Germany International EFT Internal Bank Validation

German Check Format

Use the German Check Format to produce checks. The amount is printed in German. The table below presents basic information about the German Check Format.

German Check Format Basic Information

Format Name	Format Type	Template Name	Extract Description
German Check Format	Printed	German Check Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the German Check Format are presented in the table below.

German Check Format Details

Payment Process Profile Name	Currency Name	Validation Name
German Check	Euro	Not Applicable

German Check Format (Stub After Payment)

The table below presents basic information about the German Check Format (Stub After Payment).

German Check Format (Stub After Payment) Basic Information

Format Name	Format Type	Template Name	Extract Description
German Check Format (Stub After Payment)	Printed	German Check Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the German Check Format (Stub After Payment) are presented in the table below.

German Check Format (Stub After Payment) Details

Payment Process Profile Name	Currency Name	Validation Name
German Check (Stub After Payment)	Euro	Not Applicable

German Wire Format

Use the German Wire Format for wire payments.

The table below presents basic information about the German Wire Format.

German Wire Format Basic Information

Format Name	Format Type	Template Name	Extract Description
German Wire Format	Printed	German Wire Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the German Wire Format are presented in the table below.

German Wire Format Details

Payment Process Profile Name	Currency Name	Validation Name
German Wire	Euro	Not Applicable

German Direct Debit EFT Format

Use the German Direct Debit EFT Format to create electronic remittances.

The table below presents basic information about the German Direct Debit EFT Format.

German Direct Debit EFT Format Basic Information

Format Name	Format Type	Template Name	Extract Description
German Direct Debit EFT Format	Electronic	German Direct Debit Format	Data extract for credit card, purchase card, debit card, and bank account funds capture transactions.

Additional details about the German Direct Debit EFT Format are presented in the table below.

German Direct Debit EFT Format Details

Payment Process Profile Name	Currency Name	Validation Name
German Direct Debit EFT	Euro?	Not Applicable

German Direct Debit Accompanying Letter Format

The German Direct Debit Accompanying Letter Format produces a letter to accompany the payment instruction to the bank.

Submitting German Direct Debit EFT Format Processing

Prerequisites

- Set up all the remittance banks and bank accounts you use to remit your direct debit payments. (See: *Define Remittance Banks, Defining Banks and Defining Bank Accounts, Oracle Receivables User Guide.*)
- Set up direct debit Receipt Class with an appropriate Receipt Methods. (See: *Setting Up Receipt Classes and Receipt Methods, Oracle Receivables User Guide.*)
- Set up Payment Details for your customer site:
 - In the Receipt Methods region, assign the Receipt Methods setup for the direct debit.
 - Define Bank Account Transfer information. Assign Customer Bank Accounts that you will remit your receipts to. (See: *Customers, Oracle Receivables User*

Guide.)

- When you assign Customer Bank Accounts, set up the Bank Account Transfer Attributes. Select the direct debit authorization code from the list of values in the Direct Debit Bank Instruction field: '04 - Direct Debiting Procedure', '05 - Bank Collection Procedure'.

Italy

This section presents information on Italian payment formats.

Italian EFT Format

The table below presents basic information about the Italian EFT Format.

Italian EFT Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Italian EFT Format	Electronic	Italian EFT Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Italian EFT Format are presented in the table below.

Italian EFT Format Details

Payment Process Profile Name	Currency Name	Validation Name
Italian EFT	Euro	Not Applicable

Italian Wire Format

Use the Italian Wire Format to pay your suppliers with the Italian wire process.

The table below presents basic information about the Italian Wire Format.

Italian Wire Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Italian Wire Format	Printed	Italian Wire Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Italian Wire Format are presented in the table below.

Italian Wire Format Details

Payment Process Profile Name	Currency Name	Validation Name
Italian Wire	Euro	Not Applicable

Italian Bills Receivable Remittance Format

Use the Italian Bills Receivable Remittance Format to transmit bills receivable for bank remittance.

The table below presents basic information about the Italian Bills Receivable Remittance Format.

Italian Bills Receivable Remittance Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Italian Bills Receivable Remittance Format	Electronic	Italian Bills Receivable Remittance Format	Data extract for credit card, purchase card, debit card, and bank account funds capture transactions.

Additional details about the Italian Bills Receivable Remittance Format are presented in the table below.

Italian Bills Receivable Remittance Format Details

Payment Process Profile Name	Currency Name	Validation Name
Italian Bills Receivable Remittance	Euro?	Not Applicable

Japan

This section presents information on Japanese payment formats.

Japanese Zengin Format

It is common in Japan for customers to pay suppliers by transferring funds from the customer's bank to the supplier's bank. The customer's bank charges a fee to complete the transfer, and the customer and supplier negotiate who will bear the fee.

The table below presents basic information about the Japanese Zengin Format.

Japanese Zengin Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Japanese Zengin Format	Electronic	Japanese Zengin Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Japanese Zengin Format are presented in the table below.

Japanese Zengin Format Details

Payment Process Profile Name	Currency Name	Validation Name
Japanese Zengin	Any	Japan Zengin EFT Payee Validation
Japanese Zengin	Any	Japan Zengin EFT Internal Bank Validation

Netherlands

This section presents information on Nordic payment formats.

Netherlands Domestic EFT Format

In the Netherlands, companies can choose to make payments by electronic funds transfer (EFT) rather than by check. With EFT, you send an electronic file to your bank that instructs the bank to transfer funds from your account to your supplier's account. You can make either foreign or domestic EFT payments.

Domestic EFT payments are made in euros (EUR) to a company that is registered with the Dutch Chamber of Commerce.

The table below presents basic information about the Netherlands Domestic EFT Format.

Netherlands Domestic EFT Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Netherlands Domestic EFT Format	Electronic	Netherlands Domestic EFT Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Netherlands Domestic EFT Format are presented in the table below.

Netherlands Domestic EFT Format Details

Payment Process Profile Name	Currency Name	Validation Name
Netherlands Domestic EFT	Euro	Not Applicable

Netherlands Foreign EFT Format

In the Netherlands, companies can choose to make payments by electronic funds transfer (EFT) rather than by check. With EFT, you send an electronic file to your bank that instructs the bank to transfer funds from your account to your supplier's account. You can make either foreign or domestic EFT payments.

Foreign EFT payments are made in a foreign currency or payment to a company that is not registered with the Dutch Chamber of Commerce. If the foreign payment amount exceeds the reporting threshold set by the Dutch National Bank (DNB), you must report the payment to the DNB.

The table below presents basic information about the Netherlands Foreign EFT Format.

Netherlands Foreign EFT Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Netherlands Foreign EFT Format	Electronic	Netherlands International EFT Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Netherlands Foreign EFT Format are presented in the table below.

Netherlands Foreign EFT Format Details

Payment Process Profile Name	Currency Name	Validation Name
Netherlands Foreign EFT	Euro	Not Applicable

New Zealand

This section presents information on New Zealand payment formats.

Bank of New Zealand EFT Format

The table below presents basic information about the Bank of New Zealand EFT Format.

Bank of New Zealand EFT Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Bank of New Zealand EFT Format	Electronic	Bank of New Zealand EFT Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Bank of New Zealand EFT Format are presented in the table below.

Bank of New Zealand EFT Format Details

Payment Process Profile Name	Currency Name	Validation Name
Bank of New Zealand EFT	Any	Not Applicable

Norway

This section presents information on Norwegian payment formats.

Norwegian BBS Format

The Norwegian BBS Format relies on the BBS organization to receive transactions by file transfer, tape, or diskette. An example of BBS payment transfer is a payment to a bank account or a Postgiro account. The BBS organization can also return acknowledgments to the sender by electronic media to confirm that the transactions were received.

The table below presents basic information about the Norwegian BBS Format.

Norwegian BBS Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Norwegian BBS Format	Electronic	Norwegian BBS Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Norwegian BBS Format are presented in the table below.

Norwegian BBS Format Details

Payment Process Profile Name	Currency Name	Validation Name
Norwegian BBS	Norwegian Krone	Not Applicable

Norwegian Telepay EFT Format

The Norwegian Telepay EFT Format is available as a common service from most commercial banks. Savings banks also offer a comparable payment service. You can make both domestic and foreign payment transfers with the Telepay format.

Your bank can return data with information about the transfer.

The table below presents basic information about the Norwegian Telepay EFT Format.

Norwegian Telepay EFT Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Norwegian Telepay EFT Format	Electronic	Norwegian Telepay Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Norwegian Telepay EFT Format are presented in the table below.

Norwegian Telepay EFT Format Details

Payment Process Profile Name	Currency Name	Validation Name
Norwegian Telepay EFT	Any	Not Applicable

Poland

This section presents information on Polish payment formats.

Polish Pekao Credit Transfers Format

Use the Polish Pekao Credit Transfers Format for domestic credit transfers, special orders, and payments to the Social Insurance Institution (ZUS) or related special orders.

The table below presents basic information about the Polish Pekao Credit Transfers Format.

Polish Pekao Credit Transfers Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Polish Pekao Credit Transfers Format	Electronic	Polish Pekao Credit Transfers Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Polish Pekao Credit Transfers Format are presented in the table below.

Polish Pekao Credit Transfers Format Details

Payment Process Profile Name	Currency Name	Validation Name
Polish Pekao Credit Transfers	Polish Zloty	Not Applicable

Polish Pekao Payment Order Format

The table below presents basic information about the Polish Pekao Payment Order Format.

Polish Pekao Payment Order Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Polish Pekao Payment Order Format	Electronic	Polish Pekao Standard Payment Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Polish Pekao Payment Order Format are presented in the

table below.

Polish Pekao Payment Order Format Details

Payment Process Profile Name	Currency Name	Validation Name
Polish Pekao Payment Order	Polish Zloty	Not Applicable

Portugal

This section presents information on Portuguese payment formats.

Portuguese EFT Format

Use Electronic Funds Transfer (EFT) to pay suppliers with a payment file that you send to the company's bank.

The table below presents basic information about the Portuguese EFT Format.

Portuguese EFT Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Portuguese EFT Format	Electronic	Portuguese EFT Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Portuguese EFT Format are presented in the table below.

Portuguese EFT Format Details

Payment Process Profile Name	Currency Name	Validation Name
Portuguese EFT	Portuguese Escudo	Not Applicable

Portuguese Check Format

Oracle Payments provides checks in standard Portuguese Check Format as well as an

accompanying letter that details the invoices that make up the payment batch.

The check/remittance is printed as one document: the remittance is at the top of the document, and the check is at the bottom.

When you define the payment format, enter ten or fewer invoices per check. If the number of invoices paid per check exceeds ten, all checks are voided except the last check, which includes the total amount for all invoices.

The table below presents basic information about the Portuguese Check Format.

Portuguese Check Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Portuguese Check Format	Printed	Portuguese Check Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Portuguese Check Format are presented in the table below.

Portuguese Check Format Details

Payment Process Profile Name	Currency Name	Validation Name
Portuguese Check	Euro	Portugal Generic Check Validation

Portuguese Wire Format

The table below presents basic information about the Portuguese Wire Format.

Portuguese Wire Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Portuguese Wire Format	Printed	Portuguese Wire Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Portuguese Wire Format are presented in the table below.

Portuguese Wire Format Details

Payment Process Profile Name	Currency Name	Validation Name
Portuguese Wire	Any	Not Applicable

Portuguese Direct Debit File Format

The table below presents basic information about the Portuguese Direct Debit File Format.

Portuguese Direct Debit File Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Portuguese Direct Debit File Format	Electronic	Portuguese Direct Debit Format	Data extract for credit card, purchase card, debit card, and bank account funds capture transactions.

Additional details about the Portuguese Direct Debit File Format are presented in the table below.

Portuguese Direct Debit File Format Details

Payment Process Profile Name	Currency Name	Validation Name
Portuguese Direct Debit File	Any	Not Applicable

Spain

This section presents information on Spanish payment formats.

Spanish Magnetic Format

Use the Spanish Magnetic Format to obtain batch payments in electronic format.

The table below presents basic information about the Spanish Magnetic Format.

Spanish Magnetic Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Spanish Magnetic Format	Electronic	Spanish EFT Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Spanish Magnetic Format are presented in the table below.

Spanish Magnetic Format Details

Payment Process Profile Name	Currency Name	Validation Name
Spanish Magnetic	Euro	Not Applicable

Spanish Bill of Exchange Format

Use the Spanish Bill of Exchange Format to see promissory notes for a batch payment.

The table below presents basic information about the Spanish Bill of Exchange Format.

Spanish Bill of Exchange Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Spanish Bill of Exchange Format	Printed	Spanish Payables Bills of Exchange Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Spanish Bill of Exchange Format are presented in the table below.

Spanish Bill of Exchange Format Details

Payment Process Profile Name	Currency Name	Validation Name
Spanish Bill of Exchange	Spanish Peseta	Not Applicable

Spanish Check Format

The table below presents basic information about the Spanish Check Format.

Spanish Check Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Spanish Check Format	Printed	Spanish Check Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Spanish Check Format are presented in the table below.

Spanish Check Format Details

Payment Process Profile Name	Currency Name	Validation Name
Spanish Check	Euro	Not Applicable

Spanish CSB32 Remittance Format

Use the Spanish CSB32 Remittance Format to transmit bills receivable to your remittance bank. The Spanish CSB32 Remittance Format creates a magnetic file in either of two formats: CSB32 or CSB58. Both of these formats conform to Spanish legal standards.

The Spanish CSB32 Remittance Format lets you send more than one remittance file to the bank on the same day. The first four digits of the remittance deposit number identifies each remittance file.

The table below presents basic information about the Spanish CSB32 Remittance

Format.

Spanish CSB32 Remittance Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Spanish CSB32 Remittance Format	Electronic	Spanish CSB32 Remittance Format	Data extract for credit card, purchase card, debit card, and bank account funds capture transactions.

Additional details about the Spanish CSB32 Remittance Format are presented in the table below.

Spanish CSB32 Remittance Format Details

Payment Process Profile Name	Currency Name	Validation Name
Spanish CSB32 Remittance	Any	Not Applicable

Spanish CSB58 Remittance Format

Use the Spanish CSB58 Remittance Format to transmit bills receivable to your remittance bank. The Spanish CSB58 Remittance Format creates a magnetic file in either of two formats: CSB32 or CSB58. Both of these formats conform to Spanish legal standards.

The Spanish CSB58 Remittance Format uses a concatenation of the taxpayer ID and remittance bank EFT number to identify the presenter. There are also optional records that account for the transactions assigned to each bill included in the remittance.

The table below presents basic information about the Spanish CSB58 Remittance Format.

Spanish CSB58 Remittance Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Spanish CSB58 Remittance Format	Electronic	Spanish CSB58 Remittance Format	Data extract for credit card, purchase card, debit card, and bank account funds capture transactions.

Additional details about the Spanish CSB58 Remittance Format are presented in the table below.

Spanish CSB58 Remittance Format Details

Payment Process Profile Name	Currency Name	Validation Name
Spanish CSB58 Remittance Format	Any	Not Applicable

Spanish CSB19 Direct Debit Magnetic Format

Use the Spanish CSB19 Direct Debit Magnetic Format to create a magnetic file of bills of exchange to submit to your bank. This file complies with the format set by the Spanish banking standards authority (CSB19). The Spanish CSB19 Direct Debit Magnetic Format supports both the procedure 1 (standard) and procedure 2 (simplified) formats offered by the banking standards authority.

The table below presents basic information about the Spanish CSB19 Direct Debit Magnetic Format.

Spanish CSB19 Direct Debit Magnetic Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Spanish CSB19 Direct Debit Magnetic Format	Electronic	Spanish Direct Debit Magnetic Format	Data extract for credit card, purchase card, debit card, and bank account funds capture transactions.

Additional details about the Spanish CSB19 Direct Debit Magnetic Format are presented in the table below.

Spanish CSB19 Direct Debit Magnetic Format Details

Payment Process Profile Name	Currency Name	Validation Name
Spanish CSB19 Direct Debit Magnetic	Any	Not Applicable

Sweden

This section presents information on Swedish payment formats.

Swedish Domestic Bankgiro Format

Use Swedish Domestic Bankgiro Format for domestic payments in Swedish kronas or euros. Swedish Domestic Bankgiro Format handles giro payments through the Swedish Bankgiro central clearing house, as well as direct bank deposits to the supplier's account and remittance vouchers to the supplier's address. Swedish Domestic Bankgiro Format supports immediate payments, payments on a specified date, and payments on the invoice due or discount date.

The table below presents basic information about the Swedish Domestic Bankgiro Format.

Swedish Domestic Bankgiro Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Swedish Domestic Bankgiro Format	Electronic	Swedish Bankgiro Inland Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Swedish Domestic Bankgiro Format are presented in the table below.

Swedish Domestic Bankgiro Format Details

Payment Process Profile Name	Currency Name	Validation Name
Swedish Domestic Bankgiro	Euro	Sweden EFT Bankgiro Inland Document Validation

Swedish Foreign Postgiro Format

Use Swedish Foreign Postgiro Format for foreign payments in any valid foreign currency. These payments go through the Swedish Postal Giro clearing house. Swedish Foreign Postgiro Format supports immediate payments, payments on a specified date, and payments on the invoice due or discount date.

The table below presents basic information about the Swedish Foreign Postgiro Format.

Swedish Foreign Postgiro Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Swedish Foreign Postgiro Format	Electronic	Swedish Postgiro Utland Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Swedish Foreign Postgiro Format are presented in the table below.

Swedish Foreign Postgiro Format Details

Payment Process Profile Name	Currency Name	Validation Name
Swedish Foreign Postgiro	Euro	Sweden EFT Postgiro Utland Document Validation

Swedish Domestic Postgiro Format

Use Swedish Domestic Postgiro Format for domestic payments in Swedish kronas or euros. Swedish Domestic Postgiro Format handles giro payments through the Swedish

Postal Giro clearing house, as well as direct bank deposits to the supplier's account. Postgiro Inland supports immediate payments, payments on a specified date, and payments on the invoice due or discount date.

The table below presents basic information about the Swedish Domestic Postgiro Format.

Swedish Domestic Postgiro Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Swedish Domestic Postgiro Format	Electronic	Swedish Postgiro Inland Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Swedish Domestic Postgiro Format are presented in the table below.

Swedish Domestic Postgiro Format Details

Payment Process Profile Name	Currency Name	Validation Name
Swedish Domestic Postgiro	Euro	Sweden EFT Postgiro Inland Document Validation

Swedish SISU Bankgiro Format

Use Swedish SISU Bankgiro Format for foreign payments in any valid foreign currency, including the euro. These payments go through the Swedish Bankgiro central clearing house and are processed by SE Banken. The Swedish SISU Bankgiro Format supports immediate payments, payments on a specified date, and payments on the invoice due or discount date.

The table below presents basic information about the Swedish SISU Bankgiro Format.

Swedish SISU Bankgiro Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Swedish SISU Bankgiro Format	Electronic	Swedish Bankgiro SISU Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Swedish SISU Bankgiro Format are presented in the table below.

Swedish SISU Bankgiro Format Details

Payment Process Profile Name	Currency Name	Validation Name
Swedish SISU Bankgiro	Any	Sweden EFT Bankgiro Utland SISU Document Validation

Swedish UTLI Bankgiro Format

Use Swedish UTLI Bankgiro Format for foreign payments in any valid foreign currency, including the euro. These payments go through the Swedish Bankgiro central clearing house and are processed by Svenska Handelsbanken. Bankgiro UTLI supports immediate payments, payments on a specified date, and payments on the invoice due or discount date.

The table below presents basic information about the Swedish UTLI Bankgiro Format.

Swedish UTLI Bankgiro Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Swedish UTLI Bankgiro Format	Electronic	Swedish Bankgiro UTLI Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Swedish UTLI Bankgiro Format are presented in the table below.

Swedish UTLI Bankgiro Format Details

Payment Process Profile Name	Currency Name	Validation Name
Swedish UTLI Bankgiro	Any	Sweden EFT Bankgiro Utland UTLI Document Validation

Switzerland

This section presents information on Swiss payment formats.

Swiss DTA Format

Oracle Payments provides two different electronic funds transfer (EFT) processes: DTA and SAD. Use the Swiss DTA Format for domestic or foreign payments that originate from accounts at a Swiss bank. Your payment amounts are automatically deducted from your accounts and credited to your supplier's accounts, and balances are updated in Oracle Payments.

Each data carrier must be accompanied by a completely filled in delivery note. Delivery notes can be requested from the principal bank and must be used as follows:

- Page 1 (white) and page 2 (green) to be sent together with the data carrier to the Telekurs CC.
- Page 3 (yellow) is for the sender of the data carrier.

The table below presents basic information about the Swiss DTA Format.

Swiss DTA Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Swiss DTA Format	Electronic	Swiss DTA Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Swiss DTA Format are presented in the table below.

Swiss DTA Format Details

Payment Process Profile Name	Currency Name	Validation Name
Swiss DTA	Swiss Franc	Switzerland Generic EFT Payee Validation
Swiss DTA	Swiss Franc	Switzerland Generic EFT Document Validation

Swiss SAD Format

Oracle Payments provides two different electronic funds transfer (EFT) processes: DTA and SAD. Use the Swiss SAD Format for domestic payments that originate from accounts at die Post, the Swiss mailing company. Your payment amounts are automatically deducted from your accounts and credited to your supplier's accounts, and balances are updated in Oracle Payments.

The table below presents basic information about the Swiss SAD Format.

Swiss SAD Format Basic Information

Format Name	Format Type	Template Name	Extract Description
Swiss SAD Format	Printed	Swiss SAD Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the Swiss SAD Format are presented in the table below.

Swiss SAD Format Details

Payment Process Profile Name	Currency Name	Validation Name
Swiss SAD Format	Swiss Franc	Switzerland Generic EFT Document Validation

Payment Process Profile Name	Currency Name	Validation Name
Swiss SAD Format	Swiss Franc	Switzerland Generic EFT Payee Validation

UK

This section presents information on English payment formats.

UK BACS 1/2 Inch Tape Format

The table below presents basic information about the UK BACS 1/2 Inch Tape Format.

UK BACS 1/2 Inch Tape Format Basic Information

Format Name	Format Type	Template Name	Extract Description
UK BACS 1/2 Inch Tape Format	Electronic	UK BACS 1/2 Inch Tape Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the UK BACS 1/2 Inch Tape Format are presented in the table below.

UK BACS 1/2 Inch Tape Format Details

Payment Process Profile Name	Currency Name	Validation Name
UK BACS 1/2 Inch Tape	Pound Sterling	Not Applicable

US

This section presents information on American payment formats.

US NACHA CCD Format

The table below presents basic information about the US NACHA CCD Format.

US NACHA CCD Format Basic Information

Format Name	Format Type	Template Name	Extract Description
US NACHA CCD Format	Electronic	US NACHA CCD Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the US NACHA CCD Format are presented in the table below.

US NACHA CCD Format Details

Payment Process Profile Name	Currency Name	Validation Name
US NACHA CCD	US Dollar	Not Applicable

US NACHA CCDP Format

The table below presents basic information about the US NACHA CCDP Format.

US NACHA CCDP Format Basic Information

Format Name	Format Type	Template Name	Extract Description
US NACHA CCDP Format	Electronic	US NACHA CCDP Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the US NACHA CCDP Format are presented in the table below.

US NACHA CCDP Format Details

Payment Process Profile Name	Currency Name	Validation Name
US NACHA CCDP	US Dollar	Not Applicable

US NACHA PPD Format

The table below presents basic information about the US NACHA PPD Format.

US NACHA PPD Format Basic Information

Format Name	Format Type	Template Name	Extract Description
US NACHA PPD Format	Electronic	US NACHA PPD Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the US NACHA PPD Format are presented in the table below.

US NACHA PPD Format Details

Payment Process Profile Name	Currency Name	Validation Name
US NACHA PPD	US Dollar	Not Applicable

US NACHA Generic Format

The table below presents basic information about the US NACHA Generic Format.

US NACHA Generic Format Basic Information

Format Name	Format Type	Template Name	Extract Description
US NACHA Generic Format	Electronic	US NACHA Generic Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the US NACHA Generic Format are presented in the table below.

US NACHA Generic Format Details

Payment Process Profile Name	Currency Name	Validation Name
US NACHA Generic	US Dollar	US NACHA Internal Bank Validation
US NACHA Generic	US Dollar	US NACHA Payee Validation
US NACHA Generic	US Dollar	US NACHA Payment Instruction Validation

US Treasury Format

The table below presents basic information about the US Treasury Format.

US Treasury Format Basic Information

Format Name	Format Type	Template Name	Extract Description
US Treasury Format	Printed	US Treasury Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the US Treasury Format are presented in the table below.

US Treasury Format Details

Payment Process Profile Name	Currency Name	Validation Name
US Treasury	US Dollar	Not Applicable

US Federal

This section presents information on United States Federal payment formats.

US Federal ECS CCD Format

The table below presents basic information about the US Federal ECS CCD Format.

US Federal ECS CCD Format Basic Information

Format Name	Format Type	Template Name	Extract Description
US Federal ECS CCD Format	Electronic	Federal ECS CCD Payments Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the US Federal ECS CCD Format are presented in the table below.

US Federal ECS CCD Format Details

Payment Process Profile Name	Currency Name	Validation Name
US Federal ECS CCD	US Dollar	Federal Financials ECS CCD Payment Validation

US Federal ECS CCDP Format

The table below presents basic information about the US Federal ECS CCDP Format.

US Federal ECS CCDP Format Basic Information

Format Name	Format Type	Template Name	Extract Description
US Federal ECS CCDP Format	Electronic	Federal ECS CCDP Payments Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the US Federal ECS CCDP Format are presented in the table below.

US Federal ECS CCDP Format Details

Payment Process Profile Name	Currency Name	Validation Name
US Federal ECS CCDP	US Dollar	Federal Financials ECS CCD+ Payment Validation

US Federal ECS PPD Format

The table below presents basic information about the US Federal ECS PPD Format.

US Federal ECS PPD Format Basic Information

Format Name	Format Type	Template Name	Extract Description
US Federal ECS PPD Format	Electronic	Federal ECS PPD Payments Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the US Federal ECS PPD Format are presented in the table below.

US Federal ECS PPD Format Details

Payment Process Profile Name	Currency Name	Validation Name
US Federal ECS PPD	US Dollar	Federal Financials ECS PPD Payment Validation

US Federal ECS PPDP Format

The table below presents basic information about the US Federal ECS PPDP Format.

US Federal ECS PPDP Format Basic Information

Format Name	Format Type	Template Name	Extract Description
US Federal ECS PPDPFormat	Electronic	Federal ECS PPDP Payments Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the US Federal ECS PPDP Format are presented in the table below.

US Federal ECS PPDP Format Details

Payment Process Profile Name	Currency Name	Validation Name
US Federal ECS PPDP	US Dollar	Federal Financials ECS PPDP+ Payment Validation

US ECS NCR Format

The table below presents basic information about the US ECS NCR Format.

US ECS NCR Format Basic Information

Format Name	Format Type	Template Name	Extract Description
US ECS NCR Format	Electronic	US ECS NCR Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the US ECS NCR Format are presented in the table below.

US ECS NCR Format Details

Payment Process Profile Name	Currency Name	Validation Name
US ECS NCR	US Dollar	Federal Financials ECS NCR Payment Validation

US Federal CTX Format

The table below presents basic information about the US Federal CTX Format.

US Federal CTX Format Basic Information

Format Name	Format Type	Template Name	Extract Description
US Federal CTX Format	Electronic	Federal CTX ACH Vendor Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the US Federal CTX Format are presented in the table below.

US Federal CTX Format Details

Payment Process Profile Name	Currency Name	Validation Name
US Federal CTX	US Dollar	Federal Financials CTX Payment Validation

US Federal Bulk CCDP Format

The table below presents basic information about the US Federal Bulk CCDP Format.

US Federal Bulk CCDP Format Basic Information

Format Name	Format Type	Template Name	Extract Description
US Federal Bulk CCDP Format	Electronic	Federal Bulk Data CCDP Payments Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the US Federal Bulk CCDP Format are presented in the table below.

US Federal Bulk CCDP Format Details

Payment Process Profile Name	Currency Name	Validation Name
US Federal Bulk CCDP	US Dollar	Federal Financials Bulk Data CCD+ Payment Validation

US Federal Bulk PPDP Format

The table below presents basic information about the US Federal Bulk PPDP Format.

US Federal Bulk PPDP Format Basic Information

Format Name	Format Type	Template Name	Extract Description
US Federal Bulk PPDP Format	Electronic	Federal Bulk Data PPDP Payments Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the US Federal Bulk PPDP Format are presented in the table below.

US Federal Bulk PPDP Format Details

Payment Process Profile Name	Currency Name	Validation Name
US Federal Bulk PPDP	US Dollar	Federal Financials Bulk Data PPD+ Payment Validation

US Bulk Salary and Travel NCR Format

The table below presents basic information about the US Bulk Salary and Travel NCR Format.

US Bulk Salary and Travel NCR Format Basic Information

Format Name	Format Type	Template Name	Extract Description
US Bulk Salary and Travel NCR Format	Electronic	US Bulk Salary and Travel NCR Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the US Bulk Salary and Travel NCR Format are presented in the table below.

US Bulk Salary and Travel NCR Format Details

Payment Process Profile Name	Currency Name	Validation Name
US Bulk Salary and Travel NCR	US Dollar	Federal Financials Bulk Salary/Travel NCR Payment Validation

US Bulk NCR Format

The table below presents basic information about the US Bulk NCR Format.

US Bulk NCR Format Basic Information

Format Name	Format Type	Template Name	Extract Description
US Bulk NCR Format	Electronic	US Bulk NCR Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the US Bulk NCR Format are presented in the table below.

US Bulk NCR Format Details

Payment Process Profile Name	Currency Name	Validation Name
US Bulk NCR	US Dollar	Federal Financials Bulk Data NCR Payment Validation

US Federal ECS Summary Schedule Format

The table below presents basic information about the US Federal ECS Summary Schedule Format.

US Federal ECS Summary Schedule Format Basic Information

Format Name	Format Type	Template Name	Extract Description
US Federal ECS Summary Schedule Format	Electronic	Federal ECS Summary Schedule	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

US Federal SPS CCD Format

The table below presents basic information about the US Federal SPS CCD Format.

US Federal SPS CCD Format Basic Information

Format Name	Format Type	Template Name	Extract Description
US Federal SPS CCD Format	Electronic	Federal SPS CCD Payments Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the US Federal SPS CCD Format are presented in the table below.

US Federal SPS CCD Format Details

Payment Process Profile Name	Currency Name	Validation Name
US Federal SPS CCD	US Dollar	Federal Financials SPS CCD Payment Validation

US Federal SPS CCDP Format

The table below presents basic information about the US Federal SPS CCDP Format.

US Federal SPS CCDP Format Basic Information

Format Name	Format Type	Template Name	Extract Description
US Federal SPS CCDP Format	Electronic	Federal SPS CCDP Payments Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the US Federal SPS CCDP Format are presented in the table below.

US Federal SPS CCDP Format Details

Payment Process Profile Name	Currency Name	Validation Name
US Federal SPS CCDP	US Dollar	Federal Financials SPS CCD+ Payment Validation

US Federal SPS PPD Format

The table below presents basic information about the US Federal SPS PPD Format.

US Federal SPS PPD Format Basic Information

Format Name	Format Type	Template Name	Extract Description
US Federal SPS PPD Format	Electronic	Federal SPS PPD Payments Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the US Federal SPS PPD Format are presented in the table below.

US Federal SPS PPD Format Details

Payment Process Profile Name	Currency Name	Validation Name
US Federal SPS PPD	US Dollar	Federal Financials SPS PPD Payment Validation

US Federal SPS PPDP Format

The table below presents basic information about the US Federal SPS PPDP Format.

US Federal SPS PPDP Format Basic Information

Format Name	Format Type	Template Name	Extract Description
US Federal SPS PPDP Format	Electronic	Federal SPS PPDP Payments Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the US Federal SPS PPDP Format are presented in the table below.

US Federal SPS PPDP Format Details

Payment Process Profile Name	Currency Name	Validation Name
US Federal SPS PPDP	US Dollar	Federal Financials SPS PPD+ Payment Validation

US SPS NCR Format

The table below presents basic information about the US SPS NCR Format.

US SPS NCR Format Basic Information

Format Name	Format Type	Template Name	Extract Description
US SPS NCR Format	Electronic	US SPS NCR Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the US SPS NCR Format are presented in the table below.

US SPS NCR Format Details

Payment Process Profile Name	Currency Name	Validation Name
US SPS NCR	US Dollar	Federal Financials SPS NCR Payment Validation

US Federal SPS Summary Schedule Format

The table below presents basic information about the US Federal SPS Summary Schedule Format.

US Federal SPS Summary Schedule Format Basic Information

Format Name	Format Type	Template Name	Extract Description
US Federal SPS Summary Schedule Format	Electronic	Federal SPS Summary Schedule	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

US Federal CCDP Consolidated Format

The table below presents basic information about the US Federal CCDP Consolidated Format.

US Federal CCDP Consolidated Format Basic Information

Format Name	Format Type	Template Name	Extract Description
US Federal CCDP Consolidated Format	Electronic	US CCDP Consolidated Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the US Federal CCDP Consolidated Format are presented in the table below.

US Federal CCDP Consolidated Format Details

Payment Process Profile Name	Currency Name	Validation Name
US Federal CCDP Consolidated	US Dollar	Federal Financials Bulk Data CCD+ Payment Validation

US Federal CTX Consolidated Format

The table below presents basic information about the US Federal CTX Consolidated Format.

US Federal CTX Consolidated Format Basic Information

Format Name	Format Type	Template Name	Extract Description
US Federal CTX Consolidated Format	Electronic	US CTX Consolidated Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the US Federal CTX Consolidated Format are presented in the table below.

US Federal CTX Consolidated Format Details

Payment Process Profile Name	Currency Name	Validation Name
US Federal CTX Consolidated	US Dollar	Federal Financials CTX Payment Validation

US Federal PPDP Consolidated Format

The table below presents basic information about the US Federal PPDP Consolidated Format.

US Federal PPDP Consolidated Format Basic Information

Format Name	Format Type	Template Name	Extract Description
US Federal PPDP Consolidated Format	Electronic	US PPDP Consolidated Format	Oracle Payments Funds Disbursement Payment Instruction Extract, Version 1.0

Additional details about the US Federal PPDP Consolidated Format are presented in the table below.

US Federal PPDP Consolidated Format Details

Payment Process Profile Name	Currency Name	Validation Name
US Federal PPDP Consolidated	US Dollar	Federal Financials Bulk Data PPD+ Payment Validation

Validations

Other Validations

This section lists validations for the Oracle e-Commerce Gateway Format.

Validation Name: e-Commerce Gateway Internal Bank Validation

Code: ECE_INTERNAL_BANK

Contents:

1. Validate Payer Bank Branch Type:
Payer bank branch type is not null.
Error Message: <Internal bank branch type> is required.

Validation Point: Document

Validation Assigned by Default to: Oracle e-Commerce Gateway payment format

Validation Name: e-Commerce Gateway Payee Validation

Code: ECE_PAYEE

Contents:

1. Validate Target Bank Branch Type:
Payee bank branch type is not null.
Error Message: <Payee bank branch type> is required.

Validation Point: Document

Validation Assigned by Default to: Oracle e-Commerce Gateway payment format

Validation Name: e-Commerce Gateway Document Validation

Code: ECE_DOC

Contents:

1. Validate payment method:
Payment method is not null.
Error Message: <Payment method> is required.

Validation Point: Document

Validation Assigned by Default to: Oracle e-Commerce Gateway payment format

Validation Name: e-Commerce Gateway Payee Validation

Code: ECE_PAYEE

Contents:

1. Validate bank instruction 2:
Bank instruction 2 is not null.
Error Message: <Bank instruction 2> is required.

Validation Point: Document

Validation Assigned by Default to: Oracle e-Commerce Gateway payment format

Validation Name: e-Commerce Gateway Document Validation

Code: ECE_DOC

Contents:

1. Validate delivery channel:
Delivery channel is not null.
Error Message: <Delivery channel> is required.

Validation Point: Document

Validation Assigned by Default to: Oracle e-Commerce Gateway payment format

Validation Name: e-Commerce Gateway Document Validation

Code: ECE_DOC

Contents:

1. Validate bank instruction 1:

Bank instruction 1 is not null if payment method code = ACH

Error Message: <Bank instruction 1> is required.

Validation Point: Document

Validation Assigned by Default to: Oracle e-Commerce Gateway payment format

Country-Specific Validations

This section lists validations for the following countries:

- Austria
- Belgium
- Finland
- France
- Germany
- Japan
- Poland
- Portugal
- Sweden
- United States
- U.S. Federal

Argentina

No Validations

Austria

The section lists the validations for Austria.

Validation Name: Austria International EFT Internal Bank

Code: AT_EFT_INT_INTERNAL_BANK

Contents:

1. Validate Payer Bank Sort Code:

Payer bank branch number is not null.

Error Message: <Internal bank branch number> is required.

Validation Point: Document

Validation Assigned by Default to: Austrian International EFT payment format

Validation Name: Austria International EFT Payer

Code: AT_EFT_INT_PAYER

Contents:

1. Validate Payer Company Description:

Payer name (company name, as defined in Legal Entity) is not null.

Error Message: <Payer name> is required.

2. Validate Payer Location Telephone Number:

Payer phone number is not null.

Error Message: <Payer telephone number> is required.

Validation Point: Document

Validation Assigned by Default to: Austrian International EFT payment format

Validation Name: Austria International EFT Payee

Code: AT_EFT_INT_PAYEE

Contents:

1. Validate Payee Bank Account Number:

Payee bank account number is not null.

Error Message: <Payee bank account number> is required.

2. Validate Payer Bank Sort Code:

Payee bank branch number is not null.

Error Message: <Payee bank branch country> is required.

Validation Point: Document

Validation Assigned by Default to: Austrian International EFT payment format

Belgium

This section lists the validations for Belgium.

Validation Name: Belgium Domestic EFT Document

Code: BE_EFT_DOM_DOC

Contents:

1. Validate Payment Currency:

Document payment currency is EUR (actually valid euro code).

Error Message: <Document payment currency> is invalid.

Validation Point: Document

Validation Assigned by Default to: Belgian Domestic EFT payment format

Validation Name: Belgium International EFT Payee

Code: BE_EFT_INT_PAYEE

Contents:

1. Validate Supplier Site Cost Code:

Bank charge bearer should be valid from bank charge bearers of Belgium.

Error Message: <Bank charge bearer> is invalid.

2. Validate Supplier Bank/Branch Number:

Payee bank number or payee bank branch number is not null.

Error Message: <Payee bank branch number> is required.

Validation Point: Document

Validation Assigned by Default to: Belgian International EFT payment format

Validation Name: Belgium International EFT Document

Code: BE_EFT_INT_DOC

Contents:

1. Validate payment method:

Payment method should be from valid payment methods for Belgian International EFT.

Error Message: <Payment method> is invalid.

Validation Point: Document

Validation Assigned by Default to: Belgian International EFT payment format

Validation Name: Belgium International EFT Payment

Code: P_BE_EFT_INT_PMT

Contents:**1. Validate Payment Amount:**

Payment amount <= Maximum check amount (a parameter, of value 9,999,999,999,999.99).

Error Message: <Payment amount> is invalid.

2. Validate Number of IBLC Codes:

The number of payment reasons <= Max payment reason number (a parameter, of value 24).

Error Message: <Number of payment reasons> is incorrect.

Parameter: Parameter Code: MAX_CHECK_AMOUNT (of value 9,999,999,999,999.99).

Validation: P_BE_EFT_INT_TRXN

Parameter Code: MAX_PAYMENT_REASON_NUMBER (of value 24).

Validation: P_BE_EFT_INT_TRXN

Validation Point: Payment

Validation Assigned by Default to: Belgian International EFT payment format

Validation Name: Belgium International EFT Payment Instruction

Code: I_BE_EFT_INT_INSTR

Contents:**1. Validate Number of Detail Records:**

The number of documents <= Max number of documents (a parameter, of value 999999).

Error Message: <Number of documents> is invalid.

Parameter: Parameter Code: MAX_NUMBER_OF_DOCUMENTS (of value 999999).

Validation: I_BE_EFT_INT_TRXN

Validation Point: Payment Instruction

Validation Assigned by Default to: Belgian International EFT payment format

Brazil

No Validations

Chile

No Validations

Colombia

No Validations

Czechoslovakia

No Validations

Finland

This section lists the validations for Finland.

Validation Name: Finland Domestic EFT Payment

Code: P_CITI_FI_EFT_DOM_PMT

Contents:

1. Validate Payment Amount:

Length of payment amount ≤ 12 .

Error Message: <Payment amount> has an incorrect length.

Validation Point: Payment

Validation Assigned by Default to: Finnish Domestic EFT payment format

France

This section lists the validations for France.

Germany

This section lists the validations for Germany.

Validation Name: Germany Domestic EFT Internal Bank

Code: DE_EFT_DOM_INTERNAL_BANK

Contents:

1. Validate User EFT Number:

Length of internal user EFT number ≤ 10 .

Error Message: <EFT number> has an incorrect length.

2. Validate User Bank BLZ:

Internal bank branch number is not null.

Error Message: <Internal bank branch number> is required.

Internal bank branch number does not start with 0 or 1.

Error Message: <Internal bank branch number> is incorrect.

Validation Point: Document

Validation Assigned by Default to: German Domestic EFT

Validation Name: Germany Domestic EFT Payer

Code: DE_EFT_DOM_PAYER

Contents:

1. Validate Location Description for Organization ID:

Payer LE name is not null.

Error Message: <Payer name> is required.

Validation Point: Document

Validation Assigned by Default to: German Domestic EFT

Validation Name: Germany Domestic EFT Payee

Code: DE_EFT_DOM_PAYEE

Contents:

1. Validate Customer/Supplier Bank Name:

External bank name is not null.

Error Message: <Payee bank name> is required.

2. Validate Customer/Supplier Bank BLZ:

External bank branch number is not null.

Error Message: <Payee bank branch number> is required.

External bank branch number does not start with 0 or 1.

Error Message: <Payee bank branch number> is incorrect.

Validation Point: Document

Validation Assigned by Default to: German Domestic EFT

Validation Name: Germany Domestic EFT Payment

Code: P_DE_EFT_DOM_PMT

Contents:

1. Validate Payment Amount:
Payment amount is not 0.
Error Message: <Payee amount> is invalid.

Validation Point: Payment

Validation Assigned by Default to: German Domestic EFT

Validation Name: Germany International EFT Internal Bank

Code: DE_EFT_INT_INTERNAL_BANK

Contents:

1. Validate User EFT:
Length of internal user EFT number <= 10.
Error Message: <EFT number> has an incorrect length.
2. Validate User Bank BLZ:
Internal bank branch number is not null.
Error Message: <Internal bank branch number> is required.
Internal bank branch number does not start with 0 or 9.
Error Message: <Internal bank branch number> is incorrect.

Validation Point: Document

Validation Assigned by Default to: German International EFT

Validation Name: Germany International EFT Payee

Code: DE_EFT_INT_PAYEE

Contents:

1. Validate Customer/Supplier Bank Name:
External bank name is not null.
Error Message: <Payee bank name> is required.
2. Validate Document Country:
Payee party site country and payer LE country are same.

Error Message: <Payee country> is incorrect.

Validation Point: Document

Validation Assigned by Default to: German International EFT

Validation Name: Germany International EFT Payer

Code: DE_EFT_INT_PAYER

Contents:

1. Validate EFT Company Number:

Internal bank assigned ID 2 is not null.

Length of internal bank assigned Id 2 <= 8.

Internal bank assigned ID 2 is not null.

Error Message: Bank assigned identifier 2 has an incorrect length.

2. Validate EFT LZB Area:

Internal bank assigned ID 1 is not null.

Error Message: Bank assigned identifier 1 is required.

Length of internal bank assigned ID 1 <= 2.

Error Message: Bank assigned identifier 1 has an incorrect length.

Validation Point: Document

Validation Assigned by Default to: German International EFT

Validation Name: Germany International EFT Payment

Code: P_DE_EFT_INT_PMT

Contents:

1. Validate Payment Amount:

Payment amount is not 0.

Error Message: <Payment amount> is invalid.

Validation Point: Payment

Validation Assigned by Default to: German International EFT

Hungary

No Validations

Japan

This section lists the validations for Japan.

Validation Name: Japan Zengin EFT Internal Bank

Code:JP_EFT_ZENGIN_INTERNAL_BANK

Contents:

1. Validate Internal Bank Account Type:
Internal bank account type is like 1%, 2%, 4% or 9%.
Error Message: <Internal bank account type> is incorrect.
2. Validate EFT Requester Id:
Internal EFT user number contains digits only.
Error Message: <EFT user number> is incorrect.

Validation Point: Document

Validation Assigned by Default to: Japan Zengin EFT

Validation Name: Japan Zengin EFT Payee

Code: JP_EFT_ZENGIN_PAYEE

Contents:

1. Validate External Bank Account Type:
Length of payment Id <= 16.
Error Message: <Payment reference> has an incorrect length.
2. Validate Transaction Amount:
External bank account type is like 1%, 2%, 4% or 9%.
Error Message: <Payee bank account type> is incorrect.

Validation Point: Document

Validation Assigned by Default to: Japan Zengin EFT

Luxembourg

No Validations

Netherlands

No Validations

Poland

This section lists the validations for Poland.

Validation Name: Pekao Poland Transfer Format Internal Bank

Code: PEKAO_PL_WIRE_INTERNAL_BANK

Contents:

1. Validate Payer Account Number:

Internal bank account IBAN number is not null.

Error Message: <Internal bank account IBAN number> is required.

Internal bank account number is not null.

Error Message: <Internal bank account number> is required.

Length of internal bank account number <= 34.

Error Message: <Internal bank account number> has an incorrect length.

Validation Point: Document

Validation Assigned by Default to: Pekao Poland Transfer Format

Validation Name: Pekao Poland Transfer Format Payee

Code: PEKAO_PL_WIRE_PAYEE

Contents:

1. Validate Supplier Bank Account Number:

External bank account IBAN number is not null.

Error Message: <Payee bank account IBAN number> is required.

External bank account number is not null.

Error Message: <Payee bank account number> is required.

Length of external bank account number <= 34.

Error Message: <Payee bank account number> has an incorrect length.

2. Validate Company Tax Payer Id:

Length of payee tax payer Id <= 15.

Error Message: <Payee Taxpayer ID> has an incorrect length.

3. Validate Supplier Detail:

Length of payee party name, payee party site name, payee party site address line 1, payee party site city and payee party site postal code concatenated <= 143.

Error Message: <Payee name> has an incorrect length.

4. Validate Supplier Address:

Length of payee party site address line 1 <= 35.

Error Message: <Payee address line 1> has an incorrect length.

Validation Point: Document

Validation Assigned by Default to: Pekao Poland Transfer Format

Validation Name: Pekao Poland Transfer Format Document

Code: PEKAO_PL_WIRE_ST_DOC

Contents:

1. Validate Insurance Premium Type:

Payment reason code is not null.

Error Message: <Payment reason> is required.

2. Validate Number of Invoice:

Pay alone flag is Y.

Error Message: <Pay Alone> is required.

3. Validate Declaration Number:

Calling application document reference number does not start with 99.

Error Message: <Document reference number> is incorrect.

Validation Point: Document

Validation Assigned by Default to: Pekao Poland Transfer Format

Validation Name: Pekao Poland Transfer Format Payment Validation

Code: P_PEKAO_PL_WIRE_TR_PMT

Contents:

1. Validate Payment Amount:

Payment amount <= Maximum payment amount allowed (Value of

999999999999.99, a parameter).

Error Message: <Payment amount> exceeds maximum allowed.

2. Validate Concatenated List of Invoices:

Length of payment detail <= 140.

Error Message: <Payment detail> has an incorrect length.

Validation Point: Payment

Validation Assigned by Default to: Pekao Poland Transfer Format

Validation Name: Pekao Poland Standard Payment Internal Bank

Code: PEKAO_PL_EFT_INTERNAL_BANK

Contents:

1. Validate Payer Account Number:

Internal bank account IBAN number is not null.

Error Message: <Internal bank account IBAN number> is required.

Internal bank account number is not null.

Error Message: <Internal bank account number> is required.

Length of internal bank account number <= 34.

Error Message: <Internal bank account number> has an incorrect length.

Validation Point: Document

Validation Assigned by Default to: Pekao Poland Standard Payment

Validation Name: Pekao Poland Standard Payment Payee

Code: PEKAO_PL_EFT_PAYEE

Contents:

1. Validate Supplier Bank Account Number:

External bank account IBAN number is not null.

Error Message: <Payee bank account IBAN number> is required.

External bank account number is not null.

Error Message: <Payee bank account number> is required.

Length of external bank account number <= 34.

Error Message: <Payee bank account number> has an incorrect length.

Validation Point: Document

Validation Assigned by Default to: Pekao Poland Standard Payment

Portugal

This section lists the validations for Portugal.

Validation Name: Portugal Check Generic

Code: PT_CHECK_GEN

Contents:

1. Validate First Party Office Site:
Payer LE name is not null.
Error Message: <Payer name> is required.

Validation Point: Document

Validation Assigned by Default to: Portuguese Check Format

Sweden

This section lists the validations for Sweden.

Validation Name: Sweden EFT Bankgiro Inland Document

Code: SE_EFT_BANKGIRO_INLAND_DOC

Contents:

1. Validate Payment Currency:
Payment currency is SEK or EUR.
Error Message: <Payment currency> is invalid.
2. Validate Payment Type:
Delivery channel code is not CHECK.
Error Message: <Delivery channel> is incorrect.

Validation Point: Document

Validation Assigned by Default to: Sweden EFT Bankgiro Inland

Validation Name: Sweden EFT Bankgiro Utland SISU Document

Code: SE_EFT_BANKGIRO_UTLAND_SI_DOC

Contents:

1. Validate Payment Type:

Delivery channel code is not KONTTOAVI, AVI, or GIRO.

Error Message: <Delivery channel> is incorrect.

Validation Point: Document

Validation Assigned by Default to: Sweden EFT Bankgiro Utland SISU

Validation Name: Sweden EFT Bankgiro Utland UTLI Document

Code: SE_EFT_BANKGIRO_UTLAND_UT_DOC

Contents:

1. Validate Payment Type:

Delivery channel code is not KONTTOAVI, AVI, or GIRO.

Error Message: <Delivery channel> is incorrect.

Validation Point: Document

Validation Assigned by Default to: Sweden EFT Bankgiro Utland UTLI

Validation Name: Sweden EFT Postgiro Inland Document

Code: SE_EFT_POSTGIRO_INLAND_DOC

Contents:

1. Validate Payment Type:

Delivery channel code is not KONTTOAVI or CHECK.

Error Message: <Delivery channel> is incorrect.

Validation Point: Document

Validation Assigned by Default to: Sweden EFT Postgiro Inland

Validation Name: Sweden EFT Postgiro Utland Document

Code: SE_EFT_POSTGIRO_UTLAND_DOC

Contents:

1. Validate Payment Type:

Length of payment Id <= 12.

Error Message: <Delivery channel> is incorrect.

Validation Point: Document

Validation Assigned by Default to: Sweden EFT Postgiro Utland

Switzerland

This section lists the validations for Switzerland.

Validation Name: Switzerland Generic EFT Payee

Code: CH_EFT_GEN_PAYEE

Contents:

1. Validate Bank Account Type:
Delivery channel (code) should be valid from delivery channels of Switzerland.
Error Message: <Delivery channel> is invalid.
2. Validate Bank Account Number:
External bank account number is not null, when delivery channel is BANK or DATACHECK.
Error Message: <Payee bank account number> is required.

Validation Point: Document

Validation Assigned by Default to: Swiss International EFT and Swiss Domestic EFT

Validation Name: Switzerland Generic EFT Document

Code: CH_EFT_GEN_DOC

Contents:

1. Validate Document Payment Currency:
When the delivery channel is BANK, document payment currency is CHF.
Error Message: <Document payment currency> is invalid.
2. Validate Exclusive Payment Flag for ESR payments:
When the delivery channel is ESR, exclusive payment flag is Y.
Error Message: <Pay Alone> is required.
3. Validate ESR Number:
When the delivery channel is ESR, unique remittance Id code is not null, of length 15, 16, or 27, numeric, and if its length is 15, it satisfies the check algorithm.
Error Message: <Unique remittance identifier> is invalid.
4. Validate Invoice Amount for ESR Payments:
When the delivery channel is ESR, document amount is greater than or equal to 0.

Error Message: <Document amount> is incorrect.

5. Validate ESR number for non-ESR Payments:

When the delivery channel is ESR, unique remittance Id code must be null.

Error Message: <Unique remittance identifier> is not allowed.

Validation Point: Document

Validation Assigned by Default to: Swiss International EFT and Swiss Domestic EFT

Turkey

No Validations

United States

This section lists the validations for the United States.

Validation Name: US NACHA Internal Bank

Code: US_NACHA_INTERNAL_BANK

Contents:

1. Validate Originating Bank Routing Number:

Length of internal bank branch number = 9.

Error Message: <Internal bank branch number> has an incorrect length.

Validation Point: Document

Validation Assigned by Default to: Nacha EFT

Validation Name: US NACHA Payee

Code: US_NACHA_PAYEE

Contents:

1. Validate Supplier Bank Routing Number:

Length of external bank branch number = 9.

Error Message: <Payee bank branch number> has an incorrect length.

Validation Point: Document

Validation Assigned by Default to: NACHA EFT

Validation Name: US NACHA Payment Instruction

Code: I_US_NACHA_INSTR

Contents:**1. Validate Payment Batch Total Amount:**

Instruction amount <= Maximum Nacha amount (a parameter of value 100,000,000).

Error Message: <Payment instruction total amount> exceeds maximum allowed.

Parameter: Parameter Code: MAX_PAYMENT_INSTR_AMOUNT (of value 100,000,000).

Validation: I_US_NACHA_INSTR

Validation Point: Payment Instruction

Validation Assigned by Default to: NACHA EFT

U.S. Federal

This section lists the U.S. Federal validations.

Federal Financials Bulk Data CCD+ Payment Validation

Code: FVBLCCDP

Contents:**1. Validate Agency Location Code:**

Agency Location Code must be 8 characters.

Error Message: The Agency Location Code, captured in the bank account details, is not defined for bank account <bank account>.

2. Validate Routing Transit Number:

Routing Transit Number is stored in the bank branch number. It must be a nine digit numeric-only field. The ninth digit is the Check Digit, which is validated using the Modulus formula.

Error Message: Invalid routing number.

3. Validate Supplier Type:

Supplier Type of the Supplier must match the Supplier Type of the documents payable.

Error Message: The vendor for invoice <invoice number> cannot be of type Employee.

4. Validate Maximum Treasury Symbols:

A maximum of 10 treasury symbols can be included in a payment instruction, unless profile FV: Limit Bulk Format to 10 Treasury Symbol is set to No.

Error Message: This error message refers to obsolete entities, and is therefore invalid. Payment format aborts if it contains more than 10 Treasury symbols.

5. Validate Federal Employee Identification Number:

Federal Employee Identification Number is required.

Error Message: The Federal Employer ID Number (FEIN) must be defined on the Define Federal Options window in Federal Administrator for the operating unit <operating unit>.

6. Validate Payee Social Security Number:

Social Security Number/Tax Identification Number is required.

Error Message: SSN/TIN must be supplied for supplier <supplier name>.

7. Validate Payee Bank Account Number:

Payee Bank Account Number is required.

Error Message: Bank account number missing for vendor.

8. Validate Bank Account Type:

Valid values for bank account type are CHECKING for Checking account and SAVINGS for Savings account.

Error Message: The Payee bank account must have a bank account type of either Checking or Savings.

9. Validate RFC Identifier:

RFC Identifier is required.

Error Message: RFC Identifier not defined in the Banks Window for Bank <bank name>.

10. Validate Schedule Number:

Schedule number is derived from the reference assigned by the administrator if specified on a payment instruction, or else the payment instruction ID becomes the schedule number.

Error Message: The reference assigned by the administrator number must contain only valid characters of 0-9, A-Z, or dash (-) and the first character must not be a zero.

11. Validate Treasury Account Symbols:

Must contain only the following characters: 0-9, A-Z, ".", "(", ")", or "/".

Error Message: Treasury symbols should only contain the following characters: 0-9, A-Z, ".", "(", ")", or "/".

12. Validate Pay Alone Option:

Pay alone option is required on the invoice.

Error Message: Invoices for this payment format must have the Pay Alone flag checked on the invoice <invoice number>.

Validation Point: Payment Instruction

Validation Assigned by Default to: US Federal Bulk CCDP Format AND US Federal CCDP Consolidated Format

Federal Financials Bulk Data NCR Payment Validation

Code: FVBLNCR

Contents:

1. Validate Agency Location Code:

Agency Location Code must be 8 characters.

Error Message: The Agency Location Code, captured in the bank account details, is not defined for bank account <bank account>.

2. Validate Agency Address:

Agency Address is required.

Error Message: Invalid address for Agency: <agency name>.

3. Validate Payment Amount:

Payment Amount must be less than 9,999,999.999 USD.

Error Message: Payment Amount exceeds the limit of \$9,999,999.99.

4. Validate Payee Address:

At least one address line is required.

Error Message: Invalid address for Vendor <vendor name>.

5. Validate Payment Reason:

Payment Reason required depends on the type of supplier in the payment. There are two different validations that occur, which depend on whether the supplier is an employee or non-employee.

Error Message: Payments to an Internal Employee can only have the following payment reason codes: SSA, VA, SSI, OPM, or RRB Benefit or Tax. Alternatively, payments to a Standard Supplier can only have a payment reason code of Vendor Payment Sub-Type.

6. Validate Supplier Type:

Supplier Types within a payment instruction must be all employee types of suppliers or non-employee types of suppliers.

Error Message: All selected invoices must have one vendor type, either Employee or Non-employee.

7. Validate Maximum Treasury Symbols:

A maximum of 10 treasury symbols can be included in a payment instruction, unless profile FV: Limit Bulk Format to 10 Treasury Symbol is set to No.

Error Message: This error message refers to obsolete entities and is therefore invalid. Payment format aborts if it contains more than 10 Treasury symbols.

8. Validate Payee Social Security Number:

Social Security Number/Tax Identification Number is required.

Error Message: SSN/TIN must be supplied for supplier <supplier name>.

9. Validate RFC Identifier:

RFC Identifier is required.

Error Message: RFC Identifier not defined in the Banks Window for Bank <bank name>.

10. Validate Schedule Number:

Schedule number is derived from the reference assigned by the administrator if specified on a payment instruction, or else the payment instruction ID becomes the schedule number.

Error Message: The reference assigned by the administrator number must contain only valid characters of 0-9, A-Z, or dash (-) and the first character must not be a zero.

Validation Point: Payment Instruction

Validation Assigned by Default to: US Federal Bulk NCR Format

Federal Financials Bulk Data PPD+ Payment Validation

Code: FVBLPPDP

Contents:

1. Validate Agency Location Code:

Agency Location Code must be 8 characters.

Error Message: The Agency Location Code, captured in the bank account details, is not defined for bank account <bank account>.

2. Validate Routing Transit Number:

Routing Transit Number is stored in the bank branch number. It must be a nine digit numeric-only field. The ninth digit is the Check Digit, which is validated using the Modulus formula.

Error Message: Invalid routing number.

3. Validate Supplier Type:

Supplier Type of the Supplier must match the Supplier Type of the documents payable.

Error Message: The vendor for invoice <invoice number> must be of type Employee.

4. Validate Maximum Treasury Symbols:

A maximum of 10 treasury symbols can be included in a payment instruction, unless profile FV: Limit Bulk Format to 10 Treasury Symbol is set to No.

Error Message: This error message refers to obsolete entities and is therefore invalid. Payment format aborts if it contains more than 10 Treasury symbols.

5. Validate Federal Employee Identification Number:

Federal Employee Identification Number is required.

Error Message: The Federal Employer ID Number (FEIN) must be defined on the Define Federal Options window in Federal Administrator for the operating unit <operating unit>.

6. Validate Payee Social Security Number:

Social Security Number/Tax Identification Number is required.

Error Message: SSN/TIN must be supplied for supplier <supplier name>.

7. Validate Payee Bank Account Number:

Payee Bank Account Number is required.

Error Message: Bank account number missing for vendor.

8. Validate Bank Account Type:

Valid values for bank account type are CHECKING for Checking account and SAVINGS for Savings account.

Error Message: The Payee bank account must have a bank account type of either Checking or Savings.

9. Validate RFC Identifier:

RFC Identifier is required.

Error Message: RFC Identifier not defined in the Banks Window for Bank <bank

name>.

10. Validate Schedule Number:

Schedule number is derived from the reference assigned by the administrator if specified on a payment instruction, or else the payment instruction ID becomes the schedule number.

Error Message: The reference assigned by the administrator number must contain only valid characters of 0-9, A-Z, or dash (-) and the first character must not be a zero.

11. Validate Treasury Account Symbols:

Must contain only the following characters: 0-9, A-Z, ".", "(", ")", or "/".

Error Message: Treasury symbols should only contain the following characters: 0-9, A-Z, ".", "(", ")", or "/".

12. Validate Pay Alone Option:

Pay alone option is required on the invoice.

Error Message: Invoices for this payment format must have the Pay Alone flag checked on the invoice <invoice number>.

13. Validate Reason Code:

Validate a Federal payment reason is assigned to a payment.

Error Message: This payment format must have a Federal payment reason defined for each payment. The following are valid payment reasons: Allotments, SSA Benefits, VA Benefits, VAINS, Miscellaneous PPD, OPM Benefits, RRB Benefits, Salary, Travel, and Tax.

14. Validate Maximum Payment Amount:

A payment amount cannot exceed \$999,999.99 USD.

Error Message: Payment Amount exceeds the limit of \$999,999.99.

Validation Point: Payment Instruction

Validation Assigned by Default to: US Federal Bulk PPDP Format AND US Federal PPDP Consolidated Format

Federal Financials Bulk Data Salary/Travel NCR Payment Validation

Code: FVBLSLTR

Contents:

1. Validate Agency Location Code:

Agency Location Code must be 8 characters.

Error Message: The Agency Location Code, captured in the bank account details, is not defined for bank account <bank account>.

2. Validate Agency Address:

At least one address line, city, state, or ZIP is required

Error Message: Invalid address for Agency <agency name>.

3. Validate Payment Reason Code:

Payments for this format can only contain invoices with a payment reason of Salary or Travel.

Error Message: The Check Salary/Travel NCR payments can only be generated for Salary or Travel payments. The Reason Code must be related to Salary or Travel.

4. Validate Supplier Type:

Supplier Type of the Supplier must match the Supplier Type of the documents payable.

Error Message: The vendor for invoice <invoice number> must be of type Employee.

5. Validate Maximum Treasury Symbols:

A maximum of 10 treasury symbols can be included in a payment instruction, unless profile FV: Limit Bulk Format to 10 Treasury Symbol is set to No.

Error Message: This error message refers to obsolete entities and is therefore invalid. Payment format aborts if it contains more than 10 Treasury symbols.

6. Validate Payee Social Security Number:

Social Security Number/Tax Identification Number is required.

Error Message: SSN/TIN must be supplied for supplier <supplier name>.

7. Validate RFC Identifier:

RFC Identifier is required.

Error Message: RFC Identifier not defined in the Banks Window for Bank <bank name>.

8. Validate Schedule Number:

Schedule number is derived from the reference assigned by the administrator if specified on a payment instruction, or else the payment instruction ID becomes the schedule number.

Error Message: The reference assigned by the administrator number must contain only valid characters of 0-9, A-Z, or dash (-) and the first character must not be a

zero.

9. Validate Maximum Payment Amount:

A payment amount cannot exceed \$999,999.99 USD.

Error Message: Payment Amount exceeds the limit of \$999,999.99.

Validation Point: Payment Instruction

Validation Assigned by Default to: US Federal Bulk Salary and Travel NCR Format

Federal Financials SPS CCD Payment Validation

Code: FVSPCCD

Contents:

1. Validate Agency Location Code:

Agency Location Code must be 8 characters.

Error Message: The Agency Location Code, captured in the bank account details, is not defined for bank account <bank account>.

2. Validate Routing Transit Number:

Routing Transit Number is stored in the bank branch number. It must be a nine digit numeric-only field. The ninth digit is the Check Digit, which is validated using the Modulus formula.

Error Message: Invalid routing number.

3. Validate Supplier Type:

Supplier Type of the Supplier must match the Supplier Type of the documents payable.

Error Message: The vendor for invoice <invoice number> cannot be of type Employee.

4. Validate Maximum Treasury Symbols:

A maximum of 10 treasury symbols included in a payment instruction.

Error Message: This error message refers to obsolete entities and is therefore invalid. Payment format aborts if it contains more than 10 Treasury symbols.

5. Validate Payee Social Security Number:

Social Security Number/Tax Identification Number is required.

Error Message: SSN/TIN must be supplied for supplier <supplier name>.

6. Validate Payee Bank Account Number:

Payee Bank Account Number is required.

Error Message: Bank account number missing for vendor.

7. Validate Bank Account Type:

Valid values for bank account type are CHECKING for Checking account and SAVINGS for Savings account.

Error Message: The Payee bank account must have a bank account type of either Checking or Savings.

8. Validate Maximum Treasury Symbols:

Must contain only the following characters: 0-9, A-Z, ".", "(", ")", or "/".

Error Message: Treasury Symbols should only contain the following characters: 0-9, A-Z, ".", "(", ")", or "/".

9. Validate Treasury Symbols:

Must contain at least seven characters.

Error Message: The Treasury Symbol must contain a minimum of 7 characters.

10. Validate Schedule Number:

Schedule number is derived from the reference assigned by the administrator if specified on a payment instruction, or else the payment instruction ID becomes the schedule number.

Error Message: The reference assigned by the administrator number must contain only valid characters of 0-9, A-Z, or dash (-) and the first character must not be a zero.

11. Validate Pay Alone Option:

Pay alone option is required on the invoice.

Error Message: Invoices for this payment format must be have the Pay Alone flag checked on the invoice <invoice number>.

Validation Point: Payment Instruction

Validation Assigned by Default to: US Federal SPS CCD Format

Federal Financials SPS CCD+ Payment Validation

Code: FVSPCCDP

Contents:

1. Validate Agency Location Code:

Agency Location Code must be 8 characters.

Error Message: The Agency Location Code, captured in the bank account details, is not defined for bank account <bank account>.

2. Validate Routing Transit Number:

Routing Transit Number is stored in the bank branch number. It must be a nine digit numeric-only field. The ninth digit is the Check Digit, which is validated using the Modulus formula.

Error Message: Invalid routing number.

3. Validate Supplier Type:

Supplier Type of the Supplier must match the Supplier Type of the documents payable.

Error Message: The vendor for invoice <invoice number> cannot be of type Employee.

4. Validate Maximum Treasury Symbols:

A maximum of 10 treasury symbols included in a payment instruction.

Error Message: This error message refers to obsolete entities and is therefore invalid. Payment format aborts if it contains more than 10 Treasury symbols.

5. Validate Payee Social Security Number:

Social Security Number/Tax Identification Number is required.

Error Message: SSN/TIN must be supplied for supplier <supplier name>.

6. Validate Payee Bank Account Number:

Payee Bank Account Number is required.

Error Message: Bank account number missing for vendor.

7. Validate Bank Account Type:

Valid values for bank account type are CHECKING for Checking account and SAVINGS for Savings account.

Error Message: The Payee bank account must have a bank account type of either Checking or Savings.

8. Validate Treasury Account Symbols:

Must contain only the following characters: 0-9, A-Z, ".", "(", ")", or "/".

Error Message: Treasury Symbols should only contain the following characters: 0-9, A-Z, ".", "(", ")", or "/".

9. Validate Schedule Number:

Schedule number is derived from the reference assigned by the administrator if specified on a payment instruction, or else the payment instruction ID becomes the schedule number.

Error Message: The reference assigned by the administrator number must contain only valid characters of 0-9, A-Z, or dash (-) and the first character must not be a zero.

10. Validate Pay Alone Option:

Pay alone option is required on the invoice.

Error Message: Invoices for this payment format must have the Pay Alone flag checked on the invoice <invoice number>.

Validation Point: Payment Instruction

Validation Assigned by Default to: US Federal SPS CCDP Format

Federal Financials SPS NCR Payment Validation

Code: FVSPNCR

Contents:

1. Validate Agency Location Code:

Agency Location Code must be 8 characters.

Error Message: The Agency Location Code, captured in the bank account details, is not defined for bank account <bank account>.

2. Validate Payee Address:

At least one address line is required.

Error Message: Invalid address for Vendor <vendor name>.

3. Validate Payment Reason:

Payment Reason required depends on the type of supplier in the payment. There are two different validations occurring, depending on whether the supplier is an employee or non-employee.

Error Message: Payments to an Internal Employee can only have the following payment reason codes: SSA, VA, SSI, OPM, or RRB Benefit or Tax. Alternatively, Payments to a Standard Supplier can only have a payment reason code of Vendor Payment Sub-Type.

4. Validate Supplier Type:

Supplier Types within a payment instruction must be all employee types of suppliers or non-employee types of suppliers.

Error Message: All selected invoices must have one vendor type, either Employee or Non-employee.

5. Validate Maximum Treasury Symbols:

A maximum of 10 treasury symbols included in a payment instruction.

Error Message: This error message refers to obsolete entities and is therefore invalid. Payment format aborts if it contains more than 10 Treasury symbols.

6. Validate Treasury Account Symbols:

Must contain only the following characters: 0-9, A-Z, ".", "(", ")", or "/".

Error Message: Treasury Symbols should only contain the following characters: 0-9, A-Z, ".", "(", ")", or "/".

7. Validate Schedule Number:

Schedule number is derived from the reference assigned by the administrator if specified on a payment instruction, or else the payment instruction ID becomes the schedule number.

Error Message: The reference assigned by the administrator number must contain only valid characters of 0-9, A-Z, or dash (-) and the first character must not be a zero.

8. Validate Payment Amount:

Payment Amount must be less than 9,999,999.999 USD.

Error Message: Payment Amount exceeds the limit of \$9,999,999.99.

Validation Point: Payment Instruction

Validation Assigned by Default to: US Federal SPS NCR Format

Federal Financials SPS PPD Payment Validation

Code: FVSPPPD

Contents:

1. Validate Agency Location Code:

Agency Location Code must be 8 characters.

Error Message: The Agency Location Code, captured in the bank account details, is not defined for bank account <bank account>.

2. Validate Routing Transit Number:

Routing Transit Number is stored in the bank branch number. It must be a nine digit numeric-only field. The ninth digit is the Check Digit, which is validated using the Modulus formula.

Error Message: Invalid routing number.

3. Validate Supplier Type:

Supplier Type of the Supplier must match the Supplier Type of the documents payable.

Error Message: The vendor for invoice <invoice number> must be of type Employee.

4. Validate Maximum Treasury Symbols:

A maximum of 10 treasury symbols included in a payment instruction.

Error Message: This error message refers to obsolete entities and is therefore invalid. Payment format aborts if it contains more than 10 Treasury symbols.

5. Validate Payee Social Security Number:

Social Security Number/Tax Identification Number is required.

Error Message: SSN/TIN must be supplied for supplier <supplier name>.

6. Validate Payee Bank Account Number:

Payee Bank Account Number is required.

Error Message: Bank account number missing for vendor.

7. Validate Bank Account Type:

Valid values for bank account type are CHECKING for Checking account and SAVINGS for Savings account.

Error Message: The Payee bank account must have a bank account type of either Checking or Savings.

8. Validate Treasury Account Symbols:

Must contain only the following characters: 0-9, A-Z, ".", "(", ")", or "/".

Error Message: Treasury Symbols should only contain the following characters: 0-9, A-Z, ".", "(", ")", or "/".

9. Validate Schedule Number:

Schedule number is derived from the reference assigned by the administrator if specified on a payment instruction, or else the payment instruction ID becomes the schedule number.

Error Message: The reference assigned by the administrator number must contain only valid characters of 0-9, A-Z, or dash (-) and the first character must not be a zero.

10. Validate Pay Alone Option:

Pay alone option is required on the invoice.

Error Message: Invoices for this payment format must have the Pay Alone flag checked on the invoice <invoice number>.

11. Validate Reason Code:

Validate a Federal payment reason that is assigned to a payment.

Error Message: This payment format must have a Federal payment reason defined for each payment. The following are valid payment reasons: Allotments, SSA Benefits, VA Benefits, VAINS, Miscellaneous PPD, OPM Benefits, RRB Benefits, Salary, Travel, and Tax.

12. Validate Maximum Payment Amount:

A payment amount cannot exceed \$999,999.99 USD.

Error Message: Payment Amount exceeds the limit of \$999,999.99.

Validation Point: Payment Instruction

Validation Assigned by Default to: US Federal SPS PPD Format

Federal Financials SPS PPD+ Payment Validation

Code: FVSPPPDP

Contents:

1. Validate Agency Location Code:

Agency Location Code must be 8 characters.

Error Message: The Agency Location Code, captured in the bank account details, is not defined for bank account <bank account>.

2. Validate Routing Transit Number:

Routing Transit Number is stored in the bank branch number. It must be a nine digit numeric-only field. The ninth digit is the Check Digit, which is validated using the Modulus formula.

Error Message: Invalid routing number.

3. Validate Supplier Type:

Supplier Type of the Supplier must match the Supplier Type of the documents payable.

Error Message: The vendor for invoice <invoice number> must be of type Employee.

4. Validate Maximum Treasury Symbols:

A maximum of 10 treasury symbols included in a payment instruction.

Error Message: This error message refers to obsolete entities and is therefore invalid. Payment format aborts if it contains more than 10 Treasury symbols.

5. Validate Payee Social Security Number:

Social Security Number/Tax Identification Number is required.

Error Message: SSN/TIN must be supplied for supplier <supplier name>.

6. Validate Payee Bank Account Number:

Payee Bank Account Number is required.

Error Message: Bank account number missing for vendor.

7. Validate Bank Account Type:

Valid values for bank account type are CHECKING for Checking account and SAVINGS for Savings account.

Error Message: The Payee bank account must have a bank account type of either Checking or Savings.

8. Validate Treasury Account Symbols:

Must contain only the following characters: 0-9, A-Z, ".", "(", ")", or "/".

Error Message: Treasury Symbols should only contain the following characters: 0-9, A-Z, ".", "(", ")", or "/".

9. Validate Schedule Number:

Schedule number is derived from the reference assigned by the administrator if specified on a payment instruction, or else the payment instruction ID becomes the schedule number.

Error Message: The reference assigned by the administrator number must contain only valid characters of 0-9, A-Z, or dash (-) and the first character must not be a zero.

10. Validate Pay Alone Option:

Pay alone option is required on the invoice.

Error Message: Invoices for this payment format must have the Pay Alone flag checked on the invoice <invoice number>.

11. Validate Reason Code:

Validate that a Federal payment reason is assigned to a payment.

Error Message: This payment format must have a Federal payment reason defined for each payment. The following are valid payment reasons: Allotments, SSA Benefits, VA Benefits, VAINS, Miscellaneous PPD, OPM Benefits, RRB Benefits, Salary, Travel, and Tax.

12. Validate Maximum Payment Amount:

A payment amount cannot exceed \$999,999.99 USD.

Error Message: Payment Amount exceeds the limit of \$999,999.99.

Validation Point: Payment Instruction

Validation Assigned by Default to: US Federal SPS PPDP Format

Federal Financials ECS NCR Payment Validation

Code: FVTIACHB

Contents:

1. Validate RFC Identifier:

RFC Identifier is required.

Error Message: RFC Identifier not defined in the Banks Window for Bank <bank name>.

2. Validate Agency Address:

At least one address line, city, state, or ZIP is required.

Error Message: Invalid address for Agency: <agency name>.

3. Validate Payee Address:

At least one address line is required.

Error Message: Invalid address for Vendor <vendor name>.

4. Validate Payment Reason:

Payment Reason required depends on the type of supplier in the payment. There are two different validations that occur, depending on whether the supplier is an employee or non-employee.

Error Message: Payments to an Internal Employee can only have the following payment reason codes: SSA, VA, SSI, OPM, or RRB Benefit or Tax. Alternatively, payments to a Standard Supplier can only have a payment reason code of Vendor Payment Sub-Type.

5. Validate Supplier Type:

Supplier Types within a payment instruction must be employee types of suppliers or non-employee types of suppliers.

Error Message: All selected invoices must have one vendor type, either Employee or Non-employee.

6. Validate Schedule Number:

Schedule Number is derived from the reference assigned by the administrator if specified on a payment instruction, or else the payment instruction ID becomes the schedule number.

Error Message: The reference assigned by the administrator number must contain only valid characters of 0-9, A-Z, or dash (-) and the first character must not be a zero.

7. Validate Payment Amount:

Payment Amount must be less than 9,999,999.999 USD.

Error Message: Payment Amount exceeds the limit of \$9,999,999.99.

Validation Point: Payment Instruction

Validation Assigned by Default to: US Federal ECS NCR Format

Federal Financials ECS CCD+ Payment Validation

Code: FVTIACHP

Contents:

1. Validate RFC Identifier:

RFC Identifier is required.

Error Message: RFC Identifier not defined in the Banks Window for Bank <bank name>.

2. Validate Agency Location Code:

Agency Location Code must be 8 characters.

Error Message: Agency Location Code, captured in the bank account details, is not defined for bank account <bank account>.

3. Validate Routing Transit Number:

Routing Transit Number is stored in the bank branch number. It must be a nine digit numeric-only field. The ninth digit is the Check Digit, which is validated using the Modulus formula.

Error Message: Invalid routing number.

4. Validate Agency Address:

At least one address line, city, state, or ZIP is required.

Error Message: Invalid address for Agency <agency name>.

5. Validate Supplier Type:

Supplier Type of the Supplier must match the Supplier Type of the documents payable.

Error Message: The vendor for invoice <invoice number> cannot be of type Employee.

6. Validate Maximum Treasury Symbols:

A maximum of 10 treasury symbols included in a payment instruction.

Error Message: Payment format aborts if it contains more than 10 Treasury symbols.

7. Validate Payee Social Security Number:

Social Security Number/Tax Identification Number is required.

Error Message: SSN/TIN must be supplied for supplier <supplier name>.

8. Validate Payee Bank Account Number:

Payee Bank Account Number is required.

Error Message: Bank account number missing for vendor.

9. Validate Bank Account Type:

Valid values for bank account type are CHECKING for Checking account and SAVINGS for Savings account.

Error Message: The Payee bank account must have a bank account type of either Checking or Savings.

10. Validate Schedule Number:

Schedule number is derived from the reference assigned by the administrator if specified on a payment instruction, or else the payment instruction ID becomes the schedule number.

Error Message: The reference assigned by the administrator number must contain only valid characters of 0-9, A-Z, or dash (-) and the first character must not be a zero.

11. Validate Pay Alone Option:

Pay alone option is required on the invoice.

Error Message: Invoices for this payment format must have the Pay Alone flag checked on the invoice <invoice number>.

Validation Point: Payment Instruction

Validation Assigned by Default to: US Federal ECS CCDP Format

Federal Financials CTX Payment Validation

Code: FVTICTX

Contents:

1. Validate Agency Location Code:

Agency Location Code must be 8 characters.

Error Message: The Agency Location Code, captured in the bank account details, is not defined for bank account <bank account>.

2. Validate Routing Transit Number:

Routing Transit Number is stored in the bank branch number. It must be a nine digit numeric-only field. The ninth digit is the Check Digit, which is validated using the Modulus formula.

Error Message: Invalid routing number.

3. Validate Supplier Type:

Supplier Type of the Supplier must match the Supplier Type of the documents payable.

Error Message: The vendor for invoice <invoice number> cannot be of type Employee.

4. Validate Payee Social Security Number:

Social Security Number/Tax Identification Number is required.

Error Message: SSN/TIN must be supplied for supplier <supplier name>.

5. Validate Payee Bank Account Number:

Payee Bank Account Number is required.

Error Message: Bank account number missing for vendor.

6. Validate Bank Account Type:

Valid values for bank account type are CHECKING for Checking account and SAVINGS for Savings account.

Error Message: The Payee bank account must have a bank account type of either Checking or Savings.

7. Validate Schedule Number:

Schedule number is derived from the reference assigned by the administrator if specified on a payment instruction, or else the payment instruction ID becomes the

schedule number.

Error Message: The reference assigned by the administrator number must contain only valid characters of 0-9 or A-Z.

8. Validate Pay Alone Option:

Pay alone option is required on the invoice.

Error Message: Invoices for this payment format must have the Pay Alone flag checked on the invoice <invoice number>.

9. Validate Payment Amount:

Payment Amount must be less than 9,999,999.999 USD.

Error Message: Payment Amount exceeds the limit of \$9,999,999.99.

10. Validate RFC Identifier:

RFC Identifier is required.

Error Message: RFC Identifier not defined in the Banks Window for Bank <bank name>.

Validation Point: Payment Instruction

Validation Assigned by Default to: US Federal CTX Format AND US Federal CTX Consolidated Format

Federal Financials ECS CCD Payment Validation

Code: FVTPCCD

Contents:

1. Validate RFC Identifier:

RFC Identifier is required.

RFC Identifier not defined in the Banks Window for Bank <bank name>.

2. Validate Agency Location Code:

Agency Location Code must be 8 characters.

Error Message: The Agency Location Code, captured in the bank account details, is not defined for bank account <bank account>.

3. Validate Routing Transit Number:

Routing Transit Number is stored in the bank branch number. It must be a nine digit numeric-only field. The ninth digit is the Check Digit, which is validated using the Modulus formula.

Error Message: Invalid routing number.

4. Validate Agency Address:
At least one address line, city, state, or ZIP is required.
Error Message: Invalid address for Agency: <agency name>.
5. Validate Supplier Type:
Supplier Type of the Supplier must match the Supplier Type of the documents payable.
Error Message: The vendor for invoice <invoice number> cannot be of type Employee.
6. Validate Payee Social Security Number:
Social Security Number/Tax Identification Number is required.
Error Message: SSN/TIN must be supplied for supplier <supplier name>.
7. Validate Payee Bank Account Number:
Payee Bank Account Number is required.
Error Message: Bank account number missing for vendor.
8. Validate Bank Account Type:
Valid values for bank account type are CHECKING for Checking account and SAVINGS for Savings account.
Error Message: The Payee bank account must have a bank account type of either Checking or Savings.
9. Validate Schedule Number:
Schedule number is derived from the reference assigned by the administrator if specified on a payment instruction, or else the payment instruction ID becomes the schedule number.
Error Message: The reference assigned by the administrator number must contain only valid characters of 0-9, A-Z, or dash (-) and the first character must not be a zero.
10. Validate Pay Alone Option:
Pay alone option is required on the invoice.
Error Message: Invoices for this payment format must have the Pay Alone flag checked on the invoice <invoice number>.

Validation Point: Payment Instruction

Validation Assigned by Default to: US Federal ECS CCD Format

Federal Financials ECS PPD Payment Validation

Code: FVTPPPD

Contents:

1. Validate RFC Identifier:

RFC Identifier is required.

Error Message: RFC Identifier not defined in the Banks Window for Bank <bank name>.

2. Validate Agency Location Code:

Agency Location Code must be 8 characters.

Error Message: The Agency Location Code, captured in the bank account details, is not defined for bank account <bank account>.

3. Validate Routing Transit Number:

Routing Transit Number is stored in the bank branch number. It must be a nine digit numeric-only field. The ninth digit is the Check Digit, which is validated using the Modulus formula.

Error Message: Invalid routing number.

4. Validate Agency Address:

At least one address line, city, state, or ZIP is required.

Error Message: Invalid address for Agency <agency name>.

5. Validate Supplier Type:

Supplier Type of the Supplier must match the Supplier Type of the documents payable.

Error Message: The vendor for invoice <invoice number> must be of type Employee.

6. Validate Payee Social Security Number:

Social Security Number/Tax Identification Number is required.

Error Message: SSN/TIN must be supplied for supplier <supplier name>.

7. Validate Payee Bank Account Number:

Payee Bank Account Number is required.

Error Message: Bank account number missing for vendor.

8. Validate Bank Account Type:

Valid values for bank account type are CHECKING for Checking account and SAVINGS for Savings account.

Error Message: The Payee bank account must have a bank account type of either Checking or Savings.

9. Validate Schedule Number:

Schedule number is derived from the reference assigned by the administrator if specified on a payment instruction, or else the payment instruction ID becomes the schedule number.

Error Message: The reference assigned by the administrator number must contain only valid characters of 0-9, A-Z, or dash (-) and the first character must not be a zero.

10. Validate Pay Alone Option:

Pay alone option is required on the invoice.

Error Message: Invoices for this payment format must have the Pay Alone flag checked on the invoice <invoice number>.

11. Validate Reason Code:

Validate that a Federal payment reason is assigned to a payment.

Error Message: This payment format must have a Federal payment reason defined for each payment. The following are valid payment reasons: Allotments, SSA Benefits, VA Benefits, VAINS, Miscellaneous PPD, OPM Benefits, RRB Benefits, Salary, Travel, and Tax.

12. Validate Maximum Payment Amount:

A payment amount cannot exceed \$999,999.99 USD.

Error Message: Payment Amount exceeds the limit of \$999,999.99.

Validation Point: Payment Instruction

Validation Assigned by Default to: US Federal ECS PPD Format

Federal Financials ECS PPD+ Payment Validation

Code: FVTPPDP

Contents:

1. Validate RFC Identifier:

RFC Identifier is required.

Error Message: RFC Identifier not defined in the Banks Window for Bank <bank name>.

2. Validate Agency Location Code:

Agency Location Code must be 8 characters.

Error Message: The Agency Location Code, captured in the bank account details, is not defined for bank account <bank account>.

3. Validate Routing Transit Number:

Routing Transit Number is stored in the bank branch number. It must be a nine digit numeric-only field. The ninth digit is the Check Digit, which is validated using the Modulus formula.

Error Message: Invalid routing number.

4. Validate Agency Address:

At least one address line, city, state, or ZIP is required.

Error Message: Invalid address for Agency <agency name>.

5. Validate Supplier Type:

Supplier Type of the Supplier must match the Supplier Type of the documents payable.

Error Message: The vendor for invoice <invoice number> must be of type Employee.

6. Validate Payee Social Security Number:

Social Security Number/Tax Identification Number is required.

Error Message: SSN/TIN must be supplied for supplier <supplier name>.

7. Validate Payee Bank Account Number:

Payee Bank Account Number is required.

Error Message: Bank account number missing for vendor.

8. Validate Bank Account Type:

Valid values for bank account type are CHECKING for Checking account and SAVINGS for Savings account.

Error Message: The Payee bank account must have a bank account type of either Checking or Savings.

9. Validate Schedule Number:

Schedule number is derived from the reference assigned by the administrator if specified on a payment instruction, or else the payment instruction ID becomes the schedule number.

Error Message: The reference assigned by the administrator number must contain only valid characters of 0-9, A-Z, or dash (-) and the first character must not be a zero.

10. Validate Pay Alone Option:

Pay alone option is required on the invoice.

Error Message: Invoices for this payment format must have the Pay Alone flag checked on the invoice <invoice number>.

11. Validate Reason Code:

Validate that a Federal payment reason is assigned to a payment.

Error Message: This payment format must have a Federal payment reason defined for each payment. The following are valid payment reasons: Allotments, SSA Benefits, VA Benefits, VAINS, Miscellaneous PPD, OPM Benefits, RRB Benefits, Salary, Travel, and Tax.

12. Validate Maximum Payment Amount:

A payment amount cannot exceed \$999,999.99 USD.

Error Message: Payment Amount exceeds the limit of \$999,999.99.

Validation Point: Payment Instruction

Validation Assigned by Default to: US Federal ECS PPDP Format

Transmission Protocol Parameters

Transmission Protocol Parameters

The following sections provide detailed descriptions of the seeded transmission protocols and their parameters provided by Oracle Payments.

Http(s) POST Request

The Http(s) POST Request protocol delivers data and returns data and headers from the synchronous response, if any.

Http(s) POST Request Protocol

Parameter Code	Type	Mandatory	Description
HTTP_URL	String	Yes	The destination of the request, in the form of a HTTP URL.
USERNAME	String	No	Username if the URL requires HTTP authentication.
PASSWORD	String	No	Password if the URL requires HTTP authentication.
PROXY	String	No	Proxy host to use.

Parameter Code	Type	Mandatory	Description
NO_PROXY_DOMAIN	String	No	Network domain not requiring a proxy to contact. For example, if mycompany.com, then an URL such as http://host.mycompany.com will not use a proxy when contacted.
WALLET_LOC	String	No	Complete path to Oracle Wallet file containing server credentials; used if the target requires SSL client-authentication.
WALLET_PASSWD	String	No	Password of the Oracle Wallet file.
SEND_CONTENT_TYPE	String	Yes	MIME content type of the content sent to the destination URL.
RECEIVE_CONTENT_TYPE	String	Yes	MIME content type of the content returned by the URL.

Oracle Payments Tunneling

The Oracle Payments Tunneling protocol is used to delegate transmission (tunnel) to a servlet, which is typically located in a less sensitive portion of the network.

Oracle Payments Tunneling Protocol

Parameter Code	Type	Mandatory	Description
WEB_URL	String	Yes	URL of the transmission servlet.

Parameter Code	Type	Mandatory	Description
USERNAME	String	No	Username if the URL requires HTTP authentication.
PASSWORD	String	No	Password if the URL requires HTTP authentication.

Paymentech Online Specification 7.2 Socket

The Paymentech Online Specification 7.2 Socket protocol is an online socket (authorization) for the Paymentech 7.2 specification.

Paymentech Online Specification 7.2 Socket Protocol

Parameter Code	Type	Mandatory	Description
SOCKET_IP	String	Yes	IP/host name of the destination.
SOCKET_PORT	Number	Yes	Port number of the destination.
MAX_CONNECT_ATTEMPTS	Number	No	Maximum number of times to attempt reconnect if connection fails or is dropped.
CONNECT_RETRY_INTERVAL	Number	No	Number of seconds to wait before attempting reconnect.

Paymentech Batch Specification 2.1.0 Acknowledgement FTP Get

The Paymentech Batch Specification 2.1.0 Acknowledgement FTP Get protocol is the paymentech acknowledgement get protocol using FTP.

Paymenttech Batch Specification 2.1.0 Acknowledgement FTP Get Protocol

Parameter Code	Type	Mandatory	Description
HOST_IP	String	Yes	IP/host name of the destination.
HOST_PORT	Number	Yes	Port number of the destination.
USERNAME	String	No	FTP account user name.
PASSWORD	String	No	FTP account password.
LOCAL_DIR	String	Yes	Local directory where fetched file is deposited.
REMOTE_DIR	String	Yes	Location of acknowledgement on the remote system.

Paymenttech Batch Specification 2.1.0 Acknowledgement FTP Put

The Paymenttech Batch Specification 2.1.0 Acknowledgement FTP Put protocol is the paymenttech acknowledgement put protocol using FTP.

- Create a transmission configuration using the above protocol, ptk_settlement1
- Create a new FCPP for Paymenttech and use this new transmission configuration for settlement.
- A New Paymenttech_FCPP_CreditCard is created.
- Modify the Payee to use the new FCPP.

First Data North Authorization Specification 10/24/02 Socket

The First Data North Authorization Specification 10/24/02 Socket protocol is the online socket (authorization) for First Data North Authorization Specification 10/24/02.

First Data North Authorization Specification 10/24/02 Socket Protocol

Parameter Code	Type	Mandatory	Description
SOCKET_IP	String	Yes	IP/host name of the destination.
SOCKET_PORT	Number	Yes	Port number of the destination.
MAX_CONNECT_A TTEMPTS	Number	No	Maximum number of times to attempt reconnect if connection fails or is dropped.
CONNECT_RETRY_I NTERVAL	Number	No	Number of seconds to wait before attempting reconnect.

First Data North Magnetic Media Specification 2003.1 Settlement Batch FTP Put

The First Data North Magnetic Media Specification 2003.1 Settlement Batch FTP Put protocol is the First Data North batch file delivery protocol using FTP.

First Data North Magnetic Media Specification 2003.1 Settlement Batch FTP Put Protocol

Parameter Code	Type	Mandatory	Description
HOST_IP	String	Yes	IP/host name of the destination.
HOST_PORT	Number	Yes	Port number of the destination.
USERNAME	String	No	FTP account user name.
PASSWORD	String	No	FTP account password.

Parameter Code	Type	Mandatory	Description
REMOTE_DIR	String	Yes	Directory in which to deposit the batch file.
SUBMISSION_ADG	String	No	Submission file generation data group; provided by First Data.

First Data North Magnetic Media Specification 2003.1 Acknowledgment FTP Get

The First Data North Magnetic Media Specification 2003.1 Acknowledgment FTP Get protocol is the First Data North acknowledgement get protocol using FTP.

First Data North Magnetic Media Specification 2003.1 Acknowledgment FTP Get Protocol

Parameter Code	Type	Mandatory	Description
HOST_IP	String	Yes	IP/host name of the destination.
HOST_PORT	Number	Yes	Port number of the destination.
USERNAME	String	No	FTP account user name.
PASSWORD	String	No	FTP account password.
LOCAL_DIR	String	Yes	Local directory where fetched file is deposited.
REMOTE_DIR	String	Yes	Location of acknowledgement on the remote system.
ACK_ADG	String	No	Password of the Oracle Wallet file.

Parameter Code	Type	Mandatory	Description
SEND_CONTENT_TYPE	String	No	Acknowledgment generation data group; provided by First Data.

File Transfer Protocol for Static File Names

The File Transfer Protocol for Static File Names protocol is an FTP put protocol for static file names.

File Transfer Protocol for Static File Names Protocol

Parameter Code	Type	Mandatory	Description
HOST_IP	String	Yes	IP/host name of the destination.
HOST_PORT	Number	Yes	Port number of the destination.
USERNAME	String	No	FTP account user name.
PASSWORD	String	No	FTP account password.
REMOTE_DIR	String	Yes	Directory where to deposit the batch file.
FILE_NAME	String	Yes	Name of file created on the remote host. If not set, then attempts to use the concurrent request output file name.
MODE	String	No	FTP mode, using either of the following two values: PASSIVE or ACTIVE.

Parameter Code	Type	Mandatory	Description
KEEP_FILE	String	No	If Yes, then the file created in the local directory before delivery is not removed after transmission completes.

Secure File Transfer

The Secure File Transfer protocol is the secure FTP protocol for static file names.

Secure File Transfer Protocol

Parameter Code	Type	Mandatory	Description
HOST_IP	String	Yes	IP/host name of the destination.
HOST_PORT	Number	Yes	Port number of the destination.
USERNAME	String	No	FTP account user name.
PASSWORD	String	No	FTP account password.
PRIVATE_KEY_FILE	String	No	Complete path of user's private key file used for encryption. For information on using Oracle XML Publisher's Delivery Manager for this protocol, see <i>Oracle XML Publisher Administration and Developer's Guide</i> .

Parameter Code	Type	Mandatory	Description
PRIVATE_KEY_PASS WORD	String	No	<p>Password for the private key file.</p> <p>For information on using Oracle XML Publisher's Delivery Manager for this protocol, see <i>Oracle XML Publisher Administration and Developer's Guide</i>.</p>
REMOTE_DIR	String	Yes	Directory where to deposit the batch file.
FILE_NAME	String	Yes	Name of file created on the remote host. If not set, then attempts to use the concurrent request output file name.
MODE	String	No	FTP mode, using either of the following two values: PASSIVE or ACTIVE.
FILE_PERMISSION	String	No	<p>Permissions to be set on the newly created file on the remote server.</p> <p>For information on using Oracle XML Publisher's Delivery Manager for this protocol, see <i>Oracle XML Publisher Administration and Developer's Guide</i>.</p>

AS2 Send

The AS2 Send protocol is for delivering files. For information on using Oracle XML Publisher's Delivery Manager for this protocol's transmission parameter values, see

AS2 Send Protocol

Parameter Code	Type	Mandatory	Description
AS2_FROM	String	Yes	Name of the AS2 message sender.
AS2_TO	Number	Yes	Name of the AS2 message recipient.
AS2_SUBJECT	String	Yes	AS2 message subject.
FILE_NAME	String	No	Sent file name.
CONTENT_TYPE	String	Yes	MIME content type of delivered data.
HTTP_HOST	String	Yes	Host name of the receiving AS2 server.
HTTP_PORT	String	No	Port of the receiving AS2 server.
HTTP_REMOTE_DIR	String	Yes	Path portion of the URL for the AS2 receiving module, which does not include the file name portion.
HTTP_REMOTE_FILE	String	Yes	File portion of the URL for the AS2 receiving module; that is, what is left after the last directory in the URL.
HTTP_PROXY_HOST	String	No	Proxy to use when contacting the AS2 host.

Parameter Code	Type	Mandatory	Description
REQ_RECEIPT	String	No	Whether a delivery status receipt is required; Yes or No.
ENCRYPT_MSG	String	No	Whether to encrypt the AS2 message; Yes or No.
SIGN_MSG	String	No	Whether to digitally sign the message; Yes or No.

Local File System Delivery

The Local File System Delivery is a protocol for dumping files to a directory on the local system.

Local File System Delivery Protocol

Parameter Code	Type	Mandatory	Description
WALLET_FILE	String	No	Name of the AS2 message sender.
WALLET_PASSWORD	String	No	Password of the wallet file.
SERVER_CERT	String	No	Receiving server public key certificate file (full pathname).
ENCRYPTION_ALG	String	No	Encryption algorithm to use to encrypt message.
HASH_ALG	String	No	Hash/message digest algorithm to use when signing the message.

Risk Management

Using Risk Management

Oracle Payments supports risk management functionality. Electronic commerce applications can incorporate this feature and detect fraudulent payments. The following information describes how electronic commerce applications can utilize the risk management functionality of Oracle Payments.

Risk Factors and Risk Formulas

Oracle Payments is bundled with a set of risk factors. Payees can configure these factors depending on their business model. The payees can create multiple formulas using different factors and weights depending on their specific requirements. The ability to create multiple formulas provides flexibility to payees to accommodate different business scenarios. Each formula must be set up so that the sum of the weights is equal to 100. If a risk factor value is missing at the time of risk evaluation, the risk for the missing factor is considered very high in the formula.

Oracle Payments also defines an implicit formula for each payee with default factors and weights. Administrators have the flexibility to modify the implicit formula. The following information describes how and where the implicit formula is used.

Process Flow of Risk Evaluation

The following describes the process flow of risk evaluation.

1. To enable risk analysis during authorization, set up the explicit risk flag in the input transaction object or check the Enabled radio button in the Risk Management Status screen for that payee.
2. When an electronic commerce application makes a Payment Request API call, Oracle Payments first checks the risk flag and depending on its value, decides if the payee involved in the payment request is risk enabled or not. If the risk analysis

field indicates that Oracle Payments should perform risk analysis, or if a default value is added in the field and a payee is risk enabled, Oracle Payments evaluates either the risk formula passed in the Payment Request API or the implicit formula associated with that payee.

3. Electronic commerce application can pass a specific risk formula name by calling the overloaded Payment Request API. This API takes PmtRiskInfo object in which electronic commerce application can set up the formula name and additional information. If PmtRiskInfo object is not passed and the payee is risk enabled, Oracle Payments evaluates the implicit formula of that payee.
4. Oracle Payments returns the Risk Response (RiskResp) object as part of the payment response. If risk evaluation is done successfully, Risk Response object contains the risk score obtained after evaluation and the threshold value that is set up with the payee. Based on the risk score and the threshold value, the electronic commerce application can decide whether a payment can be accepted or not.
5. If the risk score is more than the threshold value, the payment request is risky.

Process Flow of Independent Risk APIs

Risk API 1

1. When an electronic commerce application invokes Risk API 1, Oracle Payments evaluates the risk formula sent in the request or the implicit formula associated with that payee.
2. Oracle Payments evaluates all the risk factors that are configured as part of this formula, except the AVS Code risk factor.
3. After evaluation, Oracle Payments returns Risk response (RiskResp) object as a response to this API. This response object contains, the status of the API call, AVSCodeFlag indicating if AVS Code risk factor was part of the formula or not, risk score, and the risk threshold value that is setup for the payee. Depending on the AVSCodeFlag value, it is decided whether to call Risk API 2 or not.

Note: Partial risk score is returned if AVS Code risk factor is part of the risk formula.

Risk API 2

1. Electronic commerce applications need to call this API with the same PayeeID and formula name that were used to call Risk API 1. The risk score that was returned as part of the Risk API 1 response also needs to be sent. When electronic commerce applications call this API, Oracle Payments checks again if the formula has AVS Code risk factor configured in it or not. If it is configured, Oracle Payments

evaluates the AVS Code risk factor.

2. After evaluating the AVS Code risk factor, Oracle Payments calculates the final risk score of the formula using the previous risk score that was sent and the AVS Code risk factor score. This risk score is sent back to the electronic commerce application as part of the response object of this API.

Risk Management Test Scenarios

The following are three business scenarios that describe how a merchant can use the Risk Management functionality.

Merchant Selling Books and Low Priced Goods

In a small business, accepting risky instruments is a critical factor. If a customer is using a stolen credit card, the merchant should consider this transaction risky and assign this risk factor a higher weight than the other risk factors. Ship to/bill to address matching and payment history are also important risk factors. To include AVS Code risk factor, a merchant can set up a formula with weights as shown in Weight B column in the Risk Formula Setup-First Case table. The total of all the weights should be 100. For a formula that a merchant would set up in this case, see Risk Formula Setup for the First Case.

Risk Formula Setup for the First Case

The table below shows the risk formula setup for a merchant selling books and low priced goods.

Factors	Weight A	Weight B
Risky Instruments	30	30
Payment Amount Limit	15	15
Transaction Amount	15	15
Ship to/Bill to	20	10
Payment History	20	10
AVS Code	0	20

Risk Factor Setup

- Payment Amount Limit

The table below shows the risk levels and the associated payment amounts.

Risk Levels	Greater than or Equal To
Low	0
Low medium	100
Medium	200
Medium high	300
High	400

- Transaction Amount

A transaction is high risk if the transaction amount exceeds 500 in one week.
Otherwise there is no risk.

- Payment History

The table below shows the risk levels and the number of payments made in the last six months by a particular customer.

Risk Levels	Greater than or Equal To
Low	6
Low medium	4
Medium	3
Medium high	2
High	0

- AVS Code

The table below shows the risk levels and the associated AVS Codes. AVS Code risk

factor evaluation is useful only for customers in the United States.

Risk Level	AVS Code
No risk	S,Y,U,X,R,E
Low	A,Z,W
Low medium	
Medium	
Medium high	
High	N

- Ship To/bill To and Risky Instruments

These risk factors do not require any setup. The evaluation will be done with the data already existing in the database.

- Risk Score

A typical threshold value would be between medium and medium high risk score. Risk Management module evaluates the payment request and returns an overall risk score. If an overall risk score exceeds the threshold value set up by the merchant, then the merchant has to decide whether to process the request or to block the request.

Merchant Selling Electronic Goods

Risky instruments is a critical factor in this case. If a customer is using a stolen credit card, the merchant should consider this transaction risky and assign it a higher weight.

Frequency of purchase is the next important risk factor. Usually customers do not buy electronic goods frequently, and if they do, the purchases could be a fraudulent.

In this scenario, time of purchase is also should be considered as an important risk factor. If someone buys many goods after 2:00 AM, it might be a fraudulent purchase.

To include an AVS Code risk factor, a merchant can sets up a formula with weights as shown in column Weight B in Risk Formula Setup-Second Case table. The total of all the weights are 100. The AVS Code risk factor evaluation will be useful only for customers in the United States.

Risk Formula Setup for the Second Case

The table below shows the risk formula set up for a merchant selling electronic goods.

Factor	Weight A	Weight B
Risky Instruments	30	30
Ship to/Bill to	15	12
Time of Purchase	15	12
Frequency of Purchase	20	10
Payment Amount	10	8
Transaction Amount Limit	10	8
AVS Code	0	20

Risk Factor Setup

- Payment Amount Limit

The table below shows the risk levels and the associated payment amounts.

Risk Levels	Greater Than or Equal To
Low	500
Low medium	1000
Medium	1500
Medium high	2000
High	2500

- Transaction Amount

This risk factor is considered high risk if the amount exceeds 2,500 in one week. Otherwise there is no risk.

- Frequency of Purchase

This risk factor is considered high risk if the frequency of purchase exceeds ten times in the previous one week.

- AVS Codes

The table below shows the risk levels and the associated AVS codes. AVS codes risk factor evaluation is only useful for customers in the United States.

Risk Level	AVS Code (Comma Separated)
No risk	S, Y, U, X, R, E
Low	A,Z,W
Low medium	
Medium	
Medium high	
High	N

- Ship To/Bill To and Risky Instruments

These risk factors do not require any setup. The evaluation is done through the data already existing in the database.

- Risk Score

A typical threshold value is to be between medium and medium high risk score.

The risk management module evaluates the payment request and returns an overall risk score. If an overall risk score exceeds the threshold value set up by the merchant, the merchant has to decide whether to process the request or to block the request.

Business to Business Customer

In a business to business scenario, a merchant has an established relationship with his customer. In this scenario, the Oracle Receivables risk factors take higher precedence. The merchant is interested in the customer's payment history, his credit rating, etc. All Oracle Receivables risk factors are set up through Oracle Receivables interface.

Risk Formula Setup in the Third Case

The table below shows a Risk Formula setup for a business to business customer.

Factors	Weight
Overall Credit Limit	30
Transaction Credit Limit	30
Risk Codes	15
Credit Rating Codes	15
Payment History	10

Risk Factor Setup

- Overall Credit Limit: 100,000
- Transaction Credit Limit: 50,000
- Risk Codes are set up through Oracle Receivables codes.

The table below shows the risk codes and the associated risk scores set up through Oracle Payments administration user interface.

Risk Codes	Risk Score
Low	Low
Average	Medium
Excellent	No risk

- Credit Rating Codes are set up through Oracle Receivables interface

The table below shows the set up of credit rating codes and the associated risk scores.

Credit Rating Codes	Risk Score
Low	Low
Average	Medium
Poor	High
Excellent	No risk

- Risk Score

A typical threshold value is between medium and medium high.

Risk management module evaluates the payment request and returns an overall risk score. If an overall risk score exceeds the threshold value set up by the merchant, then the merchant decides whether to process or block the request.

Funds Capture Error Codes

Error Handling During Payment Processing

Oracle Payments returns a response object to each API that an electronic commerce application calls. If the operation fails, then the response object contains status value (IBY_FAILURE), indicating that there was a failure while processing the request. In these cases, the electronic commerce application can get more information about the failure by checking the error code and the error message. Errors can happen in Oracle Payments for various reasons. For example, wrong or duplicate data passed by the electronic commerce application, time out while communicating with Payment Systems, etc. All the errors that can occur in Oracle Payments can be categorized in these groups:

- Common Errors, page E-1
- Errors Due to Invalid or Duplicate Data, page E-2
- Communication Errors, page E-3
- Configuration Errors, page E-3

Common Errors

The table below describes the most common errors.

Error Code	Description
IBY_0001	Communications error. The payment system, the processor, or Oracle Payments electronic commerce servlet is not accessible. You should resubmit the request at a later time.

Error Code	Description
IBY_0002	Duplicate order identifier.
IBY_0003	Duplicate batch identifier.
IBY_0004	Mandatory fields are required.
IBY_0005	Payment system specific error. Check BEPErrCode and BEPErrMsg in response objects for more information.
IBY_0006	Batch partially succeeded. Some transactions in the batch failed and some were processed correctly.
IBY_0007	The batch failed. You should correct the problem and resubmit the batch.
IBY_0008	Requested action is not supported by the payment system.
IBY_0017	Insufficient funds.
IBY_0019	Invalid credit card or bank account number.

Errors Due to Invalid or Duplicate Data

In each payment request, a payment instrument from which the money is transferred to the payee's account is involved. Generally this information is given by the end user of the electronic commerce application. Sometimes the end user might enter wrong instrument number or an instrument number that does not have enough funds. To detect these errors, Oracle Payments provides two error codes that help electronic commerce applications to prompt the end user for correct information.

The error codes due to invalid or duplicate data and their descriptions are given in the table below.

Error Code	Description
IBY_0017	Insufficient funds

Error Code	Description
IBY_0019	Invalid credit card/bank account number

Communication Errors

Since payment processing requests involve a number of different components connected over networks, time-out errors or communication errors are possible. For example, a processor successfully processes a payment request, but the network connection between the payment system and Oracle Payments, or the network connection between Oracle Payment's PL/SQL API package and Oracle Payments electronic commerce servlet break down, causing the electronic commerce application not to receive the result. In some cases, electronic commerce application might crash before receiving a response. Before the crash, payment processing may have completed. Therefore, when electronic commerce application calls the API with the same information, Oracle Payments considers this a duplicate request and raises an error. To recover from such errors, Oracle Payments provides two approaches.

In the first approach, which is applicable to OraPmtReq and OraPmtCredit, the electronic commerce application can try the request with the retry flag set up to TRUE. This makes Oracle Payments retry the request if it has not processed the request. Otherwise Oracle Payments sends the same response that was sent when this request was first made.

In the second approach, which is applicable to all other operations except OraPmtReq and OraPmtCredit, the electronic commerce application needs to find out if the transactions went through successfully to re-execute any lost transactions. To enable the merchant or business to query the status of a transaction, you need to integrate the Query Transaction Status API in the electronic commerce application. This API returns all existing records for a particular transaction identifier on a payment system.

The table below describes the communication error code and its description.

Error Code	Description
IBY_0001	The payment system, the processor, or Oracle Payment's electronic commerce servlet is not accessible. You should resubmit the request at a later time.

Configuration Errors

These errors occur if payees or payment systems are not configured properly. Make

sure that the URLs are entered correctly and the payee's payment system identifiers are configured properly.

Integrating with Oracle Payments Using Public Funds Capture APIs

Overview of Oracle Payments APIs

This chapter explains the public APIs used in Oracle Payments.

Oracle Payments provides APIs which can be implemented:

- Implementing Electronic Commerce Applications APIs, page F-1. These APIs are mainly used for payment processing.

Implementing Electronic Commerce Applications APIs

Oracle Payments provides various types of APIs to integrate electronic commerce applications with Oracle Payments.

If you are using an electronic commerce application other than the preintegrated Oracle applications, you must implement the electronic commerce application's APIs to link your application to Oracle Payments.

Electronic commerce applications can embed the Oracle Payments functionality within their application, which eliminates the need to access Oracle Payments as a standalone application, and hence improves performance, and simplifies the setup.

This section describes the various APIs that are provided to the electronic commerce applications for using the features of Oracle Payments. The APIs were categorized into these categories:

- Payment Instrument Registration APIs, page F-2
- Payment Processing APIs, page F-3
- Risk Management APIs, page F-6

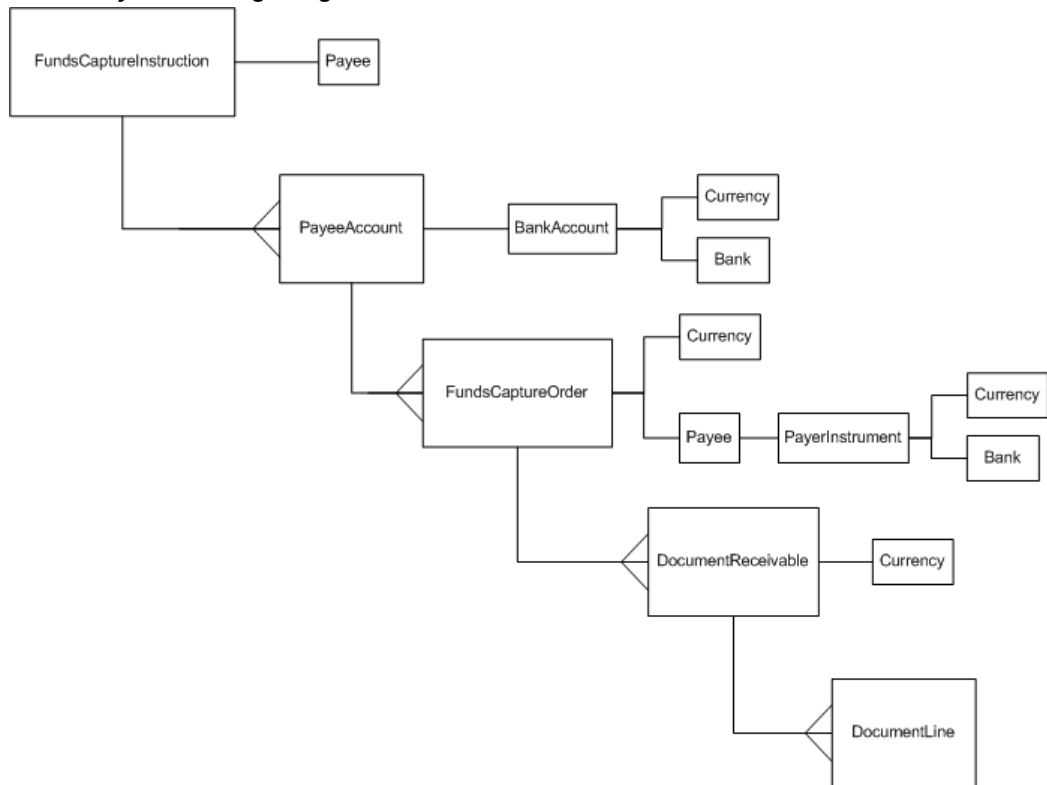
- Credit Card Validation APIs, page F-7
- Status Update API, page F-9

Oracle Payments provides APIs in these programming languages:

- Java APIs for Electronic Commerce Application, page F-12
- PL/SQL APIs for Electronic Commerce Applications, page F-21

This diagram shows the integration of APIs with Oracle Payments.

Oracle Payments Integrating with APIs



Payment Instrument Registration APIs

Payment Instrument APIs provide the functionality to register a payor's bank, credit card, PINless debit card, or purchase card.

OrainstrAdd

This API is provided to register a user's bank, credit card, PINless debit card, or purchase card account information with Oracle Payments. Oracle Payments generates a

PmtInstId if this registration is successful. This identifier is used for payment transactions or for deleting, modifying, or inquiring about this account. Instrument number (credit card number, PINless debit card, purchase card number, or bank account number) and payor identifier together have to be unique.

OralInstrMod

This API is provided to modify registered payment instrument account information with Oracle Payments.

OralInstrInq

There are two inquiry APIs. One queries instrument information for a single given instrument. The other queries all registered payment instruments for a given payor. The result may contain a mix of credit cards, PINless debit cards, purchase cards, or bank accounts.

Payment Processing APIs

These APIs are the transactional APIs that support various payment operations. The electronic commerce applications use these APIs to process various transaction types. For example, authorization of credit cards, PINless debit cards, and purchase cards, transfer of funds from one bank account to another, capture, cancel, return, and others. A list of such APIs are provided below.

OraPmtReq

This API supports authorization and authorization with capture for credit card, PINless debit card, and purchase card payments. This API also supports inbound account transfers and electronic funds transfer online validation.

When an electronic commerce application is ready to invoke a payment request (possibly due to a user action), it calls this API. If the operation is successful, a transaction identifier is generated by Oracle Payments and is returned as part of the result. This transaction identifier can be used later by the electronic commerce application to initiate any other operation on a payment.

For example, to modify a payment or capture a payment, the electronic commerce application sends this identifier with other information that is needed to perform the operation requested.

If a payment is either a credit card, PINless debit card, or a purchase card payment, and the request is online, Oracle Payments can perform risk analysis with the payment request (Authorization).

To enable risk analysis with authorization, either setup the payment request with risk flag set to true in one of its input objects (Refer to Java Documentation for details) or check the Enabled radio button in the Risk Management Status screen for that payee. If

either of the conditions are satisfied, the electronic commerce application will check the Riskresp object that is returned as part of the payment response object to the Payment Request API. Electronic commerce applications can also invoke the Payment Request API to evaluate a specific formula by passing the PaymentRiskInfo object.

This API is also used after a voice authorization is done to enable Oracle Payments to handle follow-on operations. To use it for a voice authorization, set up the payment request's input objects with the Voice Authorization flag set to true and the Authorization Code variable set to the authorization code issued by the financial institution. See *Oracle Payments Java Documentation* for details.

OraPmtCanc

A scheduled payment can be canceled by an electronic commerce application using this API.

OraPmtQryTrxn

This API provides interface for inquiring the status or history of a payment to electronic commerce application. If a payment has been scheduled and the payment system supports an inquiry operation, the latest status is obtained from the payment system. Otherwise it sends the latest status of the payment as it is in Oracle Payments. History of a payment can also be obtained.

OraPmtCapture

When a credit card or purchase card is used as part of a payment request and only an authorization is requested, the electronic commerce application has to capture the payment at a later time. The following APIs allow the electronic commerce application to capture all such payments.

OraPmtReturn

This API is used for credit card, PINless debit card, and purchase card specific operations. It allows processing returns from the payor.

Gateway model payment systems process capture operations online. If the capture is still in the Gateway's open batch (that is, the batch has not been closed) you should call OraPmtVoid. If the batch has been closed, you need to call OraPmtReturn. The batch needs to be closed again before the return is processed. This can be confusing since Gateways can be set up to close batches automatically, for example, once per day.

Processor model payment systems process captures offline. If the capture is still in Payment's open batch, call OraPmtVoid. If the batch has been closed, call OraPmtReturn. The batch needs to be closed again after the return operation for the return to be processed.

OraPmtInq

This API retrieves the payment related information that was sent at the time of a payment request (OraPmtReq API). This information includes payment instrument, payee, tangible id (bill or order), and payor. If the electronic commerce application does not store the payment information, then this is a useful API to support modification of payment requests. It can retrieve the payment information and display it to the end user for modification.

OraPmtVoid

This API allows electronic commerce application to void operations submitted earlier. OraPmtVoid API is supported only to void certain credit card, and purchase card operations. Oracle Payments supports both online and offline OraPmtVoid API calls.

Voiding auths electronically is not supported by some processors or gateways. Only a few card-issuing banks supported it while the vast majority did not. Cancelling an authorization could only be done by telephone or by letting the auth expire.

Thus, within Payments, calling OraPmtVoid for an Online Auth results in the current payment system servlets returning status 8 - Operation not Supported. For an Offline Auth, you can void the Authorization if it is still in the Payments open batch and has not yet been sent to the payment system.

OraPmtCredit

This API provides credit and Electronic Fund Transfer (EFT) operations. Electronic commerce applications can use this API to give stand-alone credit to the customer. If the operation is successful, a transaction identifier is generated by Oracle Payments. This Identifier is used later to initiate any other operation on the payment. For example, to cancel the credit, electronic commerce application sends this identifier with other information that is needed to perform the cancellation.

OraPmtCloseBatch

The Close Batch API allows a payee or an electronic commerce application to close a batch of previously performed credit card, or purchase card transactions and if necessary PINless debit card. The transaction types that are included in a batch are: capture, return, and credit. This operation is mandatory for a terminal-based merchant.

A host-based merchant may not have to explicitly close the batch because the batch is generally closed at predetermined intervals automatically by the processor. An electronic commerce application has to get this information from its merchant's acquirer.

OraPmtQueryBatch

This API provides an interface to the electronic commerce application to query the status of an existing batch and a closed batch.

Risk Management APIs

These APIs allow electronic commerce applications to do risk analysis independently for credit card, PINless debit card, and purchase card transactions. These APIs together can evaluate any risk formula that is configured for a payee.

A risk formula can contain any number of risk factors with different weights associated with them. When Risk API 1 is called, it evaluates all the factors configured in the formula except the AVS Code risk factor. If a risk formula has an AVS Code risk factor, then, Risk API 1, in the response object, indicates that the formula has an AVS Code risk factor. This allows electronic commerce applications to completely or partially check the risk formula and decide whether to perform an authorization or not.

If the response of the first Risk API 1 indicates that the payment is not risky, then electronic commerce application can perform the authorization and complete the rest of the evaluation by calling Risk API 2.

Electronic commerce applications can call Risk API 2 by passing the same payee id, the formula name, and the AVS code that was returned during the authorization response and the risk score that was returned as part of the response in Risk API 1. The response object of Risk API 2 contains the finally evaluated risk score.

Risk API 1

This API evaluates the risk formula associated with the payee id passed as part of the input object, PmtRiskInfo. This API can evaluate a specific formula or the implicit formula depending on the input object. After evaluation, this API constructs the response object indicating if the AVS Code risk factor is a part of the formula or not by setting the flag, AVSCodeFlag. If this flag is set to true, then electronic commerce applications need to call the Risk API 2 to complete the risk evaluation of the formula.

Risk API 2

This API needs to be called when the AVSCodeFlag in RiskAPI 1 response object indicates that the formula contains AVS Code factor. When this API is called, it only evaluates the AVS code factor. The input object of this API contains the same payee id and the formula name that was passed in Risk API 1 and the AVS Code that was returned by the payment system for the payment request. The response object that this API returns, contains the final risk score of the formula.

Credit Card Validation APIs

The Credit Card Validation APIs provide methods for determining the credit card type of a credit card number and for doing basic authentication. Since most credit card types specify the number of digits and a prefix for all valid credit card accounts in their company name, it is possible to determine the credit card types of most credit card numbers. Also, since the digits of most credit card types must (using a special algorithm) be evenly divisible by 10, it is possible to determine if a credit card number is valid or not. These APIs do not perform some of the more advanced credit card verification techniques available to back end payment systems, such as billing address verification. These APIs allow many common errors to be caught, such as wrongly typed or truncated credit card digits. By allowing common errors to be caught by the electronic commerce application, performance is improved, since the cost of calling these APIs is much less than sending a request to the back end payment system.

The Credit Card Validation APIs are created as part of the IBY_CC_VALIDATE package and this package is installed in the APPS schema.

Main Methods of Credit Card Validation APIs

The Credit Card Validation APIs consist of three main methods.

1. Method StripCC is used to format a raw credit card number input by the customer. StripCC removes common filler characters such as hyphens and spaces until it produces a credit card number consisting only of digits. StripCC must be called before the credit card number is passed to the other methods.
2. Method GetCCType returns the credit card type of a credit card number, where each credit card type, including values for invalid and unknown types is a constant in the package.
3. Method ValidateCC, which takes both a credit card number and date. It returns a boolean value indicating whether the credit card can still be used or not.

Note: The IN parameters p_api_version and p_init_msg_list and the OUT parameters x_msg_count and x_msg_data are ignored. If an unexpected error occurs, x_return_status will be set to FND_API.G_RET_STS_UNEXP_ERROR. This will happen if the credit card number has invalid characters in it.

```
DECLARE
-- each character specifies a possible filler
characters in the credit
-- card number; i.e. a character that can safely be
stripped away
p_fill_chars VARCHAR(3) := '* -#';
```

```

p_cc_number VARCHAR(20) := '4111*1111 1111-1111#';
p_api_version NUMBER := 1.0;
p_init_msg_list VARCHAR2(2000) := ' ';
x_return_status VARCHAR2(2000);
x_msg_count NUMBER;
x_msg_data VARCHAR2(2000);
-- will hold the credit card number stripped of all
characters except
-- digits; credit card numbers must be of this form for
the GetCCType
-- and ValidateCC methods
v_clean_cc VARCHAR(20);
-- variable to be set by GetCCType method
v_cc_type IBY_CC_VALIDATE.CCType;
-- variable set by ValidateCC method; indicates if the
credit card is
-- still usable
v_cc_valid BOOLEAN;
-- credit card expr date; rolled to the end of the
month
-- by the ValidateCC method
v_expr_date DATE := SYSDATE();
BEGIN
-- the credit card number must first be stripped of all
non-digits!!
IBY_CC_VALIDATE.StripCC( p_api_version,
p_init_msg_list, p_cc_number,
p_fill_chars, x_return_status, x_msg_count, x_msg_data,
v_clean_cc );
-- check that illegal characters were not found
IF x_return_status != FND_API.G_RET_STS_UNEXP_ERROR
THEN
    IBY_CC_VALIDATE.GetCCType( p_api_version,
p_init_msg_list, v_clean_cc,
x_return_status, x_msg_count, x_msg_data, v_cc_type);
    IF x_return_status != FND_API.G_RET_STS_UNEXP_ERROR
    THEN
        IF v_cc_type=IBY_CC_VALIDATE.c_InvalidCC THEN
DBMS_OUTPUT.PUT_LINE('Credit card number not a valid
one. ');
        ELSE
DBMS_OUTPUT.PUT_LINE('Credit card number OK. ');
        END IF;
    END IF;

```



```

        IBY_CC_VALIDATE.ValidateCC( p_api_version,
        p_init_msg_list, v_clean_cc,
        v_expr_date, x_return_status, x_msg_count, x_msg_data,
        v_cc_valid);

        IF v_cc_valid THEN

DBMS_OUTPUT.PUT_LINE('Credit card is valid.');
```

```

        ELSE

DBMS_OUTPUT.PUT_LINE('Credit card number invalid or has
expired.');
```

```

        END IF;
END IF;
END;
```

Note: An overloaded version of the StripCC method exists. It takes all the same arguments as the version used above except p_fill_chars. It gets its filler characters from the package constant c_FillerChars, which allows spaces and hyphens to be interspersed within the credit card number.

Status Update API

Oracle Payments has defined a PL/SQL API that must be implemented by electronic commerce applications when offline payment processing is performed. This API allows the electronic commerce application to receive a status update. This API must be defined in a package. The naming convention of the package and signature of the API are defined below. Electronic commerce applications must implement the package according to the syntax defined and create the package in the APPS schema if they have offline payments.

The package name has to be of the format <application_short_name>_ecapp_pkg. The application_short_name is a three-letter short name that was given in electronic commerce application registration. The package should have defined update_status procedure with the following signature:

```

PROCEDURE UPDATE_STATUS(
totalRows                IN          NUMBER,
txn_id_Tab               IN          APPS.JTF_VARCHAR2_TABLE_100,
req_type_Tab             IN          APPS.JTF_VARCHAR2_TABLE_100,
Status_Tab               IN          APPS.JTF_NUMBER_TABLE,
updatedt_Tab             IN          APPS.JTF_DATE_TABLE,
refcode_Tab              IN          APPS.JTF_VARCHAR2_TABLE_100,
o_status                 OUT         VARCHAR2,
o_errcode                OUT         VARCHAR2,
```

```
o_errmsg          OUT          VARCHAR2 ,
o_statusindiv_Tab IN OUT APPS.JTF_VARCHAR2_TABLE_100);
```

The following list describes the field names in the above signature:

1. totalRows: total number of rows being passed for the update.
2. txn_id_Tab: table of transaction identifiers for which the update is sent.
3. req_type_Tab: table of request types corresponding to the Transaction Identifier. For each transaction, there might be a req_type associated with it and the electronic commerce application has to update the correct transaction, based on txn_id and req_type. The reason for having a req-type is to uniquely identify the transaction. For the same transaction identifiers, there can be multiple transactions. e.g. Authorization and Capture. Electronic commerce applications can uniquely identify the transaction based on the values in txnid and req_type.

The table below lists the various kinds of request types and their descriptions.

req_type	Description
ORAPMTCAPTURE	Capture transaction
ORAPMTCREDIT	Credit transaction
ORAPMTREQ	Authorize transaction
ORAPMTRETURN	Return transaction
ORAPMTVOID	Void transaction

4. Status_Tab: table of statuses corresponding to each transaction.

The table below lists the various values and their statuses.

Value	Status
0	Paid
5	Payment failed
13	Scheduled
15	Failed

Value	Status
17	Unpaid
18	Submitted

Note: Please refer to Table D-15 for a complete list of values and their statuses.

5. updatedt_Tab: table for the last update date for each transaction.
6. refcode_Tab: table for the reference code for each transaction.
7. o_status: the overall status of the procedure. If there are errors in trying to execute the procedure, electronic commerce application should set up an appropriate value in this field.
8. o_errcode: the error code for any errors which might have occurred during processing.
9. o_errmsg: the error message for the error.
10. o_statusindiv_Tab: table of status values which have been updated. If the status value has been updated by the electronic commerce application for a particular transaction, it should set the value to TRUE for that transaction, otherwise, it should set the value to FALSE.

Note: In the above procedure, for each transaction there will be an entry in the table parameters. If there were ten transactions of this electronic commerce application, whose status has changed, there will be ten entries in each table parameters.

When Does the Scheduler Invoke the API?

The Scheduler picks up all the offline payment transactions to be scheduled every time it is run. After all the offline payment transactions are processed either successfully or unsuccessfully, the Scheduler has to update the status changes, if any, of each transaction, to the appropriate electronic commerce application. To update the electronic commerce application, the Scheduler calls the PL/SQL API, which is implemented by that electronic commerce application.

Pseudo Code for Implementing the PL/SQL API by Electronic Commerce Application

For each row update, the status is based on the request type and the transaction identifier. If the update is successful, then set up the status value appropriately.

```
for i in 1..totalRows
;update the tables with status, updatedate, and refinfo information
update tables using status_Tab[i], updatedt_Tab[i], refCode_Tab[i] for
    the transaction with id txn_id_Tab[i] and req_type_tab[i]
if update is successful
    o_statusindiv_Tab[i] := 'TRUE'
else
    o_statusindiv_Tab[i] := 'FALSE'
end for;
return
```

Java APIs for Electronic Commerce Application

All administration and inbound payment processing functionalities are provided via the Java PaymentService interface. The following information describes how to access and use Java APIs. Refer to Oracle Payments JavaDoc for more details.

Note: Guest user properties need to be setup in the database before any operation can be performed. Please refer to the Setup Document provided by CRM Foundation for more details.

Obtaining/Releasing the Payment Service Handle

The OraPmt class offers convenient ways to obtain Payment Service handle (PaymentService) for the user. The application can call various APIs using this handle.

- To obtain the payment service handle, use the following method:

```
static public PaymentService init() throws PSException
```

This API provides Payment Service handle to the user and takes care of all the necessary session initialization steps.

- To release a Payment Service handle with the session, use the following method:

```
static public void end() throws PSException
```

Sample Code

The following code gives an example of how these APIs are used.

```

public static void main(String[] args) {

    try {

        PaymentService paymentService = OraPmt.init();

        // now you can call all kinds of APIs

        //PSResult result = paymentService.OraPmtReq(...);

    } catch (PSException pe) {

        // exception handling

        System.out.println("Error code is: " + pe.getCode());

        System.out.println("Error message is: " + pe.getMessage());

    }

    finally {

        try {

            OraPmt.end();

        } catch (PSException pe) {

            // exception handling

            System.out.println("Error code is: " + pe.
getCode());

            System.out.println("Error message is: " +

                                pe.getMessage());

        }

    }

}

```

Checking Returned Result from Payment Service API

PSResult is the returned object of all PaymentService APIs. To obtain the status of the operation, use the following API:

```
public String getStatus();
```

This API returns one of the following constants:

```
PSResult.IBY_SUCCESS// action succeeded
```

```
PSResult.IBY_WARNING// action succeeded with warning
```

```
PSResult.IBY_INFO// not yet in use
```

```
PSResult.IBY_FAILURE// action failed
```

If SUCCESS or WARNING is invoked, a result object can always be obtained by using the following API:

```
public Object getResult();
```

If FAILURE is invoked, a result object may be returned for payment operation APIs, if this failure occurred with back- end payment system.

The actual object returned varies with each API. It could be an integer or one of the payment response objects. You need to clearly cast it. For a list of castings, refer to the Oracle Payments Java Documentation for the PaymentService interface.

If WARNING or FAILURE is invoked, a warning or error message is returned. Use the following two APIs to retrieve error codes and error messages.

```
public String getCode();// get the error code 'IBY_XXXXXX'
```

```
public String getMessage(); // get the error message text
```

The following sample code illustrates the behavior of PSResult object.

```
public Object checkResult(PSResult pr) {

    String status = pr.getStatus();

    if (status.equals(PSResult.IBY_FAILURE)) {

        // in case of failure, only error message is expected

        System.out.println("error code is : " + pr.getCode());

        System.out.println("error message is : " + pr.getMessage());

        Object res=pr.getResult();

        if (res!=null) System.out.printIn ("failure occurred with
backend Payment system");

        return res;

    }

}
```

```

        if (status.equals(PSTResult.IBY_SUCCESS)) {

            // in case of success, only result object is expected

            Object res = pr.getResult();

            return res; // you need cast this to specific object

            // based on the APIs you called

        }

        if (status.equals(PSTResult.IBY_WARNING)) {

            // in case of warning, both result object and message are

            // expected

            // warning is returned only for Payment APIs in case of

            // offline scheduling

            System.out.println("warning code is : " + pr.getCode());

            System.out.println("warning message is : " + pr.getMessage());

            Object res = pr.getResult();

            return res; // you need cast it here too

        }

        // currently IBY_INFO is not yet returned by any PaymentService API

        System.out.println("Illegal status VALUE in PSTResult! " +

            pr.getStatus());

        return null;

    }

```

Using Payment Service API

After a payment service handle is obtained via the `OraPmt` class, you can call any of the following APIs in Payment Service interface. For details, refer to JavaDoc.

Here is some sample codes for the Payment Instrument API, and Payment Processing APIs. These codes use the `checkResult` call.

Registering a Credit Card

```
public void instrAPISample(PaymentService paymentService,

                           int ecappId)    {

    PSResult pr;

    Object obj;

    CreditCard cc;

    Address addr;

    int instrid_cc;

    String payerid = "payer1";

    addr = new Address("Line1", "Line2", "Line3", "Redwood Shores",

                      "San Mateo", "CA", "US", "94065");

    // credit card

    cc = new CreditCard();

    cc.setName("My Credit Card");

    cc.setFName("Paymentech");

    cc.setInstrBuf("This is my credit card description.");

    cc.setInstrNum("4111111111111111"); // the credit card number

    cc.setCardType(Constants.CCTYPE_VISA); // the credit card type,
should

    // match the credit card number, if set

    cc.setExpDate(new java.sql.Date(101, 0, 10)); // Jan 10, 2001

    cc.setHolderName("Mary Smith");

    cc.setHolderAddress(addr);

    // add the credit card

    pr = paymentService.oraInstrAdd(ecappId, payerid, cc);
```



```

obj = checkResult(pr);

if (obj == null) return; // registration failure

instrid_cc = ((Integer) obj).intValue();

System.out.println("Credit card registered successfully " +

                    "with instrument id " + instrid_cc);

}

```

Sending a Credit Card Authorization Request

```

// perform an ONLINE credit card authorization with payment service

public void paymentAPISample(PaymentService paymentService, int ecAppId)
{

    Bill t;

    CoreCreditCardReq reqTrxn;

    CreditCard cc;

    PSResult pr;

    CoreCreditCardAuthResp resp;

    // set up the tangible object

    t = new Bill();

    t.setId("orderId1");

    t.setAmount(new Double(21.00));

    t.setCurrency("USD");

    t.setRefInfo("refInfo");

    t.setMemo("memo");

    t.setUserAccount("userAcct");

    // set up the transaction object

    reqTrxn = new CoreCreditCardReq();

```

```

reqTrxn.setNLSLang("American_America.US7ASCII");

reqTrxn.setMode(Transaction.ONLINE);

reqTrxn.setSchedDate(new java.sql.Date(100, 5, 10)); //June 10,
2000

reqTrxn.setAuthType(Constants.AUTHTYPE_AUTHONLY);

// set up the payment instrument

cc = new CreditCard();

cc.setId(100); // assuming we have previously registered credit

                // card with instrument id 100

pr = // assuming payeel has already been configured with the
payment

        // service

        paymentService.oraPmtReq(ecAppId, "payeel", "", cc, t,

        reqTrxn);

resp = (CoreCreditCardAuthResp) checkResult(pr);

if (resp == null) return;

System.out.println("Request finished with transaction id: " +

resp.getTID());

}

```

Registering a Purchase Card

```

public void instrAPISample(PaymentService paymentService,

                            int ecappId)    {

    PSResult pr;

    Object obj;

    PurchaseCard pc;

    Address addr;

```

```

int instrid_pc;

String payerid = "payer1";

addr = new Address("Line1", "Line2", "Line3",

                  "Redwood Shores", "San Mateo", "CA",

                  "US", "94065");

// purchase card

pc = new PurchaseCard();

pc.setName("My Purchase Card");

pc.setFIName("Paymentech");

pc.setInstrBuf("This is my purchase card description.");

pc.setInstrNum("4111111111111111"); // the purchase card

                                   // number

pc.setCardType("Constants.CCTYPE_VISA"); // the purchase

// card type, should match the purchase card number, if

// set

pc.setCardSubtype("P");

pc.setExpDate(new java.sql.Date(101, 0, 10));

                                   // Jan 10, 2001

pc.setHolderName("Mary Smith");

pc.setHolderAddress(addr);

// add the purchase card

pr = paymentService.oraInstrAdd(ecappId, payerid, pc);

obj = checkResult(pr);

if (obj == null) return; // registration failure

instrid_pc = ((Integer) obj).intValue();

```

```

        System.out.println("Purchase Card registered " +

                           "successfully with instrument id " +

                           instrid_pc);
    }

```

Sending a Purchase Card Authorization Request

```

// perform an ONLINE purchase card authorization with

// payment service

public void paymentAPISample(PaymentService paymentService,

                             int ecAppId)  {

    Bill t;

    PurchaseCardReq reqTrxn;

    PurchaseCard pc;

    PSResult pr;

    CoreCreditCardAuthResp resp; // since purchase card

    // authorization responses are identical to credit card

    // responses. See javadoc for details.

    // set up the tangible object

    t = new Bill();

    t.setId("orderId1");

    t.setAmount(new Double(21.00));

    t.setCurrency("USD");

    t.setRefInfo("refInfo");

    t.setMemo("memo");

    t.setUserAccount("userAcct");

    // set up the transaction object

```

```

reqTrxn = new PurchaseCardReq();

reqTrxn.setNLSLang("American_America.US7ASCII");

reqTrxn.setMode(Transaction.ONLINE);

reqTrxn.setSchedDate(new java.sql.Date(100, 5, 10));

                                // June 10, 2000

reqTrxn.setAuthType(Constants.AUTHTYPE_AUTHONLY);

reqTrxn.setPONum("PONum");

reqTrxn.setTaxAmount("1.50");

reqTrxn.setShipToZip("94065");

reqTrxn.setShipFromZip("94404");

// set up the payment instrument

pc = new PurchaseCard();

pc.setId(100); // assuming we have previously registered

                                // purchase card with instrument id 100

pr = // assuming payeel has already been configured with

                                // the payment service

                                paymentService.oraPmtReq(ecAppId, "payeel", "", pc,

                                                                t, reqTrxn);

resp = (CoreCreditCardAuthResp) checkResult(pr);

if (resp == null) return;

System.out.println("Request finished with " +

                                "transaction id: " + resp.getTID());

}

```

PL/SQL APIs for Electronic Commerce Applications

Oracle Payments provides PL/SQL APIs to those electronic commerce applications that

require or prefer PL/SQL interfaces for processing payment operations. There is an additional HTTP call when PL/SQL APIs are called. When electronic commerce applications invoke these PL/SQL APIs, the APIs in return, call the electronic commerce servlet through HTTP.

Oracle Payments PL/SQL APIs provide all payment related processing and two Risk APIs. The functionality of these APIs is the same as the Java APIs.

PL/SQL APIs are created as part of IBY_FND CPT_TRXN_PUB package and these packages are installed in the APPS schema.

New parameters are included to pass voice authorization flag and authorization date to support passing correct parameters for Payments need:

- pmtreqtrxn_rec.voiceauthflag
- pmtreqtrxn_rec.authcode
- pmtreqtrxn_rec.DateOfVoiceAuthorization

Requirements

Requirements include the following:

1. PL/SQL Package **IBY_FND CPT_TRXN_PUB** must be installed in the APPS schema.
2. An administrator must set up Oracle Payments URL property to Oracle Payments electronic commerce servlet's URL using the Oracle Payments administration user interface before invoking the APIs.

The following PL/SQL code helps you to understand how Oracle Payments PL/SQL APIs can be invoked. This example code invokes the Payment Request API using a credit card. It also passes risk related information for risk evaluation.

DECLARE

```
p_api_version          NUMBER := 1.0;

--To initialize message list.

p_init_msg_list VARCHAR2(2000) := FND_API.G_TRUE;

p_commit              VARCHAR2(2000) := FND_API.G_FALSE;

p_validation_level    NUMBER := FND_API.G_VALID_LEVEL_FULL;

p_ecapp_id            NUMBER := 0;

p_payee_rec           IBY_FND CPT_TRXN_PUB.Payee_rec_type;
```

```

p_payer_rec          IBY_FNDCPT_TRXN_PUB.Payer_rec_type;

p_pmtinstr_rec       IBY_FNDCPT_TRXN_PUB.PmtInstr_rec_type;

p_tangible_rec       IBY_FNDCPT_TRXN_PUB.Tangible_rec_type;

p_pmtreqtrxn_rec     IBY_FNDCPT_TRXN_PUB.PmtReqTrxn_rec_type;

p_riskinfo_rec       IBY_FNDCPT_TRXN_PUB.RiskInfo_rec_type;

x_return_status      VARCHAR2(2000);

                                -- output/return status

x_msg_count          NUMBER;

                                -- output message count

x_msg_data           VARCHAR2(2000);

                                -- reference string for getting
output message text

x_reqresp_rec        IBY_FNDCPT_TRXN_PUB.RegResp_rec_type;

                                -- request specific output
response object

l_msg_count          NUMBER;

l_msg_data           VARCHAR2(2000);

BEGIN

p_ecapp_id := 66;           -- iPayment generated ECAppID

p_payee_rec.Payee_ID := 'ipay-payee1';   -- payee's ID

p_payer_rec.Payer_ID  := 'ipay-cust1';    -- payer's ID

p_payer_rec.Payer_Name := 'Cust1';        -- Payer's (Customer's
name)

p_pmtreqtrxn_rec.PmtMode := 'ONLINE';

                                -- Payment mode (Can
be ONLINE/OFFLINE)

p_tangible_rec.Tangible_ID := 'tangible_id1'; -- Tangible ID / order
ID

```

```

    p_tangible_rec.Tangible_Amount := 25.50; -- Amount for the transaction

    p_tangible_rec.Currency_code := 'USD';    -- Currency for the
transaction

    p_tangible_rec.RefInfo := 'test_refinfo3';

    p_pmtreqtrxn_rec.Auth_Type := upper('authonly');    -- request type

    p_pmtinstr_rec.CreditCardInstr.CC_Type := 'Visa';

-- payment instrument type

    p_pmtinstr_rec.CreditCardInstr.CC_Num := '4111111111111111';

-- payment instrument number

    p_pmtinstr_rec.CreditCardInstr.CC_ExpDate := to_char(sysdate+300);

-- payment instr. Expiration date

--5. RISK INPUTS

    p_riskinfo_rec.Formula_Name := 'test3';    -- Risk formula name

    p_riskinfo_rec.ShipToBillTo_Flag := 'TRUE';

-- Flag showing if ship to address same
as Bill to address

    p_riskinfo_rec.Time_Of_Purchase := '08:45';

-- Time of purchase

    IBY_FNDCPT_TRXN_PUB.OraPmtReq

( p_api_version,

p_init_msg_list,

p_commit,

p_validation_level,

    p_ecapp_id ,

    p_payee_rec,

    p_payer_rec,

```



```

        p_pmtinstr_rec,

        p_tangible_rec,

        p_pmtreqtrxn_rec,

        p_riskinfo_rec ,

        x_return_status,

        x_msg_count ,

        x_msg_data ,

        x_reqresp_rec);

END;

Payment Request Related Response. Printing Only If Status Is Success

If(Char(X_Reqresp_Rec.Response.Status = 'S') Then

    -- Offline Mode Related Response

    If P_Pmtreqtrxn_Rec.Pmtmode = 'OFFLINE' Then

        Dbms_Output.Put_Line('Transaction ID = ' || To_Char
(X_Reqresp_Rec.Trxn_ID));

        Dbms_Output.Put_Line (' X_Reqresp_Rec.OfflineResp.
Earliestsettlement_Date = ' ||

To_Char(X_Reqresp_Rec.
OfflineResp.Earliestsettlement_Date));

        Dbms_Output.Put_Line('X_Reqresp_Rec.OfflineResp.
Scheduled_Date = ' ||

To_Char(X_Reqresp_Rec.OfflineResp.Scheduled_Date));

    Else

        Dbms_Output.Put_Line('Transaction ID = ' || To_Char
(X_Reqresp_Rec.Trxn_ID));

        Dbms_Output.Put_Line('X_Reqresp_Rec.Authcode = ' ||
X_Reqresp_Rec.Authcode);

        Dbms_Output.Put_Line('X_Reqresp_Rec.Avscode = ' ||
X_Reqresp_Rec.Avscode);

```

```

        Dbms_Output.Put_Line
( '-----' );

        -- Risk Related Response

        If(X_Regresp_Rec.Riskrespincluded = 'YES') Then

                Dbms_Output.Put_Line
( '-----' );

                Dbms_Output.Put_Line(' X_Regresp_Rec.Riskresponse.
Risk_Score=   ' || X_Regresp_Rec.Riskresponse.Risk_Score );

                Dbms_Output.Put_Line('X_Regresp_Rec.Riskresponse.
Risk_Threshold_Val=   ' ||

                                                X_Regresp_Rec.
Riskresponse.Risk_Threshold_Val);

                Endif;

        Endif;

End If;

```

Security Considerations

Oracle Payments is architected to send credit card details in the URL. This architecture requires the logging levels on Apache to be lowered from the default to prevent the credit card information from appearing in the log files.

In the httpds.conf file, change:

LogFormat "%h %l %u %t \"%r\" %>s %b" common

to:

LogFormat "%h %l %u %t \"%U\" %>s %b" common

Oracle Payments Back-End APIs for Gateways

Integrating with Non-Oracle Applications

If you want to integrate with Non-Oracle Applications, please see the Oracle Integration Repository at <http://irep.oracle.com/>.

Profile Options

Profile Options

This appendix lists the profile options that affect the operation of Oracle Payments. This appendix includes a brief description of each profile option that you or your system administrator can set at the site, application, responsibility, or user levels.

During implementation, your system administrator sets a value for each user profile option to specify how Oracle Applications controls access to and processes data.

See Overview of Setting User Profiles, *Oracle Applications System Administrator's Guide*.

Profile Options Summary

This table indicates whether you can view or update profile options and at which System Administrator levels the profile options can be updated: at the user, responsibility, application, or site levels.

A *Required* profile option requires you to provide a value. An *Optional* profile option already provides a default value which you can change.

The key for this table is:

- **Update** - You can update the profile option
- **View Only** - You can view the profile option but cannot change it
- **No Access** - You cannot view or change the profile option value

Profile Options

Profile Option	Value	Default	User Access	System Admin Access: User	System Admin Access: Responsibility	System Admin Access: Application	System Admin Access: Site
IBY: ECAPP URL	Required	No Default	No Access	No Access	No Access	No Access	Update
IBY: HTTP Proxy	Optional	No Default	No Access	No Access	No Access	No Access	Update
IBY: No Proxy Domain	Optional	No Default	No Access	No Access	No Access	No Access	Update
IBY: XML Base	Required	No Default	No Access	No Access	No Access	No Access	Update
IBY: JAVA XML Log File	Optional	No Default	No Access	No Access	No Access	No Access	Update
IBY: XML Temp Directory	Optional	No Default	No Access	No Access	No Access	No Access	Update
IBY: Outbound Payment Payer ID	Optional	No Default	No Access	No Access	No Access	No Access	Update

Profile Option	Value	Default	User Access	System Admin Access: User	System Admin Access: Responsibility	System Admin Access: Application	System Admin Access: Site
IBY: Outbound Payment System Suffix	Optional	No Default	No Access	No Access	No Access	No Access	Update
IBY: Default Payee for BR Remittance	Optional	No Default	No Access	No Access	No Access	No Access	Update
IBY: UI Visibility Class	Optional	No Default	No Access	No Access	No Access	No Access	Update
IBY: Wallet Location	Optional	No Default	No Access	No Access	No Access	No Access	Update
IBY: Wallet Password	Optional	No Default	No Access	No Access	No Access	No Access	Update
IBY: Registered Instrument Encryption	Optional	No	No Access	No Access	No Access	No Access	Update

Profile Option	Value	Default	User Access	System Admin Access: User	System Admin Access: Responsibility	System Admin Access: Application	System Admin Access: Site
IBY: Daily Business Close Reporting Currency	Required	USD	Update	Update	Updated	Update	Update

Oracle Payments Profile Options

You can use the System Administrator responsibility to set the Oracle Payments profile options.

IBY: ECAPP URL

This property contains the following URL:

`http://machine:port/<jsp>/ecapp?`

Replace the machine and port with the names of the actual machine and the actual port where the Oracle Payments ECServlet is installed. Also, make sure that ? is present at the end of the URL or append ? at the end.

This information is mandatory if your EC applications use Oracle Payments PL/SQL APIs or if your application is an Oracle 3i client.

IBY: HTTP Proxy

This property specifies the proxy-URL. For example, `http://www-proxy.us.oracle.com`.

To set up this property with an empty value, insert a string starting with <. For example, `<none>`.

IBY: No Proxy Domain

This property specifies the domain name for which no proxy is needed. For example, `us.oracle.com`.

To set up this property with an empty value, insert a string starting with <. For example, `<none>`.

IBY: XML Base

This property specifies the location of files required by Oracle Payment's XML framework, such as Oracle Payments DTD files. This property should give the location of the \$IBY_TOP/xml directory, where \$IBY_TOP is expanded to its fully qualified path name. For example, /usr/appl_top/iby/11.5.0/xml

IBY: JAVA XML Log File

This optional property gives the full-qualified pathname of the debug file where XML messages should be written. This file is similar in purpose to the Oracle Payments debug file, but has been separated from it since XML messages are much larger than single debug statements. If no value is specified for this property, then XML logging is disabled.

IBY: XML Temp Directory

Temporary XML work directory, which must be writable by Oracle Payment's application server. This parameter is optional, but will reduce Oracle Payment's memory usage if provided.

IBY: Outbound Payment Payer ID

Select from the list of values displayed, the payee in Oracle Payments issuing the payment order to the bank. You can set this only at the site level. You need to define this to send transactions from Oracle Payables to Oracle Payments.

IBY: Outbound Payment System Suffix

Enter the three-letter suffix of the payment system that will handle your outbound payment instructions.

IBY: Default Payee for BR Remittance

Select from the list of values displayed, the payee in Oracle Payments remitting the Bills Receivable. You can set this only at the site level. You need to define this to send BR remittance batch from Oracle Receivables to Oracle Payments.

IBY: UI Visibility Class

You can define the visibility class profile option at different levels. This value will determine what data a user can see in the Oracle Payments Operation UI and what mask is applied to the data before displaying it.

IBY: Wallet Location

Location of the Oracle Wallet.

IBY: Wallet Password

Password to open the Oracle Wallet.

IBY: Registered Instrument Encryption

Determines whether registered payment instruments must be stored in encrypted format; if set to Yes, the system security key must have been provided to the Oracle Payments engine in order to register/modify payment instruments; use encrypted registered payment instruments as part of a transaction. The default value is No.



Customizations

Search Pages

The list below is a list of Funds Disbursement search pages that you can customize.

- Payment Process Requests
- Payment Instructions
- Payments

Each page in the preceding list has a Save Search button, which enables you to create a new view.

The list below is a list of Funds Capture search pages that you can customize.

- Settlement Batches
- Authorizations
- Settlements
- Credits
- Transaction Testing

Each page in the preceding list has a Save Search button, which enables you to create a new view.

Oracle Payment does not have any List of Values that are customizable. Additionally, there are no limitations on administrative customizations. That is, someone with administrative privileges, such as the System Administrator, can make look and feel changes to each page in the application.

Funds Capture Extract

Extract Structure

The XML Schema is the data source definition for all format templates. This means a single funds capture extract definition supports both bank account and credit card instrument types.

Element Definition Table Legend

The table below is an example of an element definition table entitled BankAccount.

BankAccount

<XML Tags>	Cardinality	Datatype	Description
<BankAccountID>	0..1	<Identifier>	Bank account identifier.
<BankAccountNumber>	0..n	String	Bank account number.

Each table includes the following columns:

- <XML Tags>
If indicated in **boldface**, it denotes the element is a complex aggregate; otherwise it is in a normal font.
- Cardinality
blank = cardinality of 1
? = cardinality of 0..1

* = cardinality of 0..n

+ = cardinality of 1..n

n = cardinality of n

- Datatype

Aggregate: Complex type consisting of child elements.

Type: Element conforms to a custom-defined type described in its own table.
Elements that share a type have identical child elements.

Scalar: This can be String, Integer, Real, Date, or Boolean and are the same as equivalent SQL types.

- Description

Describes the technical or business purpose of the element.

Extract Components

The funds capture extract consists of data elements organized hierarchically. Though the data within most such elements are generated through simple, low-cost data fetches from the Oracle Payments schema, some are the result of complex, high-cost function calls which result in unacceptable performance when creating an extract instance. Therefore, the extract engine supports a series of user-defined rules wherein certain expensive elements are not populated if the rule's conditions are met.

To support payment formats with unique data requirements, an extensibility element called Extend is present at every level of the extract, allowing you to provide the required data using your own custom functions.

As the data required to create funds capture instructions change over time, the extract also changes by the addition of new elements. To support easy transition to new extract definitions, each extract has a version number associated with it which is stored with every user-defined payment format. The product retains the ability to generate all previous extract versions and, based upon the stored version number, provides each payment format with the appropriate extract instance.

Funds Capture Extract

The funds capture extract has a 5-level structure, where each level, except the last, contains one or more sub-levels.

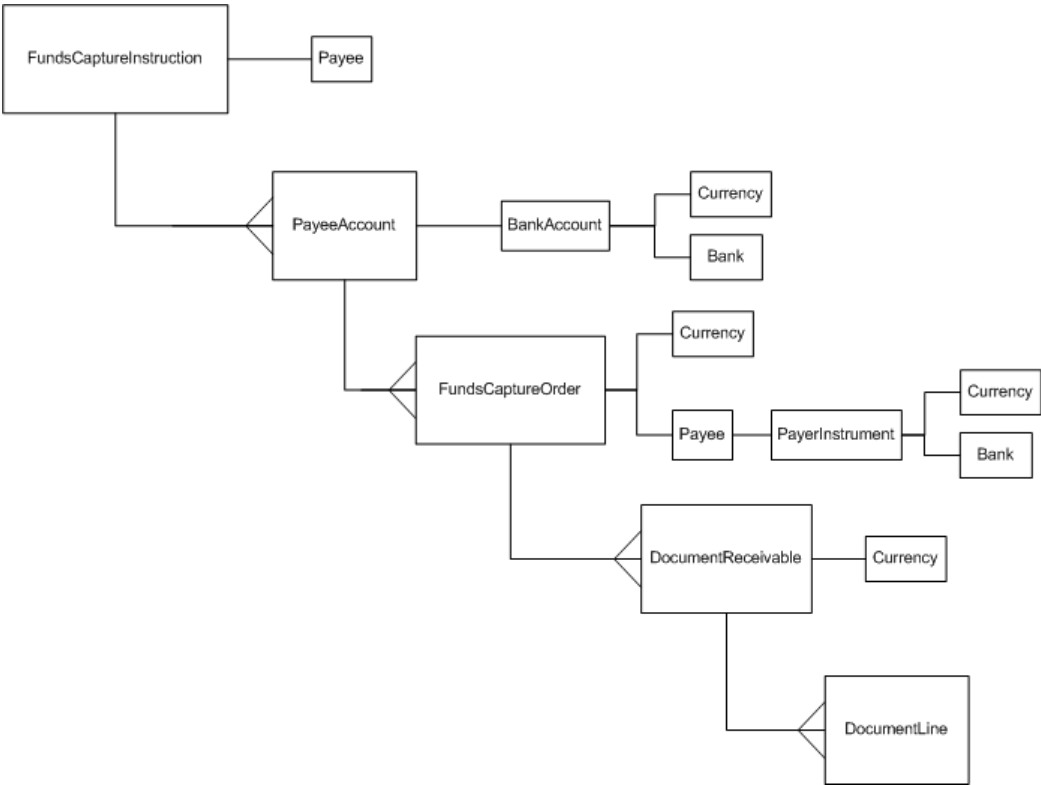
The top level is the funds capture instruction level, and represents a single instructions file to be delivered to the payment system. The 2nd level includes 1 or more payee accounts, each associated with a single currency and financial institution, or bank. The payee accounts act as destinations for a series of funds capture orders.

A funds capture order, located at the 3rd level, is associated with a payment currency,

payer information and payer bank account. A funds capture order also acts as a grouping for multiple documents receivable that reside at the 4th level. Each document receivable is associated with a single order and currency and contains multiple document lines, located at the 5th level and representing a line item from the associated order.

The diagram below illustrates the logical structure of the funds capture extract.

Logical Structure of the Funds Capture Extract



Funds Capture Instruction Elements

The root element of an funds capture extract, corresponding to the document or file to be eventually delivered to the external payment system, is FundsCaptureInstruction.

FundsCaptureInstruction

<XML Tags>	Cardinality	Datatype	Description
<InstructionInfo>	1	Aggregate	Information about the instruction.

<XML Tags>	Cardinality	Datatype	Description
<InstructionSequence>	1	Aggregate	Sequential, possibly periodic, identifier of the instruction.
<InstructionTotals>	1	Aggregate	Totals for the instruction, such as funds capture amount totals and payment instrument counts.
<InstructionGrouping>	1	Aggregate	Instruction grouping information.
<PayeeAccount>	1..n	Aggregate	Account (either a bank account or payment system merchant account) where captured funds will be deposited.
<PayeeAccount>>	1..n	Aggregate	Account (either a bank account or payment system merchant account) where captured funds will be deposited.
<Extend>	0..n	<NameValue>	Extensibility element. To be filled with custom user data.

InstructionInfo

<XML Tags>	Cardinality	Datatype	Description
<InstructionInternalID>	1	Integer	Indicates the unique identifier assigned internally to this funds capture instruction

<XML Tags>	Cardinality	Datatype	Description
<InstructionName>	1	String	Name of the instruction.
<InstructionCreationDate>	1	Time	Date of the instruction's creation.
<InstructionSentDate>	1	Time	Date the instruction was sent.
<InstructionStatus>	1	<Lookup>	Current status of the instruction.

InstructionSequence

<XML Tags>	Cardinality	Datatype	Description
<SequenceName>	1	String	Name/code of the sequence.
<LastValue>	1	Integer	Last/current value of the sequence.

InstructionTotals

<XML Tags>	Cardinality	Datatype	Description
<PayeeAccountCount>	1	Integer	The number of payee accounts in instruction.

InstructionGrouping

<XML Tags>	Cardinality	Datatype	Description
<SettlementDate>	0..1	Time	Time of settlement.

<XML Tags>	Cardinality	Datatype	Description
<SettlementCurrency>	0..1	<Currency>	Settlement currency information.
<PayeeLegalEntity>	0..1	Aggregat	Payee legal entity information.
<PayeeOrganization>	0..1	<1stPartyOrgInfo>	Payee first party organization information.
<PayeeBankAccount>	0..1	<IntBankAccount>	Payee bank account information.

PayeeLegalEntity

<XML Tags>	Cardinality	Datatype	Description
<PartyInternalID>	0..1	Integer	Indicates the source of the party data (HZ_PARTIES) with the Identifier element holding PARTY_ID.
<PartyNumber>	1	String	User-assigned identifier for this external party.
<Name>		String	Full name of the party.
<PartyType>	1	<Lookup>	Lookup code for the party type.
<PartyType>	1	String	The party type.
<LegalEntityInternalID>	1	Integer	Legal entity data source identifier, with the identifier element holding LEGAL_ENTITY_ID.

<XML Tags>	Cardinality	Datatype	Description
<LegalEntityName>	1	String	Legal name of the party.
<Address>	1	<Address>	Party's address.
<ContactInfo>	1	<ContactInfo>	Party contact information.
<TaxRegistrationNumber>	0..1	String	Tax registration number.
<LegalEntityRegistrationNumber>	0..1	String	Legal registration number.
<LEDescriptiveFlexField>	0..1	<DescFlexField>	Legal entity descriptive flex fields.

Payee Account Level Elements

The payee account level consists of a BankAccount element, followed by funds capture total elements, and then 1 or more funds capture order elements.

PayeeAccount

<XML Tags>	Cardinality	Datatype	Description
<PaymentSystemAccount>	1	Aggregate	(Merchant) account name assigned to the payee by the acting payment system.
<Payee>	1	Aggregate	The payee information.
<OrderCount>	1	Integer	Number of funds capture order using this payer account/instrument as the funds source.

<XML Tags>	Cardinality	Datatype	Description
<AccountTotals>	1	Aggregate	Amount totals for the account.
<FundsCaptureOrder>	1..n	Aggregate	Collection of funds capture orders using the current instrument as the funds destination.
<Extend>	0..n	<NameValue>	Extensibility element; to be filled with custom user data.

Payee

<XML Tags>	Cardinality	Datatype	Description
<Name>	1	String	Payee business name.
<Address>	1	<Address>	Payee business address.
<ContactInfo>	1	<Contact>	Contact points for the payee party.
<MCC>	1	String	Merchant category code.

AccountTotals

<XML Tags>	Cardinality	Datatype	Description
<AuthorizationsTotal>	1	<Amount>	Total of all authorizations.
<CapturesTotal>	1	<Amount>	Total of all captures/settlements.

<XML Tags>	Cardinality	Datatype	Description
<CreditsTotal>	1	<Amount>	Total of all credits.

PaymentSystemAccount

<XML Tags>	Cardinality	Datatype	Description
<AccountName>	1	String	(Merchant) account name assigned to the payee by the acting payment system.
<AccountOption>	0..n	<NameValue>	Account configuration option or value.

Order Level Elements

An order corresponds to an individual transaction, such as an authorization.

Data sources for order-level elements come from tables IBY_TRXN_SUMMARIES_ALL (IBY_TS) , IBY_TRXN_CORE (IBY_TC), and IBY_TANGIBLE (IBY_TG). Joins are performed using column MTRXNID that acts as a primary key for the first 3 tables. IBY_TG is joined to IBY_TS using column MTANGIBLEID.

FundsCaptureOrder

<XML Tags>	Cardinality	Datatype	Description
<OrderSourceInfo>	1	Aggregate	Information about the order requestor.
<OrderNumber>	1	Aggregate	Number and identifiers associated with the order.
<PayeeOrderMemo>	0..1	String	Payee-assigned order memo.

<XML Tags>	Cardinality	Datatype	Description
<PayeeOrderRefID>	1	String	Payee-assigned order reference identifier.
<OrderMedium>	0..1	Enumeration	Medium by which the order was received. Values include: ECOMMERCE, RETAIL.
<OrderAmount>	1	<Amount>	Amount of the order.
<Payer>	1	<3rdPartyInfo>	Party information about the funds capture payer.
<PayerBankAccount>	a	<BankAccount>	The payer's bank account.
<BankAccountTransaction>	a	Aggregate	The bank account transaction for the funds capture.
See note in Description column.		See note in Description column.	Note: There is an exclusive choice between element groups a, b, and c. One of them may appear at the order level.
<PayerCreditCard>	b	<CreditCard>	The payer's credit card (a source for the funds capture).
<CreditCardTransaction>	b	Aggregate	The credit card transaction for the funds capture.

<XML Tags>	Cardinality	Datatype	Description
<OriginalCCTransaction>	b, 0..1	Aggregate	Original credit card request which is a follow-on for the current one. In almost all cases the originating request is an authorization and this element is not present when the request is an authorization.
<PayerDebitCard>	c	<DebitCard>	The payer's debit card.
<DebitCardTransaction>	c	Aggregate	The debit card transaction.
<OriginalDCTransaction>	1	Aggregate	The original debit card request; similar to element OriginalCCTransaction.
<DocumentReceivable>	0..1	Aggregate	Document receivable associated with the funds capture, if any.
<Extend>	0..n	<NameValue>	Extensibility element; to be filled with custom user data.

OrderSourceInfo

<XML Tags>	Cardinality	Datatype	Description
<ApplicationInternalID>	1	Integer	Internal identifier of the application originating the order request.

<XML Tags>	Cardinality	Datatype	Description
<ApplicationName>	1	String	Name of the application originating the order request.

OrderNumber

<XML Tags>	Cardinality	Datatype	Description
<PayeeOrderNumber>	1	String	Payee-assigned order number associated with the funds capture.

BankAccountTransaction

<XML Tags>	Cardinality	Datatype	Description
<ActionType>	1	Enumeration	Type of EFT transaction attempted. Values include: DEBIT, CREDIT, VERIFY, VALIDATE.
<TransactionDate>	1	Date	Date of the funds capture.
<SettlementDueDate>	1	Date	Date of transaction.
<AuthorizationMethod>	1	Enumeration	Method by which the payer authorized the transaction. Values include: WRITTEN, INTERNET_FORM.

<XML Tags>	Cardinality	Datatype	Description
<DeliveryMethod>	1	Enumeration	Method by which the transaction is to be delivered. Values include: ACH, FASCIMILE.
<TransferType>	1	Enumeration	Bank account transfer type.
<SettlementCustomer Reference>	1	String	Settlement user reference.
<SettlementFactored>	1	Boolean	Settlement factor flag.
<BillReceivableData>	0..1	Aggregate	Bills receivable data.
<BankChargeBearer>	1	<Lookup>	Bearer of bank account charges.
<DebitAuthorization >	1	Aggregate	Debit authorization information for the payee bank account.
<DebitNotification>	1	Aggregate	Debit notification information for the payee bank account.
<Extend>	0..n	<NameValue>	Extensibility element; to be filled with custom data.

BillReceivableData

<XML Tags>	Cardinality	Datatype	Description
<MaturityDate>	1	Date	Maturity of the bills receivable.
<BRType>	1	Enumeration	Bills receivable transaction type.

DebitAuthorization

<XML Tags>	Cardinality	Datatype	Description
<DebitAuthRefCode>	1	String	Debit authorization reference code.
<DebitAuthMethod>	1	Enumeration	Debit authorization method.
<DebitAuthGranted>	1	Boolean	Debit authorization granted.

DebitNotification

<XML Tags>	Cardinality	Datatype	Description
<DeliveryMethod>	1	String	Method by which the transaction is to be delivered. Values include: ACH, FASCIMILE.
<EmailAddress>	1	String	E-mail address where notification is sent.
<FaxNumber>	1	String	Fax number where notification is sent.

CreditCardTransaction

<XML Tags>	Cardinality	Datatype	Description
<ActionType>	1	Enumeration	Type of credit card transaction. Values include: AUTHORIZATION, AUTHCAPTURE, VOICEAUTH, CAPTURE, CREDIT, RETURN, VOID.
<TransactionDate>	1	Date	Date of the funds capture.
<TraceNumber>	0..1	String	Payment system-provided trace number.
<POSData>	0..1	Aggregate	Point-of-sale data for card-present transactions.
<AuthCode>	0..1	String	Authorization code. Present for voice auth transactions.
<VoiceAuthFlag>	0..1	Boolean	Indicates whether the transaction was a voice authorization.
<Extend>	0..n	<NameValue>	Extensibility element. To be filled with custom data.

OriginalCCTransaction

<XML Tags>	Cardinality	Datatype	Description
<ActionType>	1	Enumeration	Type of credit card transaction. Values include: AUTHORIZATION, AUTHCAPTURE, VOICEAUTH, CAPTURE, CREDIT, RETURN, VOID.
<TransactionDate>	1	Date	Date of the funds capture.
<TraceNumber>	0..1	String	Payment system-provided trace number.
<POSDData>	0..1	Aggregate	Point-of-sale data for card-present transactions.
<AuthCode>	0..1	String	Authorization code provided by the payment system during the initial authorization.
<VoiceAuthFlag>	0..1	Boolean	Indicates whether the transaction was a voice authorization.
<Amount>	1	<Amount>	Transaction amount.
<AVSCode>	0..1	String	AVS response from the initial authorization.
<ReferenceCode>	0..1	String	Reference code.
<SecurityValueCheck>	0..1	String	Result of the security value check.

<XML Tags>	Cardinality	Datatype	Description
<PaymentSystemCode>	0..1	String	Payment system code returned during the initial authorization.
<Extend>	0..n	<NameValue>	Extensibility element. To be filled with custom data.

DebitCardTransaction

<XML Tags>	Cardinality	Datatype	Description
<ActionType>	1	Enumeration	Type of EFT transaction. Values include: DEBIT, CREDIT, VERIFY, VALIDATE.
<TransactionDate>	1	Date	Date of the funds capture.
<Extend>	0..n	<NameValue>	Extensibility element. To be filled with custom data.

OriginalDCTransaction

<XML Tags>	Cardinality	Datatype	Description
<ActionType>	1	Enumeration	Type of debit card transaction attempted.
<TransactionDate>	1	Date	Date of the funds capture.

<XML Tags>	Cardinality	Datatype	Description
<TraceNumber>	0..1	String	Payment system-provided trace number.
<AuthCode>	0..1	String	Authorization code provided by the payment system during the initial authorization.
<PaymentSystemCode>	0..1	String	Payment system code returned during the initial authorization.
<DebitNetworkCode>	1	String	Debit network code.
<Extend>	0..n	<NameValue>	Extensibility element. To be filled with custom data.

POSData

<XML Tags>	Cardinality	Datatype	Description
<ReaderCapability>	1	Enumeration	The card reader capability.
<EntryMode>	1	Enumeration	Card data entry mode.
<CardIdMethod>	1	Enumeration	Card-holder identification method.
<AuthSource>	1	Enumeration	Authorization source.
<ReaderData>	1	String	Card reader data. Must be in text encoded format if binary.

Document Level Elements

These elements correspond to individual documents receivable.

DocumentReceivable

<XML Tags>	Cardinality	Datatype	Description
<DocumentID>	1	String	Identifier assigned to this document receivable.
<DocumentStatus>	1	<LookupCode>	Document status.
<DocumentDate>	1	Date	Document date.
<DocumentCreationDate>	1	Date	Document creation date.
<PaymentDueDate>	1	Date	Document payment due date.
<DocumentType>	1	<Lookup>	Document type.
<DocumentDescription>	1	String	User-provided document description.
<TotalDocumentAmount>	1	<Amount>	Total amount of the document.
<PaymentAmount>	1	<Amount>	Amount paid.
<Charge>	0..n	Aggregate	Charges applied to the document.
<Discount>	0..n	Aggregate	Discounts applied to the document.
<Tax>	0..n	Aggregate	Taxes applied to the document.

<XML Tags>	Cardinality	Datatype	Description
<ShipmentOrigin>	1	<Address>	Shipping origin of goods provided.
<ShipmentDestination>	1	<Address>	Shipping destination of goods provided.
<DocumentLine>	0..n	Aggregate	Document lines.
<Extend>	0..n	<NameValue>	Extensibility element. To be filled with custom user data.

Document Line Level Elements

These elements correspond to lines of documents receivable.

DocumentLine

<XML Tags>	Cardinality	Datatype	Description
<LineID>	1	String	Identifier for the document line.
<LineNumber>	1	Integer	Number for the document line.
<PONumber>	0..1	String	Purchase order number.
<LineType>	1	<Lookup>	Document type.
<LineDescription>	1	String	Description of the document line.
<LineAmount>	1	<Amount>	Total amount for the line.
<UnitRate>	0..1	Real	Price per unit of this item.

<XML Tags>	Cardinality	Datatype	Description
<Quantity>	0..1	Real	Number of line item units.
<UnitOfMeasure>	0..1	String	Unit of measure lookup code.
<ProductCode>	0..1	String	Product code.
<CommodityCode>	0..1	String	Commodity code.
<Charge>	1..n	Aggregate	Charges applied to the document line.
<Discount>	1..n	Aggregate	Discounts applied to the document line.
<Tax>	1..n	Aggregate	Taxes applied to the document line.
<DocumentLine>	1..n	Aggregate	Document lines.
<Extend>	0..n	<NameValue>	Extensibility element. To be filled with custom user data.

Common Elements

Generic Elements

Currency information comes from FND_CURRENCIES (FND_C), with a join performed on the currency code if detailed information is required.

Currency

<XML Tags>	Cardinality	Datatype	Description
<Code>	1	String	Currency code. Acts as foreign key into table FND_CURRENCIES.
<Symbol>	0..1	String	Currency symbol.
<MinAccountableUnit>	0..1	Integer	Minimum accountable units.
<Precision>	1	Integer	Precision of the currency in terms of its smallest subunits.

Amount

<XML Tags>	Cardinality	Datatype	Description
<Value>	1	Real	Scale of the amount.
<Currency>	1	Aggregate	Currency of the amount.

NameValue

<XML Tags>	Cardinality	Datatype	Description
<Name>	1	String	Name.
<Value>	1	String	Value.

Lookup

<XML Tags>	Cardinality	Datatype	Description
<Code>	1	String	Lookup code.
<Meaning>	1	String	Code meaning.
<FormatValue>	0..1	String	Value required by the payment format using this lookup.

DescFlexField

<XML Tags>	Cardinality	Datatype	Description
<AttributeCategory>	1	String	Descriptive flexfield attribute category.
<Attribute>	0..n	String	Descriptive flexfield attribute.

Address Elements**Address**

<XML Tags>	Cardinality	Datatype	Description
<AddressInternalID>	1	Integer	The data source of the address data.
<AddressLine1>	1	String	Address line 1.
<AddressLine2>	0..1	String	Address line 2.
<AddressLine3>	0..1	String	Address line 3.
<City>	1	String	City.

<XML Tags>	Cardinality	Datatype	Description
<County>	0..1	String	County.
<State>	1	String	State.
<Country>	1	String	Country code.
<ISO3DigitCountry>	0..1	String	ISO country code.
CountryName	0..1	String	Descriptive country name.
<PostalCode>	1	String	postal code
<PreFormattedConcatenatedAddress>	0..1	String	Formatted concatenated address.
<PreFormattedMailingAddress>	0..1	String	Formatted mailing address.
<AddressName>	1	String	Name of the address/location.
<AlternateAddressName>	0..1	String	Alternate address/location name.

Contact Information Elements

<i>ContactInfo</i>			
<XML Tags>	Cardinality	Datatype	Description
<ContactName>	1	<PersonName>	The contact's personal name.
<ContactLocators>	1	<Locators>	Various means by which the contact may be located or reached.

PersonName

<XML Tags>	Cardinality	Datatype	Description
<FirstName>	1	String	Person's first name.
<LastName>	1	String	Person's first name.

Locators

<XML Tags>	Cardinality	Datatype	Description
<PhoneNumber>	1	String	Contact phone number.
<FaxNumber>	1	String	Contact fax machine number.
<EmailAddress>	1	String	Contact e-mail address.
<Website>	0..1	String	Contact website.

Bank Account Elements**BankAccount**

<XML Tags>	Cardinality	Datatype	Description
<BankAccountInternalID>	1	Integer	Identifies the data source of the parent aggregate.
<BankName>	1	String	Name of the bank.
<BankNumber>	1	String	Number assigned to bank.

<XML Tags>	Cardinality	Datatype	Description
<BranchInternalID>	1	Integer	Identifies the data source of all subsequent branch-related elements.
<BranchName>	1	String	Name of the bank branch.
<BranchNumber>	1	String	Number of the bank branch.
<BranchType>	1	<Lookup>	Bank branch type.
<BankAccountName>	1	String	Name of the bank account.
<AlternateBankAccountName>	0..1	String	Alternate bank account name.
<BankAccountNumber>	1	String	The bank account number.
<UserEnteredBankAccountNumber>	1	String	User-entered, denormalized bank account number.
<SwiftCode>	1	String	SWIFT code of the bank account.
<IBANNumber>	1	String	IBAN of the bank account.
<CheckDigits>	1	String	Check digits of the bank account number.
<BankAccountType>	1	<Lookup>	The account type.
<BankAccountCurrency>	1	<Currency>	Currency by which accounts funds are denominated.

<XML Tags>	Cardinality	Datatype	Description
<BankAddress>	1	<Address>	Address of the bank.
<BankContact>	0..1	<ContactInfo>	Contact at the bank.

<IntBankAcct>

<XML Tags>	Cardinality	Datatype	Description
See Note in Description column.	1	<BankAccount>	Note: All the elements of aggregate type <BankAccount> follow inline, and are not contained as the subelements of some parent element.
<DescriptiveFlexField>	1	<DescFlexField>	Bank account descriptive flexfields.
<FederalBankAccountInfo>	0..1	<FederalBankAccountInfo>	Federal bank account information.
<EFTUserNumber>	0..1	Aggregate	Electronic funds transfer user number.

<ExtBankAcct>

<XML Tags>	Cardinality	Datatype	Description
See Note in Description column.	1	<BankAccount>	Note: All the elements of aggregate type <BankAccount> follow inline, and are not contained as the subelements of some parent element.
<AccountHolderName>	1	String	The account holder name.

EFTUserNumber

<XML Tags>	Cardinality	Datatype	Description
<AccountLevelEFTNumber>	1	String	Account-level EFT user number.
<BranchLevelEFTNumber>	1	String	Branch-level EFT user number.

FederalBankAccountInfo

<XML Tags>	Cardinality	Datatype	Description
<FederalRFCIdentifier>	1	String	Identifier of the US Treasury Regional Finance Center (RFC) where disbursement originates for federal agencies.

<XML Tags>	Cardinality	Datatype	Description
<FederalAgencyLocationCode>	1	String	Agency Location Code used by federal agency.
<FederalAbbreviatedAgencyCode>	1	String	Federal agency abbreviated code.
<FederalEmployerIdentificationNumber>	1	String	Federal employer identification number.

Credit Card Elements

CreditCard

<XML Tags>	Cardinality	Datatype	Description
<CardNumber>	1	String	Credit card number.
<CardExpiration>	1	Date	Card expiration date.
<SecurityCode>	0..1	String	CVV2 or similar security value.
<CardIssuer>	0..1	Enumeration	Credit card issuer type: For example, VISA and MASTERCARD.
<CardHolder>	0..1	Aggregate	Card holder information.
<CardSubtype>	0..1	Enumeration	Purchase card subtype. Possible values defined by lookup IBY_PURCHASECARD_SUBTYPE.

<XML Tags>	Cardinality	Datatype	Description
<CardDataLevel>	0..1	Enumeration	Level of data supported with this instrument; possible values defined by lookup IBY_PCARD_DATA_LEVEL.

CardHolder

<XML Tags>	Cardinality	Datatype	Description
<HolderName>	1	<PersonName>	Card holder name.
<BillingAddress>	1	Aggregate	Billing address of the card holder.
<PhoneNumber>	0..1	String	Card holder phone number.
<EmailAddress>	0..1	String	Card holder e-mail address.

Debit Card Elements

<DebitCard>

<XML Tags>	Cardinality	Datatype	Description
<CardNumber>	1	String	Credit card number.
<CardExpiration>	1	Date	Card expiration date.
<SecurityValue>	0..1	String	Account security code.

<XML Tags>	Cardinality	Datatype	Description
<CardHolder>	0..1	Aggregate	Card holder information.

Party Elements

Party elements are those which describe a participant in a financial transaction, with varying details depending on whether they are the 1st party (user) or 3rd party (external).

1stPartyInfo

<XML Tags>	Cardinality	Datatype	Description
<PartyInternalID>	0..1	Integer	Indicates the source of the party data (HZ_PARTIES) with the Identifier element holding PARTY_ID.
<PartyNumber>	1	String	User-assigned identifier for this external party.
<Name>	1	String	Full name of the party.
<PartyType>	1	<Lookup>	Lookup code for the party type.
<PartyType>	1	String	The party type.
<LegalEntityInternalID>	1	Integer	Legal entity data source identifier, with the identifier element holding LEGAL_ENTITY_ID.
<LegalEntityName>	1	String	Legal name of the party.

<XML Tags>	Cardinality	Datatype	Description
<Address>	1	<Address>	Party's address.
<ContactInfo>	1	<ContactInfo>	Party contact information.

3rdPartyInfo

<XML Tags>	Cardinality	Datatype	Description
<PartyInternalID>	0..1	Integer	Indicates the source of the party data (HZ_PARTIES) with the Identifier element holding PARTY_ID.
<PartyNumber>	1	String	User-assigned identifier for this external party.
<Name>	1	String	Full name of the party.
<PartyType>	1	<Lookup>	Lookup code for the party type.
<PartyType>	1	String	The party type.
<TaxIdentifier>	0..1	String	Tax number of party.
<Address>	1	<Address>	Party's address.
<FirstPartyReference>	1	String	Identifier by which this third party refers to the first.

1stPartyOrgInfo

<XML Tags>	Cardinality	Datatype	Description
<OrganizationInternalID>	1	Integer	Organization internal identifier.
<OrganizationType>	0..1	<Lookup>	Organization type.
<OrganizationName>	0..1	String	Organization name.

Document Line Elements**Discount**

<XML Tags>	Cardinality	Datatype	Description
<Amount>	1	Aggregate	Amount of the discount.
<RatePercent>	1	Real	Discount rate in percentage.
<DiscountType>	0..1	String	Type of discount.

Charge

<XML Tags>	Cardinality	Datatype	Description
<Amount>	1	Aggregate	Amount of the charge.
<RatePercent>	1	Real	Charge rate as a percentage.
<ChargeType>	0..1	String	Type of charge.

Tax

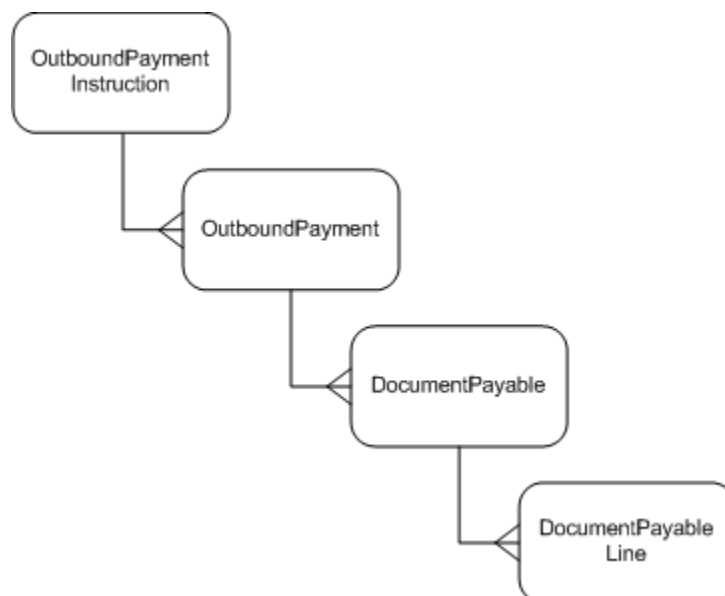
<XML Tags>	Cardinality	Datatype	Description
<Amount>	1	Aggregate	Amount of the tax.
<RatePercent>	1	Real	Tax rate as a percentage.
<Type>	0..1	String	Type of tax.
<TaxJurisdiction>	0..1	String	Tax jurisdiction.

Funds Disbursement Extract

Extract Structure Overview

The disbursement payment instruction extract has a 4 level structure. Each level except the last containing one or more sub-levels.

The top level is the payment instruction level. One disbursement extract will have one and only payment instruction to be delivered to the payment system. The second level is the payment. It is called OutboundPayment to distinguish with funds capture payment orders. One payment must have one or more document payables. The third level is document payable. Each document payable contains multiple document lines, located at the forth level and equivalent to a line item of an invoice.



Element Definition Table Legend

The table below is an example of an element definition table entitled BankAccount.

BankAccount

<XML Tags>	Cardinality	Datatype	Description
<BankAccountID>	0..1	<Identifier>	Bank account identifier.
<BankAccountNumber>	0..n	String	Bank account number.

Each table includes the following columns:

- **<XML Tags>**
If indicated in **boldface**, it denotes the element is a complex aggregate; otherwise it is in a normal font.
- **Cardinality**
blank = cardinality of 1
? = cardinality of 0..1
* = cardinality of 0..n
+ = cardinality of 1..n
n = cardinality of n
- **Datatype**
Aggregate: Complex type consisting of child elements.
Type: Element conforms to a custom-defined type described in its own table. Elements that share a type have identical child elements.
Scalar: This can be String, Integer, Real, Date, or Boolean and are the same as equivalent SQL types.
- **Description**
Describes the technical or business purpose of the element.

Payment Instruction Level Elements

OutboundPaymentInstruction

OutboundPaymentInstruction

<XML Tags>	Cardinality	Datatype	Description
<PaymentInstructionInfo>	1	Aggregate	Information about the payment instruction.
<PaymentProcessProfile>	1	Aggregate	Payment process profile used to create this payment instruction.
<PaymentFormat>	1	Aggregate	Format of the payment instruction.
<CheckFormatInfo>	0..1	Aggregate	Check format specific information such as paper document name and paper stock type.
<InstructionTotals>	1	Aggregate	Totals for the payment instruction, such as the payment count.
<InstructionGrouping>	1	Aggregate	Criteria by which payments were grouped to form the instruction; the presence of an element within this structure indicates all payments in the instruction meet the criterion.
<FederalInstructionInfo>	0..1	Aggregate	US Federal payment fields at the instruction level.

<XML Tags>	Cardinality	Datatype	Description
<OutboundPayment>	1..n	Aggregate	Payment.
<Extend>	0..n	Aggregate	Extensibility element; to be filled with custom user data.

PaymentInstructionInfo

PaymentInstructionInfo

<XML Tags>	Cardinality	Datatype	Description
<InstructionReference Number>	1	Integer	Primary key of the payment instruction; exposed as instruction reference number in disbursement dashboard User Interface.
<FormatProgramRequestID>	1	Integer	Concurrent program request ID.
<InstructionCreation Date>	1	Date	Date of the payment instruction's creation.
<MultiOrgAccessControl>	0..1	Aggregate	Multiple organizations access control information.
<InstructionStatus>	1	<Lookup>	Current status of the instruction.
<ProcessingType>	1	String	How the payment instruction will be processed. Possible values are: PRINTED and ELECTRONIC.

<XML Tags>	Cardinality	Datatype	Description
<ProcessType>	1	String	Lookup IBY_PROCESS_TYPE S include STANDARD, IMMEDIATE, and MANUAL.
<UserAssignedRefCode>	0..1	String	User (payment admin) - assigned reference code. In case of some US Federal formats this reference code is generated by the Federal Financials module and used as the payment schedule number in the formats.
<BankAssignedRefCode>	0..1	String	Bank-assigned reference code.
<PaymentSequence>	Minimum 0, maximum 3	Aggregate	Store the last used payment sequence numbers. Used to support sequential numbering of payment records across payment files in some EFT formats.
<BankInstruction>	Minimum 0, maximum 2	<Lookup with Format Value>	Bank instructions at the instruction level.
<BankInstructionDetails>	0..1	String	Additional free text bank instruction details.
<PaymentTextMessage>	Minimum 0, maximum 2	String	Payment instruction level text messages. Used to send free format messages to the bank.

<XML Tags>	Cardinality	Datatype	Description
<RegulatoryReportingOption>	0..1	Enumeration	Specifies the customer's chosen declaration option. Possible values are: NO_DECLARATION, DECLARE_THROUGH_BANK, DECLARE_DIRECT_TO_CENTRAL_BANK.
<PaymentSystemAccount>	0..1	Aggregate	The payment system account to be used to send the payment instruction if applicable.
<Notes>	0..1	String	User entered notes.
<DescriptiveFlexField>	0..1	<Descriptive Flexfield>	Descriptive flexfield for the payment instruction.

MultiOrgAccessControl

MultiOrgAccessControl

<XML Tags>	Cardinality	Datatype	Description
<HasBlockedData>	1	String	Y or N flag indicating whether the payment instruction has payments that are not accessible (viewable) for the user who submitted the format program. This flag is used in the seeded Payment Instruction Register format to display a warning (some transactions are blocked) if the flag is Y. Similar usage in the seeded Payment Process Request Status Report.

PaymentSequence

PaymentSequence

<XML Tags>	Cardinality	Datatype	Description
<SequenceName>	1	String	Name of the periodic sequence.
<LastValue>	1	Integer	Last value of the periodic sequence. Stored on the account payment profile.

PaymentSystemAccount

PaymentSystemAccount

<XML Tags>	Cardinality	Datatype	Description
<AccountName>	1	String	Name of the payment system account.
<AccountSettings>	0..n	Aggregate	Name value pairs of account settings. Bank assigned identifiers are exposed through this structure.
<DescriptiveFlexField>	0..1	<Descriptive Flex Field>	Descriptive flexfield for the payment system account.

AccountSettings

AccountSettings

<XML Tags>	Cardinality	Datatype	Description
<Name>	1	String	Name of account setting.
<Value>	1	String	Value of account setting.

PaymentProcessProfile

PaymentProcessProfile

<XML Tags>	Cardinality	Datatype	Description
<PaymentProcessProfileInternalID>	1	String	Account payment process profile ID.
<PaymentProcessProfileName>	1	String	Name of the payment process profile.
<DescriptiveFlexField>	0..1	<Descriptive Flex Field>	Descriptive flexfield for the payment process profile.

PaymentFormat

PaymentFormat

<XML Tags>	Cardinality	Datatype	Description
<PaymentFormatInternalID>	1	String	User entered code for the payment format.
<PaymentFormatName>	1	String	Name of the payment process profile.
<DescriptiveFlexField>	0..1	<Descriptive Flex Field>	Descriptive flexfield for the payment process profile.

CheckFormatInfo

CheckFormatInfo

<XML Tags>	Cardinality	Datatype	Description
<PaymentDocument Name>	1	String	Paper document name. Derived from the payment_document_id of the instructions table.
<PaperStockType>	0..1	<Lookup>	Paper stock type: blank or pre-numbered.

InstructionTotals

InstructionTotals

<XML Tags>	Cardinality	Datatype	Description
<PaymentCount>	1	Integer	Total number of payments in the instruction.
<TotalPaymentAmount>	0..1	<Amount>	Total payment amount value if the instruction is created with group by currency (single currency).

InstructionGrouping

InstructionGrouping

<XML Tags>	Cardinality	Datatype	Description
<Payment Date>	0..1	Date	Payment date for all payments in the instruction.
<PaymentCurrency>	0..1	<Currency>	Currency of all payments in the instruction.
<PaymentServiceReq uestID>	0..1	String	Service request id for all payments in the instruction.
<Payer>	0..1	Aggregate	First party payer for all payments in the instruction.
<PayerOrganization>	0..1	Aggregate	First party payer organization for all payments in the instruction.
<BankAccount>	0..1	Aggregate	Internal bank account for all payments in the instruction.
<PaymentReason>	0..1	<Lookup with Format Value>	Payment reason for all payments in the instruction.
<PaymentReasonCom ments>	0..1	String	Payment reason comments for all payments in the instruction.

<XML Tags>	Cardinality	Datatype	Description
<FederalRFCIdentifier>	0..1	String	US Federal RFC Identifier applicable to all payments in the instruction. Associated with the payer bank branch party.

FederalInstructionInfo

FederalInstructionInfo

<XML Tags>	Cardinality	Datatype	Description
<TreasurySymbols>	1	Aggregate	Treasury symbols for Federal summary formats.
<ControlNumber>	1	String	Treasury assigned control number for bulk payment files.
<ECSSummaryDosSequenceNumber>	1	Integer	ECS summary dos file name sequence number. Generated by the Federal module.

TreasurySymbol

TreasurySymbols

<XML Tags>	Cardinality	Datatype	Description
<TreasurySymbol>	1	String	Treasury symbol.

<XML Tags>	Cardinality	Datatype	Description
<Amount>	1	<Amount>	The total amount in the bulk payment file under the treasury symbol.

Payment Level Elements

OutboundPayment

OutboundPayment

<XML Tags>	Cardinality	Datatype	Description
<PaymentSourceInfo>	1	Aggregate	Payment source information such as the calling product or payment function.
<PaymentNumber>	1	Aggregate	Various number assigned to the payment.
<PaymentDate>	1	Date	Payment date.
<PaymentDueDate>	0..1	Date	Date when the payment is due. Populate only if the child documents were grouped by their due dates.
<MaturityDate>	0..1	Date	The calculated maturity date on a bill payable payment (future dated payment).
<PaymentStatus>	1	<Lookup>	Current payment status.

<XML Tags>	Cardinality	Datatype	Description
<PaymentError>	1	Aggregate	Errors associated with the payment.
<Payee>	1	<3rdPartyInfo>	Recipient of the payment.
<PayeeBankAccount>	0..1	<PayeeBankAccount>	Destination bank account.
<Beneficiary>	1	Aggregate	The actual beneficiary in case of third party payments; otherwise same as the payee.
<Payer>	1	Aggregate	First party payer. This is the legal entity that is stamped on the payments. The legal entity is derived as the owner of the internal bank account of the payment.
<BankAccount>	1	Aggregate	The payer's bank account; the source of funds for making the payment.
<PaymentAmount>	1	<Amount>	Amount of the payment in the payment currency.
<PaymentAmountText>	0..1	String	The payment amount as words. Formatted according to the user language.
<PaymentMethod>	1	Aggregate	Payment method assigned to the documents that were grouped into this payment.

<XML Tags>	Cardinality	Datatype	Description
<PayAloneFlag>	0..1	String	Y or N flag indicating if the payment is created for a pay alone document.
<SettlementPriority>	0..1	<Lookup>	Instructions to the bank related to the urgency of the payment. Populated only if the child documents are grouped by the settlement priority.
<AmountWithheld>	0..1	<Amount>	Amount withheld in payment currency.
<DiscountTaken>	0..1	Aggregate	Information about discount taken on this payment.
<BankCharges>	0..1	Aggregate	Bank charge info about this payment.
<DeliveryChannel>	0..1	<Lookup with Format Value>	Instructions related to which channel should be used by the bank to deliver the payment to the payee. Populated only if the child documents are grouped by the delivery channels.
<BankInstruction>	minimum 0, maximum 2	<Lookup with Format Value>	Enumerated bank instruction information at the payment level.
<BankInstructionDetails>	0..1	String	Additional free text bank instruction details.

<XML Tags>	Cardinality	Datatype	Description
<PaymentTextMessage>	minimum 0, maximum 3	String	Text message for use in payment processing – at payment level.
<PaymentDetails>	0..1	String	Concatenated string that represents the details of the child documents.
<UniqueRemittanceIdentifier>	0..1	Aggregate	Contains remittance identifiers assigned by the payee to the child documents. Populated only if the child documents are grouped by the unique remittance identifiers.
<PaymentReason>	0..1	<Lookup with Format Value>	Used mostly for regulatory reporting purposes. Also called: payment nature, payment category code, and declaration code. Populated only if the child documents are grouped by payment reasons.
<PaymentReasonComments>	0..1	String	Free text field available for payment reason. Populated only if the child documents are grouped by payment reason comments.

<XML Tags>	Cardinality	Datatype	Description
<RemittanceMessage>	minimum 0, maximum 3	String	Remittance free text message entered at the document level. Populated only if the child documents are grouped by remittance messages.
<RegulatoryReporting>	0..1	Aggregate	Regulatory reporting information.
<FederalPaymentInfo>	0..1	Aggregate	Additional information at the payment level for US Federal Financials.
<DocumentPayableCount>	1	Integer	Number of documents in this payment.
<DocumentPayable>	1..n	Aggregate	Document payable.
<DescriptiveFlexField>	0..1	<Descriptive Flex Field>	Descriptive flex field for the payment.
<Extend>	0..n	Aggregate	Extensibility element; to be filled with custom user data.

PaymentSourceInfo

PaymentSourceInfo

<XML Tags>	Cardinality	Datatype	Description
<ApplicationInternalID>	1	Integer	ID of the originating application.

<XML Tags>	Cardinality	Datatype	Description
<ApplicationName>	1	String	Short name of the originating application.
<PaymentServiceRequestID>	1	String	Calling application payment process request ID.
<FunctionalCategory>	1	<Lookup>	The functional category of the payment.
<PayerOrganization>	0..1	Aggregate	First party payer organization.
<EmployeePaymentFlag>	0..1	String	Y or N flag indicating if the payment is for an employee; that is, the payee is an employee.

PayerOrganization

PayerOrganization

<XML Tags>	Cardinality	Datatype	Description
<OrganizationInternalID>	1	Integer	HR_ALL_ORGANIZATION_UNITS. ORGANIZATION_ID.
<OrganizationType>	0..1	<Lookup>	Organization type. Depending on the calling product, the organization types can be either operating unit or legal entity.
<OrganizationName>	0..1	String	Organization name.

PaymentNumber

PaymentNumber

<XML Tags>	Cardinality	Datatype	Description
<DocCategory>	0..1	String	AOL sequence attribute. This sequence is generated by Oracle Payments, not calling products.
<SequenceName>	0..1	String	AOL sequence attribute. This sequence is generated by Oracle Payments, not calling products.
<SequenceValue>	0..1	String	AOL sequence attribute. This sequence is generated by Oracle Payments, not calling products.
<PaymentReferenceNumber>	1	Integer	Corresponding to the payment reference number field in disbursement dashboard user interface.
<CheckNumber>	0..1	Integer	Check number; used in printed payment instructions.

PaymentError

PaymentError

<XML Tags>	Cardinality	Datatype	Description
<ErrorType>	1	<Lookup>	Type of error. Values from the lookup IBY_TRANSACTION_ERROR_TYPES include VALIDATION and BANK.
<ErrorCode>	1	String	Error code for failed transaction. If ERROR_TYPE is BANK, this is a bank-specific error code.
<ErrorMessage>	0..1	String	Error message for failed transaction. If ERROR_TYPE is BANK, this is a bank-specific error message.
<ErrorDate>	1	Date	Date time of the error.
<ErrorStatus>	1	<Lookup>	Whether the error is still active, that is, the error still applies, passed, or overridden.

Payee

Payee

<XML Tags>	Cardinality	Datatype	Description
<PartyInternalID>	1	Integer	Payee party ID.

<XML Tags>	Cardinality	Datatype	Description
<PartyNumber>	1	String	Party number.
<Name>	1	String	Full name of the party.
<TaxRegistrationNumber>	0..1	String	Tax number of party. Use this element if the data was previously stored in PO_VENDORS. VAT_REGISTRATION_NUM in 11i.
<LegalEntityRegistrationNumber>	0..1	String	LE registration number. Use this element if the data was previously stored in PO_VENDORS. NUM_1099 in 11i.
<PartyDescriptiveFlexField>	0..1	<Descriptive Flex Field>	Descriptive flex field for the party record.
<AlternateName>	0..1	String	Alternate name of the party.
<SupplierNumber>	0..1	String	Supplier number.
<SupplierDescriptiveFlexField>	0..1	<Descriptive Flex Field>	Descriptive flex field for the supplier record.
<SupplierSiteDescriptiveFlexField>	0..1	<Descriptive Flex Field>	Descriptive flex field for the supplier site record.

<XML Tags>	Cardinality	Datatype	Description
<Address>	0..1	<Address>	Third party payee's address. Generated based on the remit-to location id of payment. The remit-to location is a hard coded grouping rule for creating payments from documents payable.
<ContactInfo>	0..1	<ContactInfo>	Contact of the third party.
<FirstPartyReference>	1	String	Payee assigned ID for the first party payer.

PayeeBankAccount

PayeeBankAccount

<XML Tags>	Cardinality	Datatype	Description
<BankAccountInternalID>	0..1	Integer	Primary key of the IBY (external) bank account table.
<BankName>	0..1	String	Name of the bank.
<AlternateBankName>	0..1	String	Alternate name of the bank.
<BankNumber>	0..1	String	Bank Number. From the HZ organization profile linked to the bank party.
<BranchInternalID>	0..1	Integer	Party ID of the bank branch party.

<XML Tags>	Cardinality	Datatype	Description
<BranchName>	0..1	String	Name of the bank branch.
<AlternateBranchName>	0..1	String	Alternate name of the bank branch.
<BranchNumber>	0..1	String	Bank branch number. From the HZ organization profile linked to the bank branch party.
<BankAccountName>	0..1	String	Name of the bank account.
<AlternateBankAccountName>	0..1	String	Alternate name of the bank account.
<BankAccountNumber>	1	String	The bank account number. Formatted for electronic transmission. In case the account number for transmission is the same as user entered, the same value will be saved to the two columns and extracted.
<UserEnteredBankAccountNumber>	1	String	The bank account number as user entered. This field should be used for formats that are intended for human read, such as checks.
<SwiftCode>	0..1	String	SWIFT code of the bank account. Stored in HZ_CONTACT_POINTS associated with bank branch.

<XML Tags>	Cardinality	Datatype	Description
<IBANNumber>	0..1	String	IBAN of the bank account.
<CheckDigits>	0..1	String	Check digits of the bank account number.
<BankAccountType>	0..1	<Lookup>	The account type.
<BankAccountCurrency>	0..1	<Currency>	Currency by which accounts funds are denominated.
<BankAddress>	0..1	<Address>	Primary address of the bank branch.
<BankContact>	0..1	<ContactInfo>	Contact at the bank.
<PrimaryOwner>	1	Aggregate	Primary owner of bank account in case of single/joint owned; payment factor party in case of factor bank account.
<FactorAccount>	0..1	<Lookup>	Y or N flag indicating if the bank account is a factor account.
<DescriptiveFlexField>	0..1	<Descriptive Flex Field>	Descriptive flexfield for the bank account.
<BranchDescriptiveFlexField>	0..1	<Descriptive Flex Field>	Descriptive flexfield for the bank branch. From the HZ party of the bank branch.

PrimaryOwner

PrimaryOwner

<XML Tags>	Cardinality	Datatype	Description
<Name>	1	String	Party name of the primary owner or payment factor.

Beneficiary

Beneficiary

<XML Tags>	Cardinality	Datatype	Description
<Name>	1	String	Party name of the actual beneficiary or payee.

Payer

Payer

<XML Tags>	Cardinality	Datatype	Description
<PartyInternalID>	1	Integer	The party that is linked to the legal entity.
<PartyNumber>	1	String	Party number.
<Name>	1	String	Full name of the party.
<TaxRegistrationNumber>	0..1	String	Tax registration number.

<XML Tags>	Cardinality	Datatype	Description
			Use this element if the data was previously stored in FINANCIALS_SYSTEM_PARAMETERS_ALL. VAT_REGISTRATION_NUM in 11i.
<LegalEntityRegistrationNumber>	0..1	String	Legal entity registration number. Use this element if the data was previously stored in AP_REPORTING_ENTITIES_ALL. TAX_IDENTIFICATION_NUM in 11i.
<DescriptiveFlexField>	0..1	<Descriptive Flex Field>	Descriptive flexfield for the party record.
<LegalEntityInternalID>	1	Integer	Legal entity stamped on the payments. Owner of the transactions.
<LegalEntityName>	1	String	Legal entity name.
<Address>	1	<Address>	Legal entity address.
<ContactInfo>	0..1	<ContactInfo>	Contact at the first party.
<DescriptiveFlexField>	0..1	<Descriptive Flex Field>	Descriptive flexfield for the legal entity record.

BankAccount

BankAccount

<XML Tags>	Cardinality	Datatype	Description
<BankAccountInternalID>	1	Integer	Primary key of the CE (internal) bank account table.
<BankName>	0..1	String	Bank name.
<BankNumber>	0..1	String	Bank number. From the HZ organization profile linked to the bank party.
<BranchInternalID>	1	Integer	Party id of the bank branch party.
<BranchName>	0..1	String	Name of the bank branch.
<AlternateBranchName>	0..1	String	Alternate name of the bank branch.
<BranchNumber>	0..1	String	Number of the bank branch. From the HZ organization profile linked to the bank branch party.
<BankAccountName>	1	String	Name of the bank account.
<AlternateBankAccountName>	0..1	String	Alternate name of the bank account.

<XML Tags>	Cardinality	Datatype	Description
<BankAccountNumber>	1	String	The bank account number. Formatted for electronic transmission. In case the account number for transmission is the same as user entered, the same value will be saved to the two columns and extracted.
<UserEnteredBankAccountNumber>	1	String	The bank account number as user entered. This field should be used for formats that are intended for human read, such as checks.
<SwiftCode>	0..1	String	SWIFT code of the bank account. Stored in HZ_CONTACT_POINTS associated with bank branch.
<IBANNumber>	0..1	String	IBAN of the bank account.
<CheckDigits>	0..1	String	Check digits of the bank account number.
<BankAccountType>	1	<Lookup>	The account type.
<BankAccountCurrency>	1	<Currency>	Currency by which accounts funds are denominated.
<BankAddress>	1	<Address>	Primary address of the bank branch.
<BankContact>	0..1	<ContactInfo>	Contact at the bank.

<XML Tags>	Cardinality	Datatype	Description
<DescriptiveFlexField>	0..1	<Descriptive Flex Field>	Descriptive flexfield for the bank account.
<BranchDescriptiveFlexField>	0..1	<Descriptive Flex Field>	Descriptive flexfield for the bank branch. From the HZ party of the bank branch.
<FederalBankAccountInfo>	0..1	Aggregate	Additional information for use by US Federal customers.
<EFTUserNumber>	0..1	Aggregate	EFT numbers assigned by the bank to the user.

FederalBankAccountInfo

FederalBankAccountInfo

<XML Tags>	Cardinality	Datatype	Description
<FederalRFCIdentifier>	1	String	Identifier of the US Treasury Regional Finance Center (RFC) from which the disbursement should originate for federal agencies. Stored in HZ_CODE_ASSIGNMENTS with CLASS_CATEGORY as RFC_IDENTIFIER, associated with the bank branch party.
<FederalAgencyLocationCode>	1	String	US Federal Agency Location Code used by federal agency.

<XML Tags>	Cardinality	Datatype	Description
<FederalAbbreviatedAgencyCode>	1	String	US Federal Abbreviated Agency Code.
<FederalEmployerIdentificationNumber>	1	String	US Federal Employer ID.

EFTUserNumber

EFTUserNumber

<XML Tags>	Cardinality	Datatype	Description
<AccountLevelEFTNumber>	1	String	Account-level EFT user number.
<BranchLevelEFTNumber>	1	String	Branch-level EFT user number.

PaymentMethod

PaymentMethod

<XML Tags>	Cardinality	Datatype	Description
<PaymentMethodInternalID>	1	String	Internal id of the payment method.
<PaymentMethodName>	1	String	Describes the payment method.

<XML Tags>	Cardinality	Datatype	Description
<PaymentMethodFormatValue>	1	String	This field is currently mapped to the same DB column as PaymentMethodInternalID, pending the change to add it's own column.
<DescriptiveFlexField>	0..1	<Descriptive Flex Field>	Descriptive flexfield for the payment method.

DiscountTaken

DiscountTaken

<XML Tags>	Cardinality	Datatype	Description
<Amount>	1	<Amount>	Amount of discount taken. In payment currency.
<DiscountDate>	1	Date	Discount date. Available at document payable level only.

BankCharges

BankCharges

<XML Tags>	Cardinality	Datatype	Description
<BankChargeBearer>	1	<Lookup>	Bank charge bearer lookup.

<XML Tags>	Cardinality	Datatype	Description
<Amount>	0..1	<Amount>	Bank charge amount in payment currency. The amount is only available at the payment level.

UniqueRemittanceIdentifier

UniqueRemittanceIdentifier

<XML Tags>	Cardinality	Datatype	Description
<Number>	1	String	Unique remittance identifier.
<CheckDigit>	0..1	String	Check digit.

RegulatoryReporting

RegulatoryReporting

<XML Tags>	Cardinality	Datatype	Description
<DeclarationFlag>	1	String	Y or N flag indicating whether this payment needs to be reported to the central bank.
<Amount>	1	<Amount>	Declared payment amount in the declaration currency.

FederalPaymentInfo

FederalPaymentInfo

<XML Tags>	Cardinality	Datatype	Description
<FederalAllotmentCode>	1	String	US Federal Allotment code.
<FederalOffsetEligibilityFlag>	1	String	Y or N flag indicating if the payment can be used for US Treasury offset.
<FederalAccountingSymbol>	1	String	Payment level accounting symbol for US Federal SPS formats.

Document Payable Level Elements

DocumentPayable

DocumentPayable

<XML Tags>	Cardinality	Datatype	Description
<DocumentNumber>	1	Aggregate	Contains document identifier elements, typically specified by the calling application.
<PONumber>	0..1	String	PO number(s) to which this document line is matched.
<DocumentStatus>	1	<Lookup>	Lookup aggregate of the document payable status.

<XML Tags>	Cardinality	Datatype	Description
<DocumentError>	1	Aggregate	Errors associated with this document.
<DocumentDate>	1	Date	Document date.
<DocumentCreationDate>	0..1	Date	Date of the document's creation.
<PaymentDueDate>	0..1	Date	Date when payment is due to the payee.
<DocumentType>	0..1	<Lookup>	Code for the document type. The document type is a concatenation of the calling product doc types, such as STANDARD, CREDIT, DEBIT, INTEREST, PREPAYMENT, MIXED, or AWT for Oracle Payables and BAT for Cash Management Bank Account Transfer.
<DocumentDescription>	0..1	String	Free text description of the document payable.
<ExpenseReportCreditCardNumber>	0..1	String	Credit card number for expense report with Oracle iExpense.

<XML Tags>	Cardinality	Datatype	Description
<EmployeePaymentFlag>	0..1	String	Y or N flag indicating if the payment is for an employee, that is, the payee is an employee. This is a hard-coded grouping rule for creating payments, where the documents of a payment must all have the same Employee Payment Flag value – either Y or N.
<TotalDocumentAmount>	0..1	<Amount>	Total amount specified on the document in the document currency.
<PaymentAmount>	1	<Amount>	Payment amount in payment currency.
<PayAloneFlag>	0..1	String	Y or N flag indicating if the document should be paid by its own payment.
<SettlementPriority>	0..1	<Lookup>	Urgency of the payment.
<AmountWithheld>	0..1	<Amount>	Amount withheld in payment currency.
<DiscountTaken>	0..1	Aggregate	Document-level discounts.
<BankCharges>	0..1	Aggregate	Bank charge info for this document.
<DeliveryChannel>	0..1	<Lookup with Format Value>	Delivery channel.

<XML Tags>	Cardinality	Datatype	Description
<PaymentReason>	0..1	<Lookup with Format Value>	Used mostly for regulatory reporting purposes. Also called: payment nature, payment category code, or declaration code.
<PaymentReasonComments>	0..1	String	Free text field available for payment reason.
<RemittanceMessage>	minimum 0, maximum 3	String	Remittance message – usually used in remittance advice.
<Charge>	0..1	Aggregate	Charges. Used for remittance purposes.
<TotalDocumentTax Amount>	0..1	<Amount>	Total tax amount for the doc payable. (Not pro-rated for payment schedule). For remittance purposes.
<CreditAmountApplied>	0..1	<Amount>	Total credit amount (credit memo and debit memo) applied for the document payable. (Not pro-rated for payment schedule). Currently for Brazil only.
<InterestAmountApplied>	0..1	<Amount>	Total interest amount applied for the document payable. (Not pro-rated for payment schedule). Currently for Brazil only.

<XML Tags>	Cardinality	Datatype	Description
<InterestRate>	0..1	Number	Interest rate based on which this interest invoice was created.
<DocumentPayableLine>	0..n	Aggregate	Line items associated with the document.
<Extend>	0..n	Aggregate	Extensibility element; to be filled with custom user data.
<DescriptiveFlexField>	0..1	<Descriptive Flex Field>	Descriptive flexfield for the payment method.
<SrcDocDescriptiveFlexField>	0..1	<Descriptive Flex Field>	Descriptive flexfield for the payment method.

DocumentNumber

DocumentNumber

<XML Tags>	Cardinality	Datatype	Description
<DocumentInternalID Segment1~5>	1	String	Calling application's unique document payable identifiers. For Oracle Payables, segment1 maps to ap_inv_selection_criteria_all.checkrun_id, segment2 to invoice_id, segment3 to payment_num of the payment schedule.

<XML Tags>	Cardinality	Datatype	Description
<ApplicationInternalID>	1	Integer	Identifier of the application which submitted the document for payment.
<ApplicationDocumentSubType>	0..1	String	Document type assigned to this document by the application; used as a distinguisher for applications with multiple document sources.
<ApplicationUniqueInternalID>	1	String	Application provided identifier; will be unique per application and application document sub type.
<ReferenceNumber>	0..1	String	User-assigned reference code for this document; for example: invoice number, expense report number.
<BankAssignedReferenceCode>	0..1	String	Bank-assigned reference code for this document.
<UniqueRemittanceIdentifier>	0..1	Aggregate	Contains remittance identifiers assigned by the payee to the child document.
<DocCategory>	0..1	String	AOL sequence attribute. The sequence is from the calling product.

<XML Tags>	Cardinality	Datatype	Description
<SequenceName>	0..1	String	AOL sequence attribute. The sequence is from the calling product.
<SequenceValue>	0..1	String	AOL sequence attribute. The sequence is from the calling product.

DocumentError

DocumentError

<XML Tags>	Cardinality	Datatype	Description
<ErrorType>	1	<Lookup>	Type of error. Values from the lookup IBY_TRANSACTION_ERROR_TYPES include VALIDATION and BANK.
<ErrorCode>	1	String	Error code for failed transaction. If ERROR_TYPE is BANK, this is a bank-specific error code.
<ErrorMessage>	0..1	String	Error message for failed transaction. If ERROR_TYPE is BANK, this is a bank-specific error message.
<ErrorDate>	1	Date	Date time of the error.

<XML Tags>	Cardinality	Datatype	Description
<ErrorStatus>	1	<Lookup>	Whether the error is still active, that is, the error still applies, passed, or overridden.

Charge

Charge

<XML Tags>	Cardinality	Datatype	Description
<ChargePerType>	0..n		Amount per charge type (for future use).
<TotalDocumentChargeAmount>	1	<Amount>	Total charges for the document (in document currency). Note this is the document level total; it's not pro-rated for payment schedules.

Document Line Level Elements

DocumentPayableLine

DocumentPayableLine

<XML Tags>	Cardinality	Datatype	Description
<LineNumber>	1	Integer	Line item number.

<XML Tags>	Cardinality	Datatype	Description
<PONumber>	0..1	String	PO number(s) to which this document line is matched. If multiple POs, value will be comma delimited.
<LineType>	1	<Lookup>	Code for the line item type; for example: item, tax, or freight.
<LineDescription>	0..1	String	Description of line item.
<LineGrossAmount>	0..1	<Amount>	Total amount for document line in document currency.
<UnitPrice>	0..1	Number	Price per unit of this item.
<Quantity>	0..1	Integer	Number of line item units.
<UnitOfMeasure>	0..1	<Lookup>	Unit of measure lookup code.
<Tax>	0..1	Aggregate	Tax.
<Extend>	0..n	Aggregate	Extensibility element; to be filled with custom user data.

Tax

Tax

<XML Tags>	Cardinality	Datatype	Description
<Tax>	1	String	Tax.

<XML Tags>	Cardinality	Datatype	Description
<TaxRate>	1	Number	Tax rate.

Common Elements

Currency

Currency

<XML Tags>	Cardinality	Datatype	Description
<Code>	1	String	Currency code; acts as foreign key into table FND_CURRENCIES.
<Name>	0..1	String	Currency name.
<Symbol>	0..1	String	Currency symbol.
<MinAccountableUnit>	0..1	Integer	Minimum accountable units.
<Precision>	0..1	Integer	Precision of the currency, in terms of its smallest subunits.

Amount

Amount

<XML Tags>	Cardinality	Datatype	Description
<Value>	1	Real	Scale of the amount.

<XML Tags>	Cardinality	Datatype	Description
<Currency>	1	Aggregate	Currency of the amount.

Lookup

Lookup

<XML Tags>	Cardinality	Datatype	Description
<Code>	1	String	Lookup code.
<Meaning>	1	String	Code meaning.

Lookup with Format Value

Lookup with Format Value

<XML Tags>	Cardinality	Datatype	Description
<Code>	1	String	Code value. Joins to the primary keys of Oracle Payments' setup tables.
<Meaning>	1	String	Code meaning.
<FormatValue>	1	String	Value required by the payment format that uses this lookup.

Descriptive Flex Field

DescriptiveFlexField

<XML Tags>	Cardinality	Datatype	Description
<AttributeCategory>	1	String	Descriptive flex field attribute category.
<Attribute>	0..n	String	Descriptive flex field attribute.

Extend

Extend

<XML Tags>	Cardinality	Datatype	Description
<ExtendName>	1	String	Extensibility name.
<ExtendValue>	1	String	Extensibility value.

Payer and Payee Address

Address

<XML Tags>	Cardinality	Datatype	Description
<AddressInternalID>	1	Integer	The data source of the address data.
<AddressLine1>	1	String	Address line 1.
<AddressLine2>	0..1	String	Address line 2.
<AddressLine3>	0..1	String	Address line 3.

<XML Tags>	Cardinality	Datatype	Description
<AddressLine4>	0..1	String	HZ address only. Null for HR address.
<City>	0..1	String	City.
<County>	0..1	String	County.
<State>	0..1	String	State.
<Country>	1	String	ISO 3166 two digit country code.
<ISO3DigitCountry>	1	String	ISO 3166 three digit country code.
<CountryName>	1	String	Country name.
<PostalCode>	0..1	String	Postal code.
<PreFormattedConcat enatedAddress>	1	String	Pre-formatted concatenated address. Address fields are concatenated with the space as separator. Address fields are ordered according to country address format styles. This version of per- formatted address can be used in EFT formats.
<PreFormattedMailin gAddress>	1	String	Pre-formatted mailing address with embedded line breaks. Generated by TCA with the Postal Address TCA style. Address fields are ordered according to country address format conventions.

<XML Tags>	Cardinality	Datatype	Description
<AddressName>	1	String	HR location code for payer; vendor site name or party site name for payee.
<AlternateAddressName>	0..1	String	For payee address only. Alternate vendor site code.

Bank Account Address

Internal and external bank account addresses come from the HZ locations table. The internal (payer) bank account address is obtained based on the `int_bank_branch_location_id` column in the payments table; while the external (payee) bank account address is based on `ext_bank_branch_location_id`.

The elements are similar to those of payer/payee addresses, except they do not have the `AddressName` and `AlternateAddressName` elements.

Contact Information Elements

ContactInfo

<XML Tags>	Cardinality	Datatype	Description
<ContactName>	0..1	Aggregate	The contact's personal name.
<ContactLocators>	1	Aggregate	Various means by which the contact may be located or reached.

ContactName

<XML Tags>	Cardinality	Datatype	Description
<FirstName>	1	String	Contact person's first name.
<LastName>	1	String	Contact person's last name.

ContactLocators

<XML Tags>	Cardinality	Datatype	Description
<PhoneNumber>	0..1	String	Contact phone number.
<FaxNumber>	0..1	String	Contact fax machine number.
<EmailAddress>	0..1	String	Contact e-mail address.
<Website>	0..1	String	Contact website.

Index

A

- access control
 - action pages, 3-3
 - Disbursement System Options Setup page, 3-3
 - overview, 3-1
 - payment process concurrent programs, 3-3
 - read-only pages, 3-2
 - setup pages, 3-3
 - understanding, 3-1
- account options
 - defining, 8-17
- ACK, 8-26
- acknowledgements from payment systems
 - receiving, 6-6
- acknowledgment parser
 - developing, 8-24
 - how it works, 8-23
 - seeding, 8-23
- acquiring bank, 3-4, 3-11
- action pages
 - access control, 3-3
- APIs
 - credit card validation, F-7
 - implementing electronic commerce applications, F-1
 - Java for electronic commerce application, F-12
 - overview of Oracle Payments, F-1
 - payment instrument registration, F-2
 - payment processing, F-3
 - PL/SQL for electronic commerce applications, F-21

- risk management, F-6
 - status update, F-9
 - understanding, 3-3
- Application Programming Interface (API), 3-4, 3-4
- assigning
 - operating units, 6-11
 - validations, 6-4
- assigning responsibilities and roles to users
 - configuring Oracle Payments, 4-6
- authorizations
 - creating and testing, 7-1

B

- BankAccountBatchACK, 8-32
- bank account transfers
 - understanding, 3-15
- BankAccountTrxnACK, 8-29
- bank instruction codes
 - setting up, 5-7
- banks, 3-11
- BatchACK, 8-29
- business payment models
 - flexible support, 1-4

C

- check printing process
 - user-friendly, 1-6
- closing transaction batches, 3-17
- common elements, J-21
 - address, J-23
 - bank account, J-25

- contact information, J-24
- credit card, J-29
- debit card, J-30
- document line, J-33
- generic, J-21
- party, J-31
- common functionality
 - for funds capture and funds disbursement, 3-1
- completing
 - header, 6-10
- components
 - payment system integration, 8-1
- conditions
 - routing rules, 3-22
- configuration
 - centralized, 1-10
- configuring
 - Oracle Payments sample servlet, 4-15
 - Step 19. Configuring the ECAApp Servlet, 6-2
 - tunneling, 4-11
 - XML framework, 4-18
- configuring Oracle Payments
 - assigning responsibilities and roles to users, 4-6
- considerations
 - security, F-26
- country-specific validations, B-3
- creating
 - a wallet file, 4-8
 - payment methods, 6-4
 - risk formula, 6-11
 - routing rules, 6-12
- creating and testing
 - authorizations, 7-1
- credit card
 - security features, 1-8
- CreditCardBatchACK, 8-31
- credit card brands
 - setting up, 6-12
- credit card encryption, 3-6
- credit card owners
 - verifying, 4-9
- credit card processing, 3-11
- credit card transactions
 - understanding, 3-11
- CreditCardTrxnACK, 8-29
- custom disbursement validation sets

- creating, 8-32
- custom funds capture validation sets
 - creating, 8-46
- customizations
 - search pages, I-1

D

- default payment systems
 - selecting, 6-11
- defining
 - account options, 8-17
 - payment system, 8-16
 - wallet file password, 4-8
- delivery channel codes
 - setting up, 5-8
- direct debits, 1-8
- disbursement process
 - flexible, 1-4
- disbursement system options
 - setting up, 5-21
- document level elements, J-19
- document line level elements, J-20
- documents payable
 - understanding, 3-26

E

- ECAApp servlet
 - configuring, 6-2
- electronic
 - payment processing, 1-5
- electronic commerce
 - API, 3-4
 - applications, 3-21
- electronic commerce application
 - Java APIs, F-12
- electronic commerce applications
 - PL/SQL APIs, F-21
- electronic commerce applications APIs
 - implementing, F-1
- employing funds capture payment methods
 - source product, 6-3
- encrypting
 - payment instruments, 4-8
- encryption, 1-9
- error handling
 - during payment processing, E-1

- extract
 - funds capture, J-1
- extract
 - funds disbursement, K-1
- extract components, J-2
- extract formatter
 - how it works, 8-19
- extract generator
 - how it works, 8-19
- extract structure
 - funds capture, J-1
 - overview, K-1

F

- features
 - funds capture , 1-7
 - funds disbursement, 1-4
- field-installable servlets, 3-4
- first party payees
 - setting up, 6-9
- formats
 - developing a format template, 8-18
 - seeding a format template, 8-18
 - setting up, 4-10
 - using, 8-18
- formatting framework
 - advanced and configurable, 1-2
- funds capture
 - developing a custom payment system
 - integration for, 8-3
 - extract, J-1
 - features, 1-7
 - planning, 2-6
 - understanding, 3-8
- funds capture and funds disbursement
 - common functionality, 3-1
- funds capture bank account transfers
 - understanding, 3-15
- funds capture extract, J-2
- funds capture instruction elements, J-3
- funds capture payment methods
 - setting up, 6-3
- funds capture process profile
 - understanding, 3-8
- funds capture process profiles
 - setting up, 6-4

- funds capture side
 - payment systems, 4-14
- funds disbursement
 - developing a custom payment system
 - integration for, 8-12
 - extract, K-1
 - features, 1-4
 - understanding, 3-26
- funds disbursement payment methods
 - setting up, 5-2
- funds disbursement processing
 - configurability, 1-5
- funds disbursement side
 - payment systems, 4-14

G

- gateway-model and processor-model payment systems
 - understanding, 3-16

H

- header
 - completing, 6-10
- host-based merchant, 3-17

I

- implementing
 - electronic commerce applications APIs, F-1
- implementing Oracle Payments
 - applications conveying language information, 2-5
 - bank account transfer operations for funds capture, 2-8
 - disbursement payment methods, 2-9
 - format of the NLS_LANG parameter, 2-6
 - format of the response body data from payment system servlets, 2-6
 - funds capture planning, 2-6
 - funds disbursement planning, 2-9
 - Oracle Payments use of NlsLang, 2-5
 - organizing settlement batches, 2-6
 - payment instrument registration APIs, 2-7
 - payment processing APIs, 2-7
 - presenting information in different languages, 2-4

- risk factors, 2-8
- risk management APIs, 2-8
- security needs, 2-1
- shared planning, 2-1
- supporting credit card brands, 2-6
- using APIs for funds capture, 2-7
- using formats, 2-4
- using funds capture or funds disbursement functionality, 2-2
- using National Language Support (NLS), 2-5
- using payment systems, 2-2
- using source products, 2-3
- integrated
 - with Oracle E-Business Suite, 1-8
- integrating
 - with non-Oracle Applications, G-1
 - with other Oracle Applications, 3-9
- introduction
 - Oracle Payments, 1-1
- issuing bank, 3-11

J

- Java APIs
 - for electronic commerce application, F-12

L

- library
 - payment formats, 1-7
- loading
 - risky instruments, 6-12

M

- masking, 1-9
 - payment instruments, 4-9
- merchant
 - host-based, 3-17
 - terminal-based, 3-17
- Multi-Organization Access Control (MOAC), 3-2, 3-6

N

- non-Oracle Applications
 - integrating with, G-1

O

- offline and online payments
 - understanding, 3-17
- online authorization
 - sending and receiving, 6-5
- online debits
 - sending and receiving, 6-6
- online payer verifications
 - sending and receiving, 6-5
- online payer verifications, authorizations, and debits
 - sending and receiving, 6-5
- operating units
 - assigning, 6-11
- Oracle Applications
 - integrating with, 3-9
- Oracle E-Business Suite
 - integrated, 1-8
- Oracle Payments
 - general features, 1-2
 - global features, 1-6
 - introduction, 1-1
- Oracle Payments sample servlet
 - configuring, 4-15
- Oracle Payments servlets
 - payment system servlets, 3-4
- Oracle Payments system partner, 3-4, 3-4
- Oracle Receivables, 3-20
- Oracle XML Publisher templates
 - setting up, 4-9
- order level elements, J-9
- overview
 - Oracle Payments APIs, F-1
 - payment system integration model, 8-1
 - setup tasks for funds capture, 6-1
 - setup tasks for funds disbursement, 5-1
 - shared setup tasks for funds capture and funds disbursement, 4-1
- overview
 - extract structure, K-1

P

- pages
 - Disbursement System Options Setup page, 3-3
- parameters
 - transmission protocols, C-1
- payee account level elements, J-7

- payees
 - creating, 3-9
 - multiple, 3-9
 - understanding, 3-9
- payer notifications
 - of settlement, 1-11
- payer notifications to payers
 - sending, 6-6
- payment data repository
 - secure, 1-3
- payment formats
 - Argentina, A-4
 - Belgium, A-5
 - Brazil, A-7
 - Chile, A-8
 - Colombia, A-9
 - common disbursement, A-1
 - Finland, A-11
 - France, A-13
 - Germany, A-16
 - Italy, A-22
 - Japan, A-24
 - library, 1-7, 1-11
 - Netherlands, A-25
 - New Zealand, A-26
 - Norway, A-27
 - Poland, A-28
 - Portugal, A-30
 - Spain, A-32
 - Sweden, A-37
 - Switzerland, A-41
 - United Kingdom, A-43
 - United States, A-43
 - United States Federal, A-47
 - viewing, A-1
- payment function, 3-2
- payment grouping
 - understanding, 3-29
- payment instructions
 - understanding, 3-29
- payment instruments
 - encrypting, 4-8
 - masking, 4-9
- payment method defaulting rules
 - setting up, 5-6
- payment methods
 - creating, 6-4
 - flexible, 1-8
- payment methods (funds capture)
 - understanding, 3-8
- payment methods (funds disbursement)
 - understanding, 3-27
- payment process concurrent programs
 - access control, 3-3
- payment processing
 - electronic, 1-5
 - error handling, E-1
 - user interface, 1-5, 1-10
- payment processors, 3-4
- payment process profiles
 - setting up, 5-9
 - understanding, 3-27
- payment process requests
 - understanding, 3-28
- payment reason codes
 - setting up, 5-9
- payments
 - scheduled, 1-11
 - understanding, 3-26
- payment system, 3-4
 - attributes, 8-16
 - defining, 8-16
 - settings required, 4-14
- payment system accounts
 - specifying, 6-9
- payment system API, 3-4
- payment system integration
 - developing a custom payment system
 - integration for funds disbursement, 8-12
- payment system integration components
 - developing, 8-1
- payment system integration for funds capture
 - developing, 8-3
- payment system integration model
 - overview, 8-1
- payment systems
 - basic authentication, 3-7
 - funds capture side, 4-14
 - funds disbursement side, 4-14
 - setting up, 4-13
- payment systems and payment system accounts
 - selecting, 6-10
- payment system servlet communication
 - setting up SSL security, 4-17

- payment system servlets, 3-4, 3-4
- PINless debit card
 - transactions, 1-8
- PINless debit card transactions
 - understanding, 3-14
- PL/SQL APIs
 - for electronic commerce applications, F-21
- planning
 - funds capture, 2-6
- prerequisites
 - setting up first party payees, 6-10
 - setting up formats, 4-10
 - setting up funds capture process profiles, 6-4
 - setting up payment systems, 4-13
- processing credit cards, 3-11
- processor, 3-11
 - credit card, 3-11
- processor and gateway model payment systems
 - Oracle Payments supports, 1-9
- profile options
 - settings, H-1
- protocols
 - AS2 Send, C-9
 - File Transfer Protocol for Static File Names, C-7
 - First Data North Authorization Specification 10/24/02 Socket, C-4
 - First Data North Magnetic Media Specification 2003.1 Acknowledgment FTP Get, C-6
 - First Data North Magnetic Media Specification 2003.1 Settlement Batch FTP Put, C-5
 - Http(s) POST Request, C-1
 - Local File System Delivery, C-11
 - Oracle Payments Tunneling, C-2
 - Paymentech Batch Specification 2.1.0 Acknowledgment FTP Get, C-3
 - Paymentech Online Specification 7.2 Socket, C-3
 - Secure File Transfer, C-8
- purchase card processing transaction detail
 - selecting, 6-11
- purchase cards
 - data levels, 3-13
 - processing, 3-14
 - understanding, 3-12
- purpose
 - setting up credit card brands, 6-12

- setting up first party payees, 6-10
- setting up formats, 4-10
- setting up funds capture payment methods, 6-4
- setting up funds capture process profiles, 6-5
- setting up Oracle XML Publisher templates, 4-9
- setting up payment systems, 4-14
- setting up transmission configurations, 4-11

R

- read-only pages
 - access control, 3-2
- receiving
 - acknowledgements from payment systems, 6-6
- remittance advice
 - reporting, 1-7
- returns, 3-12, 3-17
- risk factors
 - shipped with Oracle Payments, 3-19
- risk formula
 - creating, 6-11
- risk management
 - AVS, 3-19, 3-20
 - factors in Oracle Receivables, 3-20
 - test scenarios, D-3
 - understanding, 3-18
- risk threshold
 - specifying, 6-11
- risky instruments
 - loading, 6-12
- routing
 - Oracle Payments, 3-21
 - to multiple payment systems for funds capture, 1-10
- routing and operation
 - Oracle Payments, 3-21
- routing rules
 - components, 3-21
 - conditions, 3-22
 - creating, 6-12
 - how routing works, 3-22

S

- search pages

- customizations, I-1
- security
 - considerations, F-26
 - firewall protection, 3-6
 - IP address restriction, 3-7
 - secure socket layer (SSL), 3-7
 - understanding, 3-5
- security features
 - credit card, 1-8
- security policy, 1-9
- selecting
 - default payment systems, 6-11
 - payment systems and payment system accounts, 6-10
 - purchase card processing transaction detail, 6-11
- sending
 - payer notifications to payers, 6-6
- sending and receiving
 - online authorization, 6-5
 - online debits, 6-6
 - online payer verifications, 6-5
 - online payer verifications, authorizations, and debits, 6-5
 - settlements, 6-6
- servlet
 - configuring Oracle Payments sample, 4-15
- servlet communication, 3-6
- servlets, 3-4
 - understanding, 3-4
- settings
 - profile options, H-1
 - required by payment system, 4-14
- setting up
 - bank instruction codes, 5-7
 - configuring XML framework, 4-18
 - creating Oracle Payments users, 4-5
 - credit card brands, 6-12
 - delivery channel codes, 5-8
 - disbursement system options, 5-21
 - first party payees, 6-9
 - formats, 4-10
 - funds capture payment methods, 6-3
 - funds capture process profiles, 6-4
 - funds disbursement payment methods, 5-2
 - Oracle XML Publisher templates, 4-9
 - payment method defaulting rules, 5-6
 - payment process profiles, 5-9
 - payment reason codes, 5-9
 - payment systems, 4-13
 - system security options, 4-7
 - the wallet, 4-7
 - transmission configurations, 4-11
 - validations, 4-10
- settlement, 3-12, 3-17
 - payer notifications, 1-11
- settlement batch, 3-12
- settlement grouping
 - specifying, 6-7
- settlement limits
 - specifying, 6-9
- settlements
 - sending and receiving, 6-6
- setup checklist
 - shared setup tasks, 4-4
- setup pages
 - access control, 3-3
- setup tasks
 - shared
 - setting up SSL security for payment system servlet communication, 4-17
- setup tasks for funds capture
 - overview, 6-1
- setup tasks for funds disbursement
 - overview, 5-1
- shared planning
 - implementing Oracle Payments, 2-1
- shared setup tasks
 - setting up SSL security for payment system servlet communication, 4-17
 - setup checklist, 4-4
- shared setup tasks for funds capture and funds disbursement
 - overview, 4-1
- source product
 - employing funds capture payment methods, 6-3
- specifying
 - payment system accounts, 6-9
 - risk threshold, 6-11
 - settlement grouping, 6-7
 - settlement limits, 6-9
 - supported capabilities, 4-14
- specifying or generating
 - payment process profiles, 5-9
 - payment reason codes, 5-9
 - payment systems, 4-13
 - system security options, 4-7
 - the wallet, 4-7
 - transmission configurations, 4-11
 - validations, 4-10

- system key file location, 4-8
- supported capabilities
 - specifying, 4-14
- system key file location
 - specifying or generating, 4-8
- system security options
 - setting up, 4-7

T

- terminal-based and host-based merchants
 - understanding, 3-17
- terminal-based merchant, 3-17
- testing transactions
 - Transaction Testing tab, 7-1
- test scenarios
 - risk management, D-3
- transaction reporting
 - understanding, 3-24
- transactions
 - PINless debit card, 1-8
- Transaction Testing tab
 - testing transactions, 7-1
- transmission
 - electronic, 1-3
 - understanding, 3-4
- transmission configuration, 3-5
- transmission configurations
 - setting up, 4-11
- transmission function
 - developing, 8-20
 - how it works, 8-19
- transmission protocol, 3-5
 - parameters, C-1
 - seeding, 8-21
- TrxnACK, 8-27
- tunneling
 - configuring, 4-11

U

- understanding
 - access control, 3-1
 - APIs, 3-3
 - credit card transactions, 3-11
 - documents payable, 3-26
 - funds capture, 3-8
 - funds capture bank account transfers, 3-15

- funds capture process profile, 3-8
- funds disbursement, 3-26
- gateway-model and processor-model payment systems, 3-16
- offline and online payments, 3-17
- payees, 3-9
- payment grouping, 3-29
- payment instructions, 3-29
- payment methods (funds capture), 3-8
- payment methods (funds disbursement), 3-27
- payment process profiles, 3-27
- payment process requests, 3-28
- payments, 3-26
- PINless debit card transactions, 3-14
- purchase cards, 3-12
- risk management, 3-18
- security, 3-5
- servlets, 3-4
- terminal-based and host-based merchants, 3-17
- transaction reporting, 3-24
- transmission, 3-4
- validations, 3-30
- user interface
 - payment processing, 1-5, 1-10
- users
 - creating for Oracle Payments, 4-5

V

- validation model
 - flexible, 1-2
- validations
 - Argentina, B-3
 - assigning, 6-4
 - Austria, B-3
 - Belgium, B-4
 - Brazil, B-6
 - Chile, B-7
 - Colombia, B-7
 - country-specific, B-3
 - Czechoslovakia, B-7
 - Finland, B-7
 - France, B-7
 - Germany, B-7
 - Hungary, B-10
 - Japan, B-11

- Luxembourg, B-11
- Netherlands, B-12
- Oracle e-Commerce Gateway Format, B-1
- other, B-1
- Poland, B-12
- Portugal, B-15
- setting up, 4-10
- Sweden, B-15
- Switzerland, B-17
- Turkey, B-18
- understanding, 3-30
- United States, B-18
- United States Federal, B-19
- validation sets
 - creating custom disbursement, 8-32
 - creating custom funds capture, 8-46
- verifying
 - credit card owners, 4-9
- voids, 3-12, 3-17

W

- wallet
 - setting up, 4-7
- wallet file
 - creating, 4-8
- wallet file password
 - defining, 4-8

X

- XML framework
 - configuring, 4-18

