Oracle® Receivables

API User Notes

Release 11*i*

Part No. B14166-01

August 2004

ORACLE®

Oracle Receivables API User Notes, Release 11*i*

Part No. B14166-01

Primary Authors: Ramakant Alat, Jon Beckett, Anuj Kumar, Kunal Mahajan, Ajay Pandit, Jyoti Pandey, Obaidur Rashid, Bijoy Sarkar

Contributor: Kristin Penaskovic

Contents

2 Credit Memo Approval and Creation API User Notes

3 Credit Request Creation API User Notes

4 Deposit API User Notes

5 Invoice Creation API User Notes

6 Prepayments API User Notes

7 Receipt API User Notes

8 Revenue Adjustment API User Notes

Send Us Your Comments

Oracle Receivables API User Notes, Release 11*i*

Part No. B14166-01

Oracle welcomes your comments and suggestions on the quality and usefulness of this document. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most?

If you find any errors or have any other suggestions for improvement, please indicate the document title and part number, and the chapter, section, and page number (if available). You can send comments to us in the following ways:

- Electronic mail: appsdoc_us@oracle.com
- FAX: (650) 506-7200 Attn: Oracle Applications Documentation Manager
- Postal service:
  Oracle Corporation
  Oracle Applications Documentation Manager
  500 Oracle Parkway
  Redwood Shores, CA 94065
  USA

If you would like a reply, please give your name, address, telephone number, and (optionally) electronic mail address.

If you have problems with the software, please contact your local Oracle Support Services.

Preface

Welcome to the *Oracle Receivables API User Notes*, Release 11*i*.

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

- The principles and customary practices of your business area.
- Oracle Receivables.

  If you have never used Oracle Receivables, Oracle suggests you attend one or more of the Oracle Applications training classes available through Oracle University.
- The Oracle Applications graphical user interface.

  To learn more about the Oracle Applications graphical user interface, read the *Oracle Applications User's Guide*.

See Other Information Sources for more information about Oracle Applications product information.

How To Use This Guide

The Oracle Receivables API User Notes contains the information you need to understand and use public APIs in Oracle Receivables.

This guide provides technical specifications and guidelines for using the following Receivables APIs:

- Adjustment API
- Credit Memo Approval and Creation API
- Credit Request Creation API
- Deposit API
- Invoice Creation API
- Prepayments API
- Receipt API
- Revenue Adjustment API

This guide is organized for easy access to the following information:

- Major Features
- Solution Outlines
- API Usage
- Error Messages

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For additional information, visit the Oracle Accessibility Program Web site at http://www.oracle.com/accessibility/

Accessibility of Code Examples in Documentation

JAWS, a Windows screen reader, may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, JAWS may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

Other Information Sources

You can choose from many sources of information, including documentation, training, and support services, to increase your knowledge and understanding of Oracle Receivables.

If this guide refers you to other Oracle Applications documentation, use only the Release 11*i* versions of those guides.

Online Documentation

All Oracle Applications documentation is available online (HTML or PDF).

- **PDF Documentation -** See the Documentation CD provided with each release for current PDF documentation for your product. This Documentation CD is also available on Oracle*MetaLink* and is updated frequently.
- **Online Help -** The Oracle Receivables APIs do not have an end user guide, or separate online HTML help.
- **11i Release Content Document -** Refer to the Release Content Document for new features listed release. The Release Content Document is available on Oracle*MetaLink*.
- **About document -** Refer to the About document for patches that you have installed to learn about new documentation or documentation patches that you can download. The new About document is available on Oracle*MetaLink*.

Related Guides

Oracle Receivables shares business and setup information with other Oracle Applications products. Therefore, you may want to refer to other guides when you set up and use Oracle Receivables.

You can read the guides online by choosing Library from the expandable menu on your HTML help window, by reading from the Oracle Applications Document Library CD included in your media pack, or by using a Web browser with a URL that your system administrator provides.

If you require printed guides, you can purchase them from the Oracle Store at http://oraclestore.oracle.com.

Guides Related to All Products

Oracle Applications User's Guide

This guide explains how to enter data, query, run reports, and navigate using the graphical user interface (GUI). This guide also includes information on setting user profiles, as well as running and reviewing reports and concurrent processes.

You can access this user's guide online by choosing "Getting Started with Oracle Applications" from any Oracle Applications help file.

Guides Related to This Product

Oracle Receivables User Guide

This user guide explains how to use Oracle Receivables, including how to set up your system, create transactions and receipts, and run reports.

Installation and System Administration

Oracle Applications Concepts

This guide provides an introduction to the concepts, features, technology stack, architecture, and terminology for Oracle Applications Release 11*i*. It provides a useful first book to read before an installation of Oracle Applications. This guide also introduces the concepts behind Applications-wide features such as Business Intelligence (BIS), languages and character sets, and Self-Service Web Applications.

Installing Oracle Applications

This guide provides instructions for managing the installation of Oracle Applications products. In Release 11*i*, much of the installation process is handled using Oracle Rapid Install, which minimizes the time to install Oracle Applications and the Oracle technology stack by automating many of the required steps. This guide contains instructions for using Oracle Rapid Install and lists the tasks you need to perform to finish your installation. You should use this guide in conjunction with individual product user guides and implementation guides.

Oracle Applications Implementation Wizard User Guide

If you are implementing more than one Oracle product, you can use the Oracle Applications Implementation Wizard to coordinate your setup activities. This guide describes how to use the wizard.

Upgrading Oracle Applications

Refer to this guide if you are upgrading your Oracle Applications Release 10.7 or Release 11.0 products to Release 11*i*. This guide describes the upgrade process and lists database and product-specific upgrade tasks. You must be either at Release 10.7 (NCA, SmartClient, or character mode) or Release 11.0, to upgrade to Release 11*i*. You cannot upgrade to Release 11*i* directly from releases prior to 10.7.

"About" Document

For information about implementation and user documentation, instructions for applying patches, new and changed setup steps, and descriptions of software updates, refer to the "About" document for your product. "About" documents are available on Oracle*MetaLink* for most products starting with Release 11.5.8.

Maintaining Oracle Applications

Use this guide to help you run the various AD utilities, such as AutoUpgrade, AutoPatch, AD Administration, AD Controller, AD Relink, License Manager, and others. It contains how-to steps, screenshots, and other information that you need to run the AD utilities. This guide also provides information on maintaining the Oracle applications file system and database.

Oracle Applications System Administrator's Guide

This guide provides planning and reference information for the Oracle Applications System Administrator. It contains information on how to define security, customize menus and online help, and manage concurrent processing.

Oracle Alert User's Guide

This guide explains how to define periodic and event alerts to monitor the status of your Oracle Applications data.

Oracle Applications Developer's Guide

This guide contains the coding standards followed by the Oracle Applications development staff and describes the Oracle Application Object Library components that are needed to implement the Oracle Applications user interface described in the *Oracle Applications User Interface Standards for Forms-Based Products*. This manual also provides information to help you build your custom Oracle Forms Developer forms so that the forms integrate with Oracle Applications.

Oracle Applications User Interface Standards for Forms-Based Products

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

Other Implementation Documentation

Oracle Applications Product Update Notes

Use this guide as a reference for upgrading an installation of Oracle Applications. It provides a history of the changes to individual Oracle Applications products between Release 11.0 and Release 11*i*. It includes new features, enhancements, and changes made to database objects, profile options, and seed data for this interval.

Oracle Workflow Administrator's Guide

This guide explains how to complete the setup steps necessary for any Oracle Applications product that includes workflow-enabled processes, as well as how to monitor the progress of runtime workflow processes.

Oracle Workflow Developer's Guide

This guide explains how to define new workflow business processes and customize existing Oracle Applications-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 Oracle Applications 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 Applications Flexfields Guide

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

Oracle eTechnical Reference Manuals

Each eTechnical Reference Manual (eTRM) contains database diagrams and a detailed description of database tables, forms, reports, and programs for a specific Oracle Applications product. This information helps you convert data from your existing applications, integrate Oracle Applications data with non-Oracle

applications, and write custom reports for Oracle Applications products. Oracle eTRM is available on Oracle*Metalink*

Oracle Applications Message Manual

This manual describes all Oracle Applications messages. This manual is available in HTML format on the documentation CD-ROM for Release 11*i*.

Training and Support

Training

Oracle offers a complete set of training courses to help you and your staff master Oracle Receivables and reach full productivity quickly. These courses are organized into functional learning paths, so you take only those courses appropriate to your job or area of responsibility.

You have a choice of educational environments. You can attend courses offered by Oracle University at any one of our many education centers, you can arrange for our trainers to teach at your facility, or you can use Oracle Learning Network (OLN), Oracle University's online education utility. In addition, Oracle training professionals can tailor standard courses or develop custom courses to meet your needs. For example, you may want to use your organization structure, terminology, and data as examples in a customized training session delivered at your own facility.

Support

From on-site support to central support, our team of experienced professionals provides the help and information you need to keep Oracle Receivables working for you. This team includes your technical representative, account manager, and Oracle's large staff of consultants and support specialists with expertise in your business area, managing an Oracle server, and your hardware and software environment.

Do Not Use Database Tools to Modify Oracle Applications Data

Oracle STRONGLY RECOMMENDS that you never use SQL*Plus, Oracle Data Browser, database triggers, or any other tool to modify Oracle Applications 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 Applications data, you risk destroying the integrity of your data and you lose the ability to audit changes to your data.

Because Oracle Applications tables are interrelated, any change you make using Oracle Applications can update many tables at once. But when you modify Oracle Applications data using anything other than Oracle Applications, 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 Applications.

When you use Oracle Applications to modify your data, Oracle Applications automatically checks that your changes are valid. Oracle Applications 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.

About Oracle

Oracle develops and markets an integrated line of software products for database management, applications development, decision support, and office automation, as well as Oracle Applications, an integrated suite of more than 160 software modules for financial management, supply chain management, manufacturing, project systems, human resources and customer relationship management.

Oracle products are available for mainframes, minicomputers, personal computers, network computers and personal digital assistants, allowing organizations to integrate different computers, different operating systems, different networks, and even different database management systems, into a single, unified computing and information resource.

Oracle is the world's leading supplier of software for information management, and the world's second largest software company. Oracle offers its database, tools, and applications products, along with related consulting, education, and support services, in over 145 countries around the world.

Your Feedback

Thank you for using Oracle Receivables and this user guide.

Oracle values your comments and feedback. In this guide is a reader's comment form that you can use to explain what you like or dislike about Oracle Receivables or this user guide. Mail your comments to the following address or call us directly at (650) 506-7200.

Oracle Applications Documentation Manager
Oracle Corporation
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Or, send electronic mail to appsdoc_us@oracle.com.

1

Adjustment API User Notes

Overview

This document outlines the use of the Adjustment API. This API allows users to create, approve, update, and reverse adjustments for invoices using simple calls to PL/SQL functions.

The Adjustment API is not intended to replace the existing Adjustment form, Adjustment Approval form, or the batch Auto-Adjust program.

Note: The Adjustment API requires the following receivable activity setup: the GL Account Source must be *Activity* and the Tax Code Source must be *None*.

You can access the API in two ways:

- With standard PL/SQL servers-side routine calls
- Through Forms, using the capability of Forms6 to have a procedure as its underlying base table.

Before you begin....

Initialization of ARP_STANDARD and ARP_GLOBAL

Custom code that uses AR or HZ APIs will set the ORG_ID via dbms_application_info.set_client_info() and then call the APIs. The APIs in turn might access either ARP_STANDARD and ARP_GLOBAL, which initialize the global variables that are used across Oracle Receivables when the package is first called. Most of these global variable values are organization dependent, and the first such call sets the global variables based on the current ORG_ID.

If additional custom code then changes the ORG_ID via another call to dbms_application_info.set_client_info(), then the ORG context changes, ***but the ARP_STANDARD and ARP_GLOBAL context does not.***

In such cases, you should explicitly re-initialize the global variables by a call to these two public procedures:

1. ARP_GLOBAL.INIT_GLOBAL: For setting public variables in ARP_GLOBAL.
2. ARP_STANDARD.INIT_STANDARD: For setting public variables in ARP_STANDARD.

Major Features

Flexibility

The Adjustment API has a defaulting mechanism for input parameters. This lets you create, approve, update, and reverse adjustments while passing a minimal number of API parameters.

Modular Approach

The API has been designed in a highly modular fashion, resulting in code that is:

- Easy to understand
- Easy to maintain
- Easy to expand

Error Handling

The Adjustment API provides an error-handling and error-reporting mechanism whereby all the errors encountered in the defaulting and validation phases are reported and put on the message stack. The relevant entity handler is called only if there are no errors reported during the defaulting and validation phases.

Debugging

Users must enable debugging by calling the routine arp_standard.enable_file_debug.

Arp_standard.enable_file_debug requires 2 parameters: path_name and file_name.

The path name can be identified by using the following select statement:

```
select value from v$parameter where name = 'utl_file_dir';
```

The file name can be any name, chosen by the user.

Example:

```
arp_standard.enable_file_debug ('/sqlcom/log', 'txt.log');
```

Solution Outline

To create, modify, approve, or reverse adjustments, use the following routines:

- *Ar_Adjust_pub.Create_Adjustment*: Use this routine to create an adjustment for an invoice.
- *Ar_Adjust_pub.Modify_Adjustment*: Use this routine to modify an adjustment's status, comments, and reason code. NOTE: if the existing status of the adjustment is A or R, then it cannot be modified.
- *Ar_Adjust_pub.Approve_Adjustment*: Use this routine to approve an adjustment.
- *Ar_Adjust_pub.Reverse_Adjustment*: Use this routine to reverse an adjustment.

The Adjustment API is not intended to replace the existing Adjustment form, Adjustment Approval form, or the batch Auto-Adjust program.

Note: The Adjustment API requires the following receivable activity setup: the GL Account Source must be *Activity* and the Tax Code Source must be *None*.

Modular Approach

To modularize the Adjustment API, the basic structure of the API has been divided into three steps. The API:

1. Validates information the user has entered.
2. Prepares data for the entity handlers by defaulting any values that might be needed.
3. Calls the entity handler to create/modify/approve/reverse the adjustment.

This results in code that is easy to understand and maintain. Any new functionality can be added using a simple code plug-in at any of the three steps.

Exception Handling and Result Messages

The Adjustment APIs return three types of information to their calling programs:

- A return status
- Messages describing the operations performed and/or errors encountered by the APIs
- Other output values

Return Status

The return status (p_return_status) of the API informs the caller about the result of the operation (or operations) performed by the API. The different possible values for an API return status are:

- Success (FND_API. G_RET_STS_SUCCESS)
- Error (FND_API. G_RET_STS_ERROR)
- Unexpected error (FND_API. G_RET_STS_UNEXP_ERROR)

Success

A success return status means that the API was able to perform all the operations requested by its caller. A success return status could also be accompanied by messages in the API message list.

Error

An error return status means that the API failed to perform some or all of the operations requested by its caller. An error return status is usually accompanied by messages describing the error (or errors) and suggesting how to fix them.

In most cases, end users can take corrective actions to fix regular expected errors such as missing attributes or invalid date ranges.

Unexpected error

An unexpected error status means that the API has encountered an error condition it did not expect or could not handle. In this case, the API is unable to continue with its regular processing. Examples of such errors are: irrecoverable data inconsistency errors, memory errors, and programming errors (such as attempting a division by zero).

In most cases, end users cannot correct unexpected errors. System administrators or application developers generally must fix these errors.

Messages

The APIs put result messages into a message list. Programs calling these APIs can then retrieve the messages from the list and process them by either issuing them, loading them in a database table, or writing them to a log file.

Messages are stored in an encoded format to enable the API callers to determine message names by using the standard functions provided by the message dictionary. The encoded format also allows users to store these messages in database tables and to report off these tables in different languages.

The API message list must be initialized every time a program calls an API. API callers have the choice of either calling the message list utility function FND_MSG_PUB.Initialize or having the API initialize the message list automatically by setting the p_init_msg_list parameter to TRUE.

The program calling the API can retrieve messages from the message stack using the existing FND API functions FND_MSG_PUB.Count_Msg and FND_MSG_PUB.Get.

API Usage

The following table shows Standard API parameters common to all the routines in the Adjustment API:

Parameter	Type	Data-type	Required	Default Value	Description
p_api_version	IN	NUMBER	Yes		Used to compare version numbers of incoming calls to its current version number.
p_init_msg_list	IN	VARCHAR2		FND_API.G_ FALSE	Allows API callers to request that the API does initialization of the message list on their behalf.
p_commit	IN	VARCHAR2		FND_API.G_ FALSE	Used by API callers to ask the API to commit on their behalf.
p_validation_level	IN	NUMBER		FND_API.G_ VALID_LEVEL_ FULL	Not currently for use by the user. Allow this parameter to default.
p_return_status	OUT	VARCHAR2			Represents the API overall return status. For possible values, see *Error Handling* on page 1-4.
p_msg_count	OUT	NUMBER			Number of messages in the API message list
p_msg_data	OUT	VARCHAR2			This is the message in encoded format if p_msg_count=1

Ar_Adjust_pub.Create_Adjustment

Description

Use this routine to create adjustments to invoices. The API returns the Out parameter p_new_adjust_id, which represents the newly-created adjustment id. The following is a breakdown of parameters for this routine, divided according to parameter type:

Input Parameters

Standard API parameters: 4

Create Adjustment parameters: 6 required parameters (might vary depending on the adjustment type).

Output Parameters

Standard API parameters: 3

Create Adjustment parameters: 2

Since the Create Adjustment API allows users to pass the adjustment record type to the procedure, it is not recommended that users enter values for unnecessary fields. These fields could be populated for internal use only.

The following table lists parameters that pertain specifically to the Create Adjustment routine:

Parameter	Type	Data-type	Required	Default Value	Description
p_adj_rec.type	IN	VARCHAR2	Yes		The type of adjustment to be created. Possible Values: 'INVOICE', 'LINE', 'TAX', 'FREIGHT', 'CHARGES', 'FINCHRG'
p_adj_ rec.payment_ schedule_id	IN	NUMBER	Yes		Payment Schedule id of the transaction for which the transaction is to be created.
p_adj_rec.amount	IN	NUMBER	Yes/No		If the adjustment type is any other value than 'INVOICE' then this is a required field. The amount indicates the amount to be adjusted.
p_adj_ rec.customer_trx_ line_id	IN	NUMBER	Yes/No		If the adjustment type is 'LINE' then the customer_trx_line_id indicates the line to be adjusted. For all the other adjustment types the value is not required.
p_adj_ rec.receivables_trx_ id	IN	NUMBER	Yes		The id of the activity name (from ar_ receivables_trx) should be passed.
p_adj_rec.code_ combination_id	IN	NUMBER	No		The code combination id is not required. If the value is not passed then the default is the code combination id specified in the receivables_trx_id record. If the value passed is not the same as the code_ combination_id and the profile option of allow override of the default activity is set to 'N' then this would error out.
p_adj_rec.apply_ date	IN	DATE	Yes		The apply date should be equal to or greater than the transaction date.

Parameter	Type	Data-type	Required	Default Value	Description
p_adj_rec.gl_date	IN	DATE	Yes		The gl date should be equal to or greater than the transaction gl date, and the date should be from the open/future period.
p_adj_rec.reason_code	IN	VARCHAR2	No		The reason code should a valid reason code in ar_lookups with lookup_type = 'ADJUST_REASON'.
p_adj_rec.comments	IN	VARCHAR2	No		The user can enter comments, up to 2000 bytes, for creating the adjustments which could be useful for the user, for future reference.
p_adj_rec.associated_cash_receipt_id	IN	NUMBER	No		The associated cash receipt id is the id of a valid cash receipt, and is to be associated with the adjustment.
p_adj_rec.ussgl_transaction_code	IN	VARCHAR2	No		The USSGL transaction code should be a valid USSGL transaction code in gl_ussgl_transaction_codes.
p_adj_rec.created_from	IN	VARCHAR2	Yes		Some value that indicates to the user that it was created through the Adjustment API. Eg. 'ADJ-API'
p_adj_rec.attribute_category, p_adj_rec.attribute1 - p_adj_rec.attribute15	IN	VARCHAR2	No		This attribute_category and the attribute1 through attribute15 can be entered if the user want to enter the details of the descriptive flex field for the adjustment.
p_adj_rec.adjustment_id	IN		No. Entered values will be overwritten		
p_adj_rec.acctd_amount	IN		No. Entered values will be overwritten		
p_adj_rec.gl_posted_date	IN		No. Entered values will be overwritten		

Parameter	Type	Data-type	Required	Default Value	Description
p_adj_rec.set_of_books_id	IN		No. Entered values will be overwritten		
p_adj_rec.adjustment_type	IN		No. Entered values will be overwritten		
p_adj_rec.status	IN		No. Entered values will be overwritten		
p_adj_rec.line_adjusted	IN		No. Entered values will be overwritten		
p_adj_rec.freight_adjusted	IN		No. Entered values will be overwritten		
p_adj_rec.tax_adjusted	IN		No. Entered values will be overwritten		
p_adj_rec.receivables_chages_adjusted	IN		No. Entered values will be overwritten		
p_adj_rec.batch_id	IN		No. Entered values will be overwritten		

Parameter	Type	Data-type	Required	Default Value	Description
p_adj_ rec.customer_trx_ id	IN		No. Entered values will be overwritten		
p_adj_ rec.subsequent_ trx_id	IN		No. Entered values will be overwritten		
p_adj_ rec.chargeback_ customer_trx_id	IN		No. Entered values will be overwritten		
p_adj_ rec.distribution_ set_id	IN		No. Entered values will be overwritten		
p_adj_ rec.associated_ application_id	IN		No. Entered values will be overwritten		
p_adj_ rec.automatically_ generated	IN		No. Entered values will be overwritten		
p_adj_rec.postable	IN		No. Entered values will be overwritten		
p_adj_ rec.approved_by	IN		No. Entered values will be overwritten		

Parameter	Type	Data-type	Required	Default Value	Description
p_adj_ rec.adjustment_ number	IN		No. Entered values will be overwritten		
p_adj_rec.doc_ sequence_value	IN		No. Entered values will be overwritten		
p_adj_rec.doc_ sequence_id	IN		No. Entered values will be overwritten		
p_adj_rec.posting_ control_id	IN		No. Entered values will be overwritten		
p_adj_rec.last_ updated_by	IN		No. Entered values will be overwritten		
p_adj_rec.last_ updated_date	IN		No. Entered values will be overwritten		
p_adj_rec.last_ updated_login	IN		No. Entered values will be overwritten		
p_adj_rec.created_ by	IN		No. Entered values will be overwritten		

Parameter	Type	Data-type	Required	Default Value	Description
p_adj_rec.creation_date	IN		No. Entered values will be overwritten		
p_adj_rec.program_application_id	IN		No. Entered values will be overwritten		
p_adj_rec.program_id	IN		No. Entered values will be overwritten		
p_adj_rec.program_update_date	IN		No. Entered values will be overwritten		
p_adj_rec.request_id	IN		No.		
p_chk_approval_limits	IN	VARCHAR2	No.	FND_API.G_TRUE	This value can be set to 'F' if the adjusted amount should not be validated against the users approval limit.
p_move_deferred_tax	IN	VARCHAR2	No.	Y	This parameter is only used for BR.
p_check_amount	IN	VARCHAR2	No.	FND_API.G_TRUE	This value should never be set to 'F'. It is used for some internal logic.
p_new_adjust_number	OUT	ar_adjustment.adjustment_number%type			If the adjustment is created successfully, then this parameter will contain the value of the new adjustment number.
p_new_adjust_id	OUT	ar_adjustment.adjustment_id%type			If the adjustment is created successfully, then this parameter will contain the value of the new adjustment id.
p_called_from	IN	VARCHAR2	No	NULL	This flag is only used for BR.

Note: If the user passes values for any parameter not reported in the table above, then those values will be ignored and will not show up in the record.

Default values for API parameters derive from the following:

- Values of the other parameters in the API call
- Values set in the ar_system_parameters table entered through the System Options form
- Relevant profile option values

Depending on the user's particular business needs, the minimum number of parameters required to create an adjustment may vary.

Validation of the parameters passed:

All the parameters that are passed to the API are validated, and if any of the required fields are missing or invalid, then the API returns an error message. A list of possible error messages appears in *Messages* on page 1-29.

Example

The following is the simplest test case for creating an adjustment.

Objective:

To create an adjustment, passing the minimum number of parameters.

Entered parameters:

p_adj_rec.type = 'INVOICE',

p_adj_rec.payment_schedule_id = 22222,

p_adj_rec.receivables_trx = 15,

p_adj_rec.apply_date = to_date('12-FEB-00', 'DD-MON-RR'),

p_adj_rec.gl_date = to_date('12-FEB-00', 'DD-MON-RR'),

p_adj_rec.created_from = 'ADJ-API'

Call to the API:

```
AR_ADJUST_PUB.Create_Adjustment(
```

```
p_api_name              =>   'AR_ADJUST_PUB',
p_api_version          =>   1.0,
p_msg_count            =>   msg_count ,
p_msg_data              =>   msg_data,
p_return_status       =>   return_status,
p_adj_rec                    =>   adj_rec,
p_new_adjust_number    =>   new_adj_num,
p_new_adjust_id    =>   new_adj_id );
```

Result:

Creates an adjustment, passing two standard required parameters and six adjustment record related parameters.

Ar_Adjust_pub.Approve_Adjustment

Description

Use this routine to approve an adjustment. The following is a breakdown of parameters for this routine, divided according to parameter type:

Input Parameters

Standard API parameters: 4

Approve Adjustment parameters: 1 required parameter

Output Parameters

Standard API parameters: 3

Although the Approve Adjustments API allows users to pass the adjustment record type to the procedure, all the values are overwritten by the values in the existing adjustment record except for the status and gl_date.

The following table shows parameters that pertain specifically to the Approve Adjustment routine.

> **Note:** If required parameters are not passed in a call to this API, then the call will fail. If values are not required, then the values for those fields will be copied from the existing values of the adjustment.

Parameter	Type	Data-type	Required	Default Value	Description
p_old_adjust_id	IN	NUMBER	Yes		The id of the adjustment that needs to be approved.
p_adj_rec.type	IN	VARCHAR2	No		
p_adj_ rec.payment_ schedule_id	IN	NUMBER	No		
p_adj_rec.amount	IN	NUMBER	No		
p_adj_ rec.customer_trx_ line_id	IN	NUMBER	No		

Parameter	Type	Data-type	Required	Default Value	Description
p_adj_ rec.receivables_trx_ id	IN	NUMBER	No		
p_adj_rec.code_ combination_id	IN	NUMBER	No		
p_adj_rec.apply_ date	IN	DATE	No		
p_adj_rec.gl_date	IN	DATE	No	GL date of adjustment	The GL date should be entered if it is going to be different from the one in the old adjustment.
p_adj_rec.reason_ code	IN	VARCHAR2	No		
p_adj_ rec.comments	IN	VARCHAR2	No		
p_adj_ rec.associated_ cash_receipt_id	IN	NUMBER	No		
p_adj_rec.ussgl_ transaction_code	IN	VARCHAR2	No		
p_adj_rec.created_ from	IN	VARCHAR2	No		
p_adj_ rec.attribute_ category, p_adj_ rec.attribute1 - p_ adj_rec.attribute15	IN	VARCHAR2	No		
p_adj_ rec.adjustment_id	IN		No		
p_adj_rec.acctd_ amount	IN		No		
p_adj_rec.gl_ posted_date	IN		No		
p_adj_rec.set_of_ books_id	IN		No		

Parameter	Type	Data-type	Required	Default Value	Description
p_adj_ rec.adjustment_ type	IN		No		
p_adj_rec.status	IN		No	'A' if the status is null.	Possible Value: 'A' which indicates Approval
p_adj_rec.line_ adjusted	IN		No		
p_adj_rec.freight_ adjusted	IN		No		
p_adj_rec.tax_ adjusted	IN		No		
p_adj_ rec.receivables_ chages_adjusted	IN		No		
p_adj_rec.batch_id	IN		No		
p_adj_ rec.customer_trx_ id	IN		No		
p_adj_ rec.subsequent_ trx_id	IN		No		
p_adj_ rec.chargeback_ customer_trx_id	IN		No		
p_adj_ rec.distribution_ set_id	IN		No		
p_adj_ rec.associated_ application_id	IN		No		
p_adj_ rec.automatically_ generated	IN		No		
p_adj_rec.postable	IN		No		

Parameter	Type	Data-type	Required	Default Value	Description
p_adj_ rec.approved_by	IN		No		
p_adj_ rec.adjustment_ nu,mber	IN		No		
p_adj_rec.doc_ sequence_value	IN		No		
p_adj_rec.doc_ sequence_id	IN		No		
p_adj_rec.posting_ control_id	IN		No		
p_adj_rec.last_ updated_by	IN		No		
p_adj_rec.last_ updated_date	IN		No		
p_adj_rec.last_ updated_login	IN		No		
p_adj_rec.created_ by	IN		No		
p_adj_rec.creation_ date	IN		No		
p_adj_ rec.program_ application_id	IN		No		
p_adj_ rec.program_id	IN		No		
p_adj_ rec.program_ update_date	IN		No		
p_adj_rec.request_ id	IN		No		
p_chk_approval_ limits	IN	VARCHAR2	No	FND_ API.G_ TRUE	This value can be set to 'F' if the adjusted amount should not be validated against the users approval limit.

Parameter	Type	Data-type	Required	Default Value	Description
p_move_deferred_tax	IN	VARCHAR2	No	Y	This flag is used only for Bills Receivable.

Validation of the parameters passed:

All the parameters that are passed to the API are validated, and if any required fields areNo missing or invalid, then the API returns an error message. A list of possible error messages appears in *Messages* on page 1-29.

Example

The following is the simplest test case for approving an adjustment.

Objective:

To approve an adjustment, passing the minimum number of parameters.

Entered parameters:

adjustment_id = 88888;

Call to the API:

```
AR_ADJUST_PUB.Approve_Adjustment(
    p_api_name              =>   'AR_ADJUST_PUB',
    p_api_version          =>   1.0,
    p_msg_count            =>   msg_count ,
    p_msg_data              =>   msg_data,
    p_return_status       =>   return_status,
    p_old_adjust_id      =>   adjustment_id );
```

Result:

Approves an adjustment, passing 2 standard required parameters and 1 adjustment record parameter.

Ar_Adjust_pub.Modify_Adjustment

Description

Use this routine to update an adjustment. The attributes that can be modified are comments, gl date, and status. If the status of the adjustment is already 'A' (i.e. the adjustment has already been approved), then you cannot update the adjustment. The following is a breakdown of parameters for this routine, divided according to parameter type:

Input Parameters

Standard API parameters: 4

Modify Adjustment parameters: 1 required parameter

Output Parameters

Standard API parameters: 3

Although the Modify Adjustments API allows users to pass the adjustment record type to the procedure, all the values are overwritten by the existing adjustment record except for the status, comments, and gl_date.

The following table shows parameters that pertain specifically to the Modify Adjustments routine.

> **Note:** If required parameters are not passed in a call to this API, then the call will fail. If values are not required, then the values for those fields will be copied from the existing values of the adjustment.

Parameter	Type	Data-type	Required	Default Value	Description
p_old_adjust_id	IN	NUMBER	Yes		The id of the adjustment that needs to be modified.
P_adj_rec.type	IN	VARCHAR2	No		
p_adj_rec.payment_ schedule_id	IN	NUMBER	No		
p_adj_rec.amount	IN	NUMBER	No		

Parameter	Type	Data-type	Required	Default Value	Description
p_adj_rec.customer_ trx_line_id	IN	NUMBER	No		
p_adj_rec.receivables_ trx_id	IN	NUMBER	No		
p_adj_rec.code_ combination_id	IN	NUMBER	No		
p_adj_rec.apply_date	IN	DATE	No		
p_adj_rec.gl_date	IN	DATE	No	GL date of adjustment	The GL date should be entered if the user wishes to modify the existing gl date of the adjustment.
P_adj_rec.reason_code	IN	VARCHAR2	No		
p_adj_rec.comments	IN	VARCHAR2	No		The comments should be entered if the user wishes to modify the existing comments of the adjustment.
P_adj_rec.associated_ cash_receipt_id	IN	NUMBER	No		
p_adj_rec.ussgl_ transaction_code	IN	VARCHAR2	No		
p_adj_rec.created_from	IN	VARCHAR2	No		
p_adj_rec.attribute_ category, p_adj_ rec.attribute1 - p_adj_ rec.attribute15	IN	VARCHAR2	No		
p_adj_rec.adjustment_ id	IN		No		
p_adj_rec.acctd_ amount	IN		No		
p_adj_rec.gl_posted_ date	IN		No		
p_adj_rec.set_of_ books_id	IN		No		
p_adj_rec.adjustment_ type	IN		No		

Parameter	Type	Data-type	Required	Default Value	Description
p_adj_rec.status	IN		No		The status should be entered if the user wishes to change the existing status of the adjustment. Possible Value: 'A' , 'R', 'M', 'W'
p_adj_rec.line_adjusted	IN		No		
p_adj_rec.freight_ adjusted	IN		No		
p_adj_rec.tax_adjusted	IN		No		
p_adj_rec.receivables_ chages_adjusted	IN		No		
p_adj_rec.batch_id	IN		No		
p_adj_rec.customer_ trx_id	IN		No		
p_adj_rec.subsequent_ trx_id	IN		No		
p_adj_rec.chargeback_ customer_trx_id	IN		No		
p_adj_rec.distribution_ set_id	IN		No		
p_adj_rec.associated_ application_id	IN		No		
p_adj_ rec.automatically_ generated	IN		No		
p_adj_rec.postable	IN		No		
p_adj_rec.approved_by	IN		No		
p_adj_rec.adjustment_ nu,mber	IN		No		
p_adj_rec.doc_ sequence_value	IN		No		
p_adj_rec.doc_ sequence_id	IN		No		
p_adj_rec.posting_ control_id	IN		No		

Parameter	Type	Data-type	Required	Default Value	Description
p_adj_rec.last_ updated_by	IN		No		
p_adj_rec.last_ updated_date	IN		No		
p_adj_rec.last_ updated_login	IN		No		
p_adj_rec.created_by	IN		No		
p_adj_rec.creation_date	IN		No		
p_adj_rec.program_ application_id	IN		No		
p_adj_rec.program_id	IN		No		
p_adj_rec.program_ update_date	IN		No		
p_adj_rec.request_id	IN		No		
p_chk_approval_limits	IN	VARCHAR2	No	FND_ API.G_ TRUE	This value can be set to 'F' if the adjusted amount should not be validated against the users approval limit.
p_move_deferred_tax	IN	VARCHAR2	No	Y	This flag is only used for Y.

Validations of the parameters passed:

All the parameters that are passed to the API are validated, and if any of the required fields are missing or invalid, then the API returns an error message. A list of possible error messages appears in *Messages* on page 1-29.

Example

The following is the simplest test case for updating an adjustment.

Objective:

To update an adjustment, passing the minimum number of parameters. For this example, assume the user wants to update comments.

Entered parameters:

old_adjustment_id = 88888

adj_rec.comments = 'This is the new comment'

Call to the API:

```
AR_ADJUST_PUB.Create_Adjustment(
    p_api_name              =>   'AR_ADJUST_PUB',
    p_api_version          =>   1.0,
    p_msg_count            =>   msg_count ,
    p_msg_data              =>   msg_data,
    p_return_status       =>   return_status,
    p_adj_rec                 =>   adj_rec,
    p_old_adjust_id      =>   old_adjustment_id );
```

Result:

Updates an adjustment, passing two standard required parameters and one adjustment record parameter. Users should also pass values for other parameters that the user wishes to update in the adjustment record.

Ar_Adjust_pub.Reverse_Adjustment

Description

Use this routine to reverse an adjustment. The following is a breakdown of parameters for this routine, divided according to parameter type:

Input Parameters

Standard API parameters: 4

Reverse Adjustment parameters: 1 required parameter

Output Parameters

Standard API parameters: 3

Reverse Adjustment parameters: 1

The following table shows parameters that pertain specifically to the Reverse Adjustment routine:

Parameter	Type	Data-type	Required	Default Value	Description
p_old_adjust_id	IN	NUMBER	Yes		The id of the adjustment that needs to be modified.
p_comments	IN	VARCHAR2	No		The user can specify any comments that should appear in the reverse adjustment.
p_reversal_gl_date	IN	DATE	No	Old adjustments gl date	The user can enter a gl date if he wishes it to be different from the old adjustments gl date.
p_reversal_date	IN	DATE	No	Old adjustments date	The user can enter a date if he wishes it to be different from the old adjustments date.
p_new_adj_id	OUT	NUMBER			
p_chk_approval_limits	IN	VARCHAR2	No	FND_API.G_TRUE	This value can be set to 'F' if the adjusted amount should not be validated against the users approval limit.
p_move_deferred_tax	IN	VARCHAR2	No	Y	This flag is used only for Bills Receivable.

Parameter	Type	Data-type	Required	Default Value	Description
p_called_from	IN	VARCHAR2	No	NULL	This flag is used only for Bills Receivable.

Validation of the parameters passed:

All the parameters that are passed to the API are validated, and if any of the required fields are missing or invalid, then the API returns an error message. A list of possible error messages appears in *Messages* on page 1-29.

Example

The following is the simplest test case for reversing an adjustment.

Objective:

To reverse an adjustment, passing the minimum number of parameters.

Entered parameters:

old_adjustment_id = 88888

Call to the API:

```
AR_ADJUST_PUB.Reverse_Adjustment(
    p_api_name              =>   'AR_ADJUST_PUB',
    p_api_version           =>   1.0,
    p_msg_count             =>   msg_count ,
    p_msg_data              =>   msg_data,
    p_return_status         =>   return_status,
    p_old_adjust_id         =>   old_adjustment_id
    p_new_adj_id            =>   new_adjustment_id);
```

Result:

Reverses an adjustment, passing two standard required parameters and one adjustment record parameter.

Messages

The following table describes the possible messages returned by the Adjustment API.

Message Number	Message Name	Message Description
42963	AR_AAPI_ADJ_AMOUNT_ZERO	No Adjustment amount passed.
42964	AR_AAPI_ADR_ZERO_INV	Cannot adjust, because the amount due in the Payment Schedule is zero, and the type specified is INVOICE.
42965	AR_AAPI_APPLYDATE_LT_TRXDATE	The Apply date &APPLY_DATE is earlier than the transaction date &TRX_DATE.
42966	AR_AAPI_DOC_SEQ_NOT_REQD	The specified document sequence: &DOCUMENT_SEQ is not required as the Unique Sequence Number profile option does not allow it.
42967	AR_AAPI_GLDATE_INVALID_PERIOD	The GL date: &GL_DATE is not in an open or future enterable period.
42968	AR_AAPI_GLDATE_LT_APPLYDATE	The GL date &GL_DATE is earlier than the apply date &APPLY_DATE.
42969	AR_AAPI_GLDATE_LT_TRXGLDATE	The Adjustment GL date &GL_DATE is earlier than the transaction GL date &TRX_GL_DATE.
42970	AR_AAPI_INVALID_ADJ_ID	Invalid adjustment id: &ADJUSTMENT_ID specified.
42971	AR_AAPI_INVALID_CCID	Invalid code combination id: &CCID
42972	AR_AAPI_INVALID_CREATE_STATUS	Invalid status: &STATUS passed during creation of Adjustment
42973	AR_AAPI_INVALID_DESC_FLEX	Invalid Descriptive Flexfield has been provided.
42974	AR_AAPI_INVALID_PAYSCHD	Invalid Payment Schedule Id: &PAYMENT_SCHEDULE_ID
42975	AR_AAPI_INVALID_RCVABLE_TRX_ID	Invalid receivables trx id: &RECEIVABLES_TRX_ID
42976	AR_AAPI_INVALID_REASON_CODE	The reason code &REASON_CODE is invalid.
42977	AR_AAPI_INVALID_RECEIPT_ID	Invalid Associated Cash Receipt Id &ASSOCIATED_CASH_RECEIPT_ID has been specified.

Message Number	Message Name	Message Description
42978	AR_AAPI_INVALID_TRX_CLASS	Adjustment not allowed for transactions of class: &CLASS
42979	AR_AAPI_INVALID_TYPE	Invalid type of adjustment: &TYPE
42980	AR_AAPI_INVALID_USSGL_CODE	Invalid Ussgl Transaction Code &USSGL_CODE has been specified
42981	AR_AAPI_LINE_ID_FOR_NONLINE	Customer trx line id: &CUSTOMER_TRX_LINE_ID passed for type = &TYPE
42982	AR_AAPI_NO_APPLY_DATE	Apply date has not been specified
42983	AR_AAPI_NO_APPROVAL_CODES	No valid approval codes exists for Adjustments in the Lookup table
42984	AR_AAPI_NO_CCID	No valid code combinations exist for Adjustment
42985	AR_AAPI_NO_CCID_FOR_ACTIVITY	No code combination id exists for receivables trx id: &RECEIVABLES_TRX_ID and no code combination has been specified
42986	AR_AAPI_NO_CHANGE_OR_REVERSE	No changes allowed for Adjustment with &STATUS status
42987	AR_AAPI_NO_CREATED_FROM	No values specified for the Created From attribute of the adjustment
42988	AR_AAPI_NO_CUSTOMER_ID	No customer id exists for payment schedule id: &PAYMENT_SCHEDULE_ID
42989	AR_AAPI_NO_CUSTOMER_TRX_ID	No customer trx id exists for payment schedule id: &PAYMENT_SCHEDULE_ID
42990	AR_AAPI_NO_CUSTOMER_TRX_LINEID	Invalid customer trx line id: &CUSTOMER_TRX_LINE_ID passed for customer trx id: &CUSTOMER_TRX_ID
42991	AR_AAPI_NO_GL_DATE	GL date has not been specified
42992	AR_AAPI_NO_OPEN_FUTURE_PERIOD	No valid open or future enterable GL periods exist for the set of books id &SET_OF_BOOKS_ID
42993	AR_AAPI_NO_REASON_CODES	No valid reason codes exist for Adjustments in the Lookup table
42994	AR_AAPI_NO_RECEIVABLES_TRX	No valid receivables activity exists for Adjustments
42995	AR_AAPI_NO_TYPE_CODES	No valid type codes exists for Adjustments in the Lookup table

Message Number	Message Name	Message Description
42996	AR_AAPI_NO_USSGL_CODES	No valid Ussgl Codes exist for Adjustment
42997	AR_AAPI_OVERRIDE_CCID_DISALLOW	Override Activity profile option does not allow to override the Code Combination Id provided in the Receivables Activity
42998	AR_AAPI_USSGL_CODE_DISALLOW	Ussgl Code is not allowed as the Ussgl Profile option does not allow it

2

Credit Memo Approval and Creation API User Notes

Overview

This document outlines the use of the Credit Memo Approval and Creation API. This API lets you initiate the creation of a credit memo against a specified transaction either with or without an approval process.

To create a credit memo using an existing, user-defined Credit Memo Request workflow approval process, set the p_skip_workflow_flag parameter to N. In this case, the workflow process proceeds independently of the Credit Memo Approval and Creation API. If the disputed amount of the invoice is approved, then a credit memo is automatically created.

> **Note:** You must set up the Credit Memo Request workflow before using the Credit Memo Approval and Creation API. For more information, see the *Oracle Receivables User Guide*.

To create a credit memo directly, without sending a request through the workflow approval process, set the p_skip_workflow_flag parameter to Y. If you set the p_skip_workflow_flag parameter to Y, then the Credit Memo Approval and Creation API bypasses the workflow process and calls code to automatically create the credit memo.

When you set the p_skip_workflow_flag parameter to Y, you might also have to set values for its associated parameters: p_credit_method_installments, p_credit_method_rules, and p_batch_source_name. For more information, see the description of the *AR_CREDIT_MEMO_API_PUB.Create_Request* routine on page 2-8.

You cannot use the Credit Memo Approval and Creation API to generate on-account credit memos. You must specify an existing transaction to credit.

Before you begin....

Initialization of ARP_STANDARD and ARP_GLOBAL

Custom code that uses AR or HZ APIs will set the ORG_ID via dbms_application_info.set_client_info() and then call the APIs. The APIs in turn might access either ARP_STANDARD and ARP_GLOBAL, which initialize the global variables that are used across Oracle Receivables when the package is first called. Most of these global variable values are organization dependent, and the first such call sets the global variables based on the current ORG_ID.

If additional custom code then changes the ORG_ID via another call to dbms_application_info.set_client_info(), then the ORG context changes, ***but the ARP_STANDARD and ARP_GLOBAL context does not.***

In such cases, you should explicitly re-initialize the global variables by a call to these two public procedures:

1. ARP_GLOBAL.INIT_GLOBAL: For setting public variables in ARP_GLOBAL.
2. ARP_STANDARD.INIT_STANDARD: For setting public variables in ARP_STANDARD.

Major Features

Modular Approach

The Credit Memo Approval and Creation API was designed in a highly modular fashion, to give you code that is:

- Easy to understand
- Easy to maintain
- Easy to extend

Error Handling

The Credit Memo Approval and Creation API provides an extensive error-handling and error-reporting mechanism for any errors encountered during the defaulting and the validation phases. These errors are reported and put on the message stack. The relevant entity handler is called only if no errors are reported during the defaulting and validation phases.

Debugging

The Credit Memo workflow API provides debugging messages to help you troubleshoot unexpected problems when you run the API. You must enable debugging by calling the arp_standard.enable_file_debug routine.

When you call arp_standard.enable_file_debug, you must provide two parameters: path_name and file_name.

You can determine the path name by using this select statement:

```
select value from v$parameter where name = 'utl_file_dir';
```

The file name can be any name that you choose.

Example:

```
arp_standard.enable_file_debug ('/sqlcom/log', 'txt.log');
```

Solution Outline

To initiate the Credit Memo Request workflow process by making a credit memo workflow request, call the AR_CREDIT_MEMO_API_PUB.create_request routine. To view the status of an existing request, call the AR_CREDIT_MEMO_API_PUB.Get_Request_Status routine.

Modular Approach

The Credit Memo Approval and Creation API has a three-part structure. The API:

1. Validates information that the user has entered.
2. Prepares data for the entity handlers by defaulting any values that might be needed.
3. Initiates the Credit Memo Request workflow process.

This three-part, modular structure lets you add new functionality with a simple code plug-in at any of the three parts.

Exception Handling and Result Messages

The Credit Memo Approval and Creation API returns three types of information to the calling program:

- An overall status report
- Messages that describe the operations performed or errors encountered by the API
- The output, a newly-created credit memo request ID

Return Status

You can view the results of the API's operation in the p_return_status parameter of the API. The different possible values for an API return status are:

Success (FND_API. G_RET_STS_SUCCESS)

A success return status means that the API performed all the operations that you requested. A success return status may be accompanied by messages in the API message list.

Error (FND_API. G_RET_STS_ERROR)

An error return status means that the API failed to perform some or all of the operations that you requested. An error return status is usually accompanied by messages that describe the error (or errors) and how to fix it.

In most cases, you can take corrective actions to fix regular expected errors such as missing attributes or invalid date ranges.

Unexpected error (FND_API. G_RET_STS_UNEXP_ERROR)

An unexpected error status means that the API encountered an error condition that it did not expect or could not handle. In this case, the API cannot continue with its regular processing. Examples of such errors are irrecoverable data inconsistency errors, memory errors, and programming errors (such as attempting a division by zero).

In most cases, you cannot correct unexpected errors. System administrators or application developers usually must fix these errors.

Messages

The APIs place result messages into a message list. Programs that call these APIs retrieve the messages from the list and either issue the messages, load the messages in a database table, or write the messages to a log file.

Messages are stored in an encoded format. You can determine message names by using the standard functions provided by the message dictionary. Encoding messages also makes it possible to store the messages in database tables, and to generate reports from these tables in different languages.

The API message list must be initialized every time a program calls an API. API callers have the choice of either calling the message list utility function FND_MSG_PUB.Initialize or requesting the API to do the initialization on their behalf by setting the p_init_msg_list parameter to TRUE.

The program that calls the API can retrieve messages from the message stack using the existing FND API functions FND_MSG_PUB.Count_Msg and FND_MSG_PUB.Get.

API Usage

This section describes how to use the Credit Memo Approval and Creation API to initiate a Credit Memo Request workflow process request and to check the status of an existing request. The API is made up of two routines: AR_CREDIT_MEMO_API_PUB.Create_Request and AR_CREDIT_MEMO_API_PUB.Get.Status.

Prerequisites

You must define three HTML pages that display this information:

- The credit memo dispute request
- The original transaction details
- The transaction activities

You provide the API with the URLs of these pages. When workflow notifications are sent to the collector, approver, and receivable roles, links to the URLs are set in the message body of the notification. If the URLs are not correctly set up, then you will receive an error message such as "URL not found" when you click on the links.

You must also set up the Credit Memo Request workflow before you use the Credit Memo Approval and Creation API. For more information, see "Setting Up Credit Memo Request Workflow" in Chapter 4 of the *Oracle Receivables User Guide*.

AR_CREDIT_MEMO_API_PUB.Create_Request

Description

You can call this routine to create the Credit Memo Request workflow process request. When the workflow request has been created, the API returns a unique request ID number (p_request_id) that you can use to track the status of the request. The following is a breakdown of this routine's parameters, based upon parameter type:

Standard Parameters

This table lists and describes the standard parameters common to all routines in the Credit Memo Approval and Creation API.

Parameter	Type	Data-type	Required	Default Value	Description
p_api_version	IN	NUMBER	Yes		Used to compare version numbers of incoming calls to current version number.
p_init_msg_list	IN	VARCHAR2		FND_API.G_ FALSE	Set to TRUE to have the API automatically initialize the message list.
p_commit	IN	VARCHAR2		FND_API.G_ FALSE	Set to TRUE to have the API commit automatically.
x_return_status	OUT	VARCHAR2			Overall return status of the API.
x_msg_count	OUT	NUMBER			Number of messages in the API message list.
x_msg_data	OUT	VARCHAR2			Message in encoded format if x_ msg_count=1.

Create_Request Parameters

This table lists and describes parameters that specifically pertain to the Create_ Request routine:

See *Legend* on page 2-11 for this table's legend.

Parameter	Type	Data-type	Required	Description
p_customer_trx_id	IN	ra_customer_ trx.customer_trx_ id%type	Yes	Customer_trx_id of the disputed invoice.
p_line_credit_flag	IN	ra_cm_ request.line_credit_ flag	Yes	This value should be set to Y if the dispute is at the line level.
p_line_amount	IN	ra_cm_ request.line_ amount%type	Yes/No	Amount of the line dispute at the header level. If the dispute is at the header level, you should enter either the line_amount, tax_amount or freight_amount.
p_tax_amount	IN	ra_cm_request.tax_ amount	Yes/No	Amount of the tax dispute at the header level.
p_freight_amount	IN	ra_cm_ request.freight_ amount	Yes/No	Amount of the freight dispute at header level.
p_cm_reason_code	IN	ra_cm_ requests.cm_ reason_code%type	YES	User defined lookup code that represents the reason for the invoice dispute. Should be a valid lookup_code for the lookup_type CREDIT_ MEMO_REASON.
p_comments	IN	ra_cm_ requests.comments %type	No	Comments about the credit memo request, entered if required. These comments appear in the notes region of the Transaction window.
p_orig_trx_number	IN	VARCHAR2	No	Enter the duplicate invoice number if using the "Duplicate Billing" reason code.
p_tax_ex_cert_num	IN	VARCHAR2	No	Tax exemption certificate number.
p_request_url*	IN	VARCHAR2	No**	URL that displays the information of the actual credit memo dispute request.* See *Legend* on page 2-11 for this table's legend.
p_transaction_url	IN	VARCHAR2	No**	URL that displays the information of the original transaction. See *Legend* on page 2-11 for this table's legend.
p_trans_act_url	IN	VARCHAR2	No**	URL that displays information about the original transaction activities. See *Legend* on page 2-11 for this table's legend.

Parameter	Type	Data-type	Required	Description
p_cm_line_ tbl(x).customer_ trx_line_id	IN	ra_customer_trx_ line.customer_trx_ line_id%type	Yes/No	This value must be entered only if the dispute is at the line level. This value indicates the line_id that is in dispute. Note: Where p_cm_line_tbl(x), x indicates the index. The dispute can be for multiple lines.
p_cm_line_ tbl(x).extended_ amount	IN	ra_customer_trx_ line.extended_ amount%type	Yes/No	This value must be entered only if the dispute is at the line level. This value indicates the amount that is in dispute for the line.
p_cm_line_ tbl(x).quantity_ credited	IN	NUMBER	Yes/No	This value must be entered only if the dispute is at the line level. This value indicates the quantity that is in dispute for the line.
p_cm_line_ tbl(x).price	IN	NUMBER	Yes/No	This value must be entered only if the dispute is at the line level. This value indicates the price that is in dispute for the line.
p_skip_workflow_ flag	IN	VARCHAR2	No	Defaults to N. If this value is set to Y, the entire workflow is skipped for that particular request and the credit memo is directly created.
p_credit_method_ installments	IN	VARCHAR2	No	The p_credit_method_installments is the credit method that is used for crediting a transaction that uses split payment terms. Choices include PRORATE, LIFO, FIFO, or NULL. This value may be required if the p_skip_ workflow_flag is set to Y. ■ This parameter is mandatory if the credit memo is against a transaction that uses split payment terms and LINE_TYPE = LINE or CHARGES, or you are passing header freight. ■ Do not enter a value for this parameter if LINE_TYPE = TAX, or if you are passing freight for a specific line.

Parameter	Type	Data-type	Required	Description
p_credit_method_rules	IN	VARCHAR2	No	The p_credit_method_rules is the credit method for crediting a transaction which uses an accounting rule. Choices include PRORATE, LIFO, UNIT, or NULL. This value may be required if the p_skip_workflow_flag is set to Y. ■ This parameter is mandatory if the credit memo is against a transaction which uses an accounting rule and LINE_TYPE = LINE or CHARGES, or you are passing header freight. ■ Do not enter a value for this parameter if LINE_TYPE = TAX, or if you are passing freight for a specific line.
p_batch_source_name	IN	VARHCAR2	No	This value is required if the p_skip_workflow_flag is set to Y.
x_request_id	OUT	VARCHAR2	Yes	Request_id of the credit memo that is returned if the data passed is valid and the credit memo request is created.

Legend

* The request confirmation page might need the request_id as a parameter to query the information. This will not be available to the calling program when creating the p_request_url parameter because the request_id is the out parameter of the API. Calling programs should leave the request_id value blank and the table handler will add the request_id value and pass it to Workflow. The code searches for the "req_id=" string and replaces it with req_id="req_id". The parameter name must be req_id.

For example: For the old technology stack (PL/SQL), the following represents the request URL in *i*Receivables to call the "Request Confirmation" page. Note that no value has been entered for the req_id.

```
'arw_single_trx.single_cm_page?req_id='||'req_id='||'`&component='||glb_inv_part||'`&pct_change='||glb_percent_change;
```

** If the calling application does not enter the request, transaction, and transaction activities URLs, then you will see a default page reading "Unavailable" when you click on these links in the notifications screen. It is strongly recommended that the

calling application have the UI (user interface) display these pages and pass these URLs to the API.

Parameter validation

The API validates all parameters that you enter. If any of the required fields are missing or invalid, then the API returns an error message. A list of error messages is documented in *Messages* on page 2-17.

Example

This example shows a simple test case for creating a credit memo request for a dispute at the header level:

Objective:

To create a credit memo request.

Parameters entered:

customer_trx_id = 99999

line_credit_flag = N

line_amount = -100

cm_reason_code = RETURN

Call to the API:

```
AR_CREDIT_MEMO_API_PUB.Create_Request(
    x_return_status                     => p_return_status,
    x_msg_count                         => p_msg_count,
    x_msg_data                          => p_msg_data ,
CREDIT MEMO REQUEST PARAMETERS:
    p_customer_trx_id                   => 99999,
    p_line_credit_flag                  => 'N',
    p_line_amount                       => -100,
    p_cm_reason_code                    => 'RETURN',
    p_request_url                       => 'arw_single_trx.single_trx_
    page?p1=19769&p2=1&wf=Y',

    p_transaction_url                   => 'arw_single_trx.single_trx_
    page?p1=19769&p2=1&wf=Y'

    p_trans_act_url                     => 'arw_single_trx.single_act_
    page?p1=19769&p2=1&wf=Y'
```

```
x_request_id                    => p_request_id
```

AR_CREDIT_MEMO_API_PUB.Get_Request_Status

Description

Use this routine to view the Credit Memo Request workflow process request status. The API returns the status of the request and information about where the request is in the workflow. The following is a breakdown of parameters for this routine, based on parameter type:

Standard parameters

This table shows the standard API parameters common to all routines in the Credit Memo Approval and Creation API:

Parameter	Type	Data-type	Required	Default Value	Description
p_api_version	IN	NUMBER	Yes		Used to compare version numbers of incoming calls to its current version number.
p_init_msg_list	IN	VARCHAR2		FND_API.G_ FALSE	Set to TRUE to have the API automatically initialize the message list.
x_return_status	OUT	VARCHAR2			Overall return status of the API.
x_msg_count	OUT	NUMBER			Number of messages in the API message list.
x_msg_data	OUT	VARCHAR2			Message, in encoded format if x_ msg_count=1.

Get_Request_Status parameters

This table shows parameters that specifically pertain to the Get_Request_Status routine:

Parameter	Type	Data-type	Required	Description
p_request_id	IN	ra_cm_ requests.request_ id%type	YES	ID of the credit memo request whose status you are checking.

Parameter	Type	Data-type	Required	Description
x_status_meaning	OUT	VARCHAR2		Status of the credit memo request.
x_reason_ meaning	OUT	VARCHAR2		Reason for the dispute of the credit memo request.
x_customer_trx_ id	OUT	ra_cusotmer_ trx.customer_trx_ id%type		Customer transaction ID for the dispute of the credit memo request.
x_cm_customer_ trx_id	OUT	ra_cusotmer_ trx.customer_trx_ id%type		Credit memo transaction ID that was created for the dispute.
x_line_amount	OUT	ra_cm_requests.line_ amount%type		Total amount of dispute for lines.
x_tax_amount	OUT	ra_cm_requests.tax_ amount%type		Total amount of dispute for tax.
x_freight_amount	OUT	ra_cm_requests.freight_ amount%type		Total amount of dispute for freight.
x_line_credits_ flag	OUT	ra_cm_requests.line_ credits_flag%type		Indicates whether the dispute is at the line level or the header level. If the value is set to Y, the dispute is at the line level.
x_created_by	OUT	wf_users.display_ name%type		Name of the requestor.
x_creation_date	OUT	DATE		Date of the request.
x_comments	OUT	ra_cm_ requests.comments% type		Comments entered by the requestor.
x_approval_date	OUT	DATE		Credit memo approval date if the credit memo has been created for the request.
x_cm_line_tbl	OUT	cm_line_tbl_type_cover		Table that contains the line level dispute information. The values in the table will be set if the x_line_credits_flag = Y.
x_cm_activity_tbl	OUT	cm_activity_tbl_type_ cover		Table that contains the status of the activities for the request.
x_cm_notes_tbl	OUT	cm_notes_tbl_type_ cover		Table that contains the notes inserted for the transaction that is disputed.

Note:

```
TYPE  CM_LINE_REC_TYPE_COVER IS RECORD
```

```
    customer_trx_line_id:  ra_customer_trx_lines.customer_trx_line_id%type,
    extended_amount:  ra_customer_trx_lines.extended_amount%type,
    quantity_credited:  number,
    price:  number;
TYPE CM_LINE_TBL_TYPE_COVER
    IS TABLE OF
    CM_LINE_REC_TYPE_COVER
    INDEX BY BINARY INTEGER;
x_cm_line_tbl  CM_LINE_TBL_TYPE_COVER;
TYPE  CM_ACTIVITY_REC_TYPE_COVER IS RECORD
    begin_date:  DATE,
    activity_name: VARCHAR2(80),
    status: wf_item_activity_statuses.activity_status%type,
    user: wf_item_activity_statuses.activity_user%type);
TYPE CM_ACTIVITY_TBL_TYPE_COVER
    IS TABLE OF
    CM_ACTIVITY_REC_TYPE_COVER
    INDEX BY BINARY INTEGER;
x_cm_activity_tbl   CM_ACTIVITY_TBL_TYPE_COVER;
TYPE  CM_NOTES_REC_TYPE_COVER IS RECORD
    ( NOTES   ar_notes.text%type);
TYPE CM_NOTES_TBL_TYPE_COVER
    IS TABLE OF
    CM_NOTES_REC_TYPE_COVER
    INDEX BY BINARY INTEGER;
x_cm_notes_tbl   CM_NOTES_TBL_TYPE_COVER;
```

Parameter validation

The API validates all parameters that you enter. If any of the required fields are missing or invalid, then the API returns an error message. A list of error messages is documented in *Messages* on page 2-17.

Example

The following example is a simple test case for viewing the status of the credit memo request.

Objective:

To get the status of the credit memo request.

Parameters entered:

request_id = 122

Call to the API:

```
AR_CREDIT_MEMO_API_PUB.Get_Request_Status(
    p_api_version           =>   1.0,
    x_msg_count             =>   msg_count ,
    x_msg_data               =>   msg_data,
    x_return_status         =>   return_status,
    p_request_id        =>  request_id,
    x_status_meaning =>  status_meaning,
    x_reason_meaning =>  reason_meaning,
    x_customer_trx_id =>  customer_trx_id,
    x_cm_customer_trx_id =>  cm_customer_trx_id,
    x_line_amount =>  line_amount,
    x_tax_amount =>  tax_amount,
    x_freight_amount =>  freight_amount,
    x_line_credits_flag =>  line_credits_flag,
    x_created_by =>  created_by,
    x_creation_date =>  creation_date,
    x_cm_line_tbl =>  cm_line_tbl,
    x_cm_activity_tbl =>  cm_activity_tbl,
    x_cm_notes_tbl =>  cm_notes_tbl);
```

Messages

The following table describes the possible messages returned by the Credit Memo Approval and Creation API.

Message Number	Message Name	Message Description
11936	AR_RAXTRX-1719	You must supply a reason code for your credit memo transaction.
11091	AR_CKAP_OVERAPP	You cannot overapply this transaction.
42711	AR_TAPI_LINE_NOT_EXIST	Line does not exist (customer_trx_line_id:[customer_trx_line_id]).
42756	AR_TAPI_TRANS_NOT_EXIST	Transaction does not exist (customer_trx_id:[customer_trx_id]).
294003	AR_CMWF_API_INVALID_VALUE	You specified an invalid value for the LINE_CREDIT_FLAG parameter. The valid values are Y and N.
294004	AR_CMWF_API_NO_LINES_INFO	The value for LINES_CREDIT_FLAG is Y, please provide at least one line level information.
294002	AR_CMWF_API_INVALID_REQUEST_ID	Request does not exist (REQUEST_ID : &REQUEST_ID)

3

Credit Request Creation API User Notes

Overview

This document outlines the specifications and the methodology for using the Credit Request Creation API.

The Credit Request Creation API is a PL/SQL API that creates a credit request in the Credit Management system based on the specified parameters. After the credit request is created with minimal validations, an asynchronous workflow is initiated that starts processing the credit request. The API does not cause performance degradation to the main flow calling the Credit Request API.

Basic Business Needs

Products in the Oracle e-Business Suite use the Oracle Receivables Credit Management feature as part of their respective business flows to perform credit analyses on parties, accounts, or account sites. Currently, Oracle Order Management and Oracle Contracts for Leasing are two such products.

In addition to entering the credit request (more formally known as the credit application) from within Credit Management, e-Business Suite products might want to trigger the creation of a credit request from their existing flows.

Specific requirements from each product also exist for handling the results (which are the recommendations) of a credit analysis performed in Credit Management.

Some of these products might also want to perform custom processing if the automation process of a credit request fails.

Before you begin....

Initialization of ARP_STANDARD and ARP_GLOBAL

Custom code that uses AR or HZ APIs will set the ORG_ID via dbms_application_info.set_client_info() and then call the APIs. The APIs in turn might access either ARP_STANDARD and ARP_GLOBAL, which initialize the global variables that are used across Oracle Receivables when the package is first called. Most of these global variable values are organization dependent, and the first such call sets the global variables based on the current ORG_ID.

If additional custom code then changes the ORG_ID via another call to dbms_application_info.set_client_info(), then the ORG context changes, ***but the ARP_STANDARD and ARP_GLOBAL context does not.***

In such cases, you should explicitly re-initialize the global variables by a call to these two public procedures:

1. ARP_GLOBAL.INIT_GLOBAL: For setting public variables in ARP_GLOBAL.
2. ARP_STANDARD.INIT_STANDARD: For setting public variables in ARP_STANDARD.

Major Features

- Easy to administrate and maintain the consumer product's processes.
- Credit Management has published business events.
- External systems can subscribe to the business events in Credit Management.

Solution Outline

PL/SQL APIs

To achieve the basic functionality of creating a credit request, the following API can be called:

- *Ar_cmgt_credit_request.create*: Creates a credit request.

API Usage

Ar_cmgt_credit_request.create

Description

This routine is used to create a credit request for initiating a credit review for a party, account, or account site.

This API routine has 4 output and 21 input parameters. The API returns the credit_request_id of the credit request created in Credit Management as one of the default output parameters. The following is the breakdown of the parameters:

Input

Standard API parameters: 4

Credit Request parameters: 18

Output

Standard API parameters: 3

Credit Request parameters: 1

The following table lists standard API parameters that are common to all Credit Request API routines.

Parameter	Type	Data-type	Required	Default Value	Description
p_api_version	IN	NUMBER	Yes		Used to compare version numbers of incoming calls to its current version number.Unexpected error is raised if version in-compatibility exists.In the current version of the API, you should pass in a value of 1.0 for this parameter.
p_init_msg_list	IN	VARCHAR2	FND_API.G_FALSE		Allows API callers to request that the API does initialization of the message list on their behalf.
p_commit	IN	VARCHAR2	FND_API.G_FALSE		Used by API callers to ask the API to commit on their behalf.

Parameter	Type	Data-type	Required	Default Value	Description
p_validation_ level	IN	NUMBER	FND_ API.G_ VALID_ LEVEL_ FULL		Not to be used currently.
x_return_status	OUT	VARCHAR2			Represents the API overall return status. The expected values are: --FND_API.G_RET_STS_SUCCESS --FND_API. G_RET_STS_ERROR --FND_API.G_RET_STS_UNEXP_ ERROR
x_msg_count	OUT	NUMBER			Number of messages in the API message list
x_msg_data	OUT	VARCHAR2			This is the message in encoded format if x_msg_count=1

The following table lists the parameters that are relevant to the credit request.

Parameter	Type	Data-type	Required*	Description
P_application_ number	IN	VARCHAR2		The application number on the credit request. This could be internally generated by a sequence based on the setting in the credit management system options form.
P_application_date	IN	DATE		The date on the credit request.
P_requestor_id	IN	NUMBER		The person_id (from HR tables) of the employee placing the credit request.
P_review_type	IN	VARCHAR2		The 'review type' of the credit request. The 'review types' would be created as AR lookups at the time of the credit management setup.
P_credit_ classification	IN	VARCHAR2		The 'credit classification' of the party/account/site that would be created as AR lookups at the time of the credit management setup.
p_requested_ amount	IN	NUMBER		This is the requested credit limit amount.

Parameter	Type	Data-type	Required*	Description
p_requested_ currency	IN	VARCHAR2		The currency of the requested credit limit.
p_trx_amount	IN	NUMBER		The transaction amount.
p_trx_currency	IN	VARCHAR2		The transaction currency.
credit_type	IN	VARCHAR2		The type of the credit request. The possible values are 'Term' and 'Trade'
P_term_length	IN	NUMBER		This is the term length specified as number of months. This would be relevant for the credit_type 'Term'.
p_credit_check_ rule_id	IN	NUMBER		Identifier of the credit check rule defined in Order Management. (This is OM specific attribute).
p_credit_request_ status		VARCHAR2		The credit request status. Possible values are SAVE and SUBMIT.
P_party_id	IN			The party identifier.
P_cust_account_id	IN	NUMBER		The customer account identifier.
P_cust_acct_site_id	IN	NUMBER		The customer account site identifier.
P_credit_contact_ party_id	IN	NUMBER		The party identifier for the credit contact.
p_site_use_id	IN	NUMBER		The site use identifier.
P_notes	IN	VARCHAR2		Notes
P_source_name	IN	VARCHAR2		The source name on the credit request, which is used to identify the source product that initiated the credit request.
P_source_column1	IN	VARCHAR2		The unique identifier of the entity for which the credit request was created.
P_source_column2	IN	VARCHAR2		The unique identifier of the entity for which the credit request was created.
P_source_column3	IN	VARCHAR2		The unique identifier of the entity for which the credit request was created.
P_credit_request_id	OUT	NUMBER		The credit request identifier.

Parameter	Type	Data-type	Required*	Description
p_case_folder_ number	IN	VARCHAR2		The case folder number, which is used by the Case Folder generated by this credit request.

Exception Handling and Result Messages

The Credit Request API gives back three types of information to its calling programs:

- Overall status
- Messages describing the operations performed or errors encountered by the APIs
- Some output values that the API caller might want to use (this is different for different API routines and is described in *API Usage* on page 3-6)

Return Status

The return status (x_return_status) of the API informs the caller about the result of the operation (or operations) performed by the API. The different possible values for an API return status are:

- Success (FND_API.G_RET_STS_SUCCESS)
- Error (FND_API.G_RET_STS_ERROR)
- Unexpected error (FND_API.G_RET_STS_UNEXP_ERROR)

The following section describes the different values of return status and their meanings.

Success

A success return status means that the API was able to perform all the operations requested by its caller. A success return status may be accompanied by messages in the API message list which will be informative.

Error

An error return status means that the API failed to perform some or all of the operations requested by its caller. An error return status is usually accompanied by messages describing the error (or errors) and how to fix it.

In most cases, you should be able to take corrective action to fix regular, expected errors such as missing attributes or invalid date ranges.

Unexpected error

An unexpected error status means that the API has encountered an error condition it did not expect or could not handle. In this case, the API is unable to continue with its regular processing. Examples of such errors are unrecoverable data consistency errors, memory errors, and programming errors (such as attempting a division by zero).

In most cases, only system administrators or application developers can fix these unexpected errors.

Messages

The APIs put result messages into a message list. Programs calling the APIs can then get the messages from the list and process them by issuing them, loading them into a database table, or writing them to a log file.

Messages are stored in an encoded format to let the API callers find message names using the standard functions provided by the message dictionary. It also allows the storing of these messages in database tables and reporting off these tables in different languages.

The API message list must be initialized every time a program calls an API. API callers can either call the message list utility function FND_MSG_PUB.Initialize or request that the API do the initialization on their behalf by setting the p_init_msg_ list parameter to TRUE.

The program calling the API can retrieve messages from the message stack using the existing FND API functions FND_MSG_PUB.Count_Msg and FND_MSG_ PUB.Get.

Message Level Threshold

The message level threshold is stored in a profile option named FND_API_MSG_ LEVEL_THRESHOLD. This profile option can be updated at all levels (site, application, or user). The API checks against this threshold before writing a message to the API message list.

Credit Request Business Events

Oracle Credit Management leverages the Business Events System available in Oracle Workflow Release 2.6 to create business events during the life cycle of a credit request. This lets customers/consumer products perform custom processing.

Note: Credit request events are raised by the Credit Request processing workflow engine; users should not manually raise them.

Setup

The Event Subscription

1. Log on to Oracle Applications with the System Administrator responsibility.
2. Navigate Workflow > Add subscription to Event.
3. Every Credit Request event follows the same naming convention:

   "oracle.apps.ar.cmgt.CreditRequest.<phase><action>"

For example, if you want to subscribe your business process (such as custom recommendations) to the implementation of the default recommendations for a credit request, then you should subscribe your routine or custom program (also known as the rule function) to the oracle.apps.ar.cmgt.CreditRequest.Recommendation.implement event.

Deferred Subscriptions

The Workflow Release 2.6 Business Event System allows subscriptions to be executed in deferred mode so that no overhead is added to the process raising the event. For Credit Request business events, it is recommended that user subscriptions be executed in the deferred mode.

One of the mechanisms to defer a subscription would be by setting the phase number of a user-defined subscription to greater than 99. For additional details on different mechanisms for deferring a subscription, see: Event Subscriptions, *Oracle Workflow Developer's Guide*.

When the credit request processing flow raises an event for a deferred subscription, the event message is sent to the deferred queue.

Subscriptions will be executed when the "Workflow Agent Listener" with the parameter "WF_DEFERRED" is executed. When this concurrent program is executed, every subscription from every instance of events in the DEFERRED queue at that moment will be executed.

This concurrent program is seeded in the request group of the System Administrator responsibility.

Coding a Subscription

How to subscribe

Per Oracle Workflow coding standards, two kinds of subscriptions exist for use:

- Oracle Workflow Release 2.6 rule function (which is a PL/SQL function)
- Workflow processes

For information about the standards for the event subscription rule function, see: Standard API for an Event Subscription Rule Function, *Oracle Workflow Developer's Guide*.

The parameters provided by events

Oracle Workflow uses the object type WF_EVENT_T to store event messages. WF_EVENT_T defines the event message structure that the Business Event System and the Workflow Engine use to represent a business event.

For more information about this Event Message structure, see: Event Message Structure, *Oracle Workflow API Reference*.

As part of the event message, the raised event can pass a number of parameters to the subscription rule function in a named varying array WF_PARAMETER_LIST_T. The credit request event passes 13 parameters, listed in the following table:

Parameter Name	Description
CREDIT_REQUEST_ID	The unique identifier of the credit request.
USER_ID	The user id of the initial session that had initiated the credit request.
RESP_ID	The responsibility id of the initial session that had initiated the credit request.

Parameter Name	Description
RESP_APPL_ID	The application responsibility id of the initial session that had initiated the credit request.
SECURITY_GROUP_ID	The security group id of the initial session that had initiated the credit request.
ORG_ID	The Operating Unit id of the initial session that had initiated the credit request.
SOURCE_NAME	The source name on the credit request, which is used to identify the source product that initiated the credit request.
SOURCE_COLUMN1	The unique identifier of the entity for which the credit request was created.
SOURCE_COLUMN2	The unique identifier of the entity for which the credit request was created.
SOURCE_COLUMN3	The unique identifier of the entity for which the credit request was created.

The org_id, user_id, resp_id, resp_appl_id, and security_group_id parameters let users re-create the application environment at the moment the credit request was raised. This is useful if your business logic depends on some application context parameters, such as for a multi-organization environment.

You can use FND_GLOBAL.APPS_INITIALIZE(<USER_ID>, <RESP_ID>, <RESP_APPL_ID>, <SECURITY_GROUP_ID>) to re-create the original application context in your process.

Example of a PL/SQL rule function subscription

For example, for a particular requirement, consumer products in Oracle eBusiness Suite need to implement their custom recommendations after the Credit Request workflow has implemented its own recommendations for a particular credit request.

Here is one example of rule function.

If the following rule function is subscribed to Credit Request event: "oracle.apps.ar.cmgt.CreditRequest.Reccomendation.implement":

```
FUNCTION Rule_Credit_Recco_Impl
(p_subscription_guid  in raw,
p_event               in out wf_event_t)
RETURN VARCHAR2
IS
 l_key         varchar2(240) := p_event.GetEventKey();
....
BEGIN
  l_org_id := p_event.GetValueForParameter('ORG_ID');
  l_user_id := p_event.GetValueForParameter('USER_ID');
  l_resp_id := p_event.GetValueForParameter('RESP_ID');
  l_resp_appl_id := p_event.GetValueForParameter('RESP_APPL_ID');
  l_security_group_id := p_event.GetValueForParameter('SECURITY_GROUP_ID');
  l_credit_request_id := p_event.GetValueForParameter('CREDIT_REQUEST_ID');
  l_source_name := p_event.GetValueForParameter('SOURCE_NAME');
  l_source_column1 := p_event.GetValueForParameter('SOURCE_COLUMN1');
  l_source_column2 := p_event.GetValueForParameter('SOURCE_COLUMN2');
  l_source_column3 := p_event.GetValueForParameter('SOURCE_COLUMN3');
  l_party_id := p_event.GetValueForParameter('PARTY_ID');
  l_cust_account_id := p_event.GetValueForParameter('CUST_ACCOUNT_ID');
  l_cust_acct_site_id := p_event.GetValueForParameter('CUST_ACCT_SITE_ID');
fnd_global.apps_initialize (l_user_id, l_resp_id, l_resp_appl_id,
                            l_security_group_id);
--
--Implement custom recommendations.
--
END;
```

Credit Request Events

Credit Request events follow this naming pattern: oracle.apps.ar.cmgt.CreditRequest.<Phase>.<Action>.

This name contains three parts:

1. oracle.apps.ar.cmgt.CreditRequest means that the event belongs to Credit Request entity in the Credit Management application.
2. <Phase> indicates the current stage in the life cycle of a credit request in which an action is going to be performed.
3. <Action> indicates the action performed in the current phase of the credit request.

This table lists the possible phases and actions.

Credit Request Phase	Action	Description	ID Parameter Name
Automation	Failure	Failure of an automated credit request processing because of non-availability of the required data-points on the checklist or the scoring model.	CREDIT_REQUEST_ID
Recommendation	Implement	Implementation of the recommendations, coming out of the analysis performed on the credit request.	CREDIT_REQUEST_ID

4

Deposit API User Notes

Overview

This document outlines the specifications and the methodology for using the various Commitment (Deposit) APIs. These APIs provide an extension to existing functionality of creating and manipulating deposits through the standard Oracle Receivables Transactions workbench.

You can access these APIs:

- As standard PL/SQL servers-side routine calls
- Through Forms, utilizing the capability of Forms6 to have a procedure as its underlying base table

Basic Business Needs

In its first phase, the Commitment (Deposit) API caters to the following basic functionality via different API calls:

- Creating a commitment of type Deposit
- Creates non-revenue sales credit for a deposit

Before you begin....

Initialization of ARP_STANDARD and ARP_GLOBAL

Custom code that uses AR or HZ APIs will set the ORG_ID via dbms_application_info.set_client_info() and then call the APIs. The APIs in turn might access either ARP_STANDARD and ARP_GLOBAL, which initialize the global variables that are used across Oracle Receivables when the package is first called. Most of these global variable values are organization dependent, and the first such call sets the global variables based on the current ORG_ID.

If additional custom code then changes the ORG_ID via another call to dbms_application_info.set_client_info(), then the ORG context changes, ***but the ARP_STANDARD and ARP_GLOBAL context does not.***

In such cases, you should explicitly re-initialize the global variables by a call to these two public procedures:

1. ARP_GLOBAL.INIT_GLOBAL: For setting public variables in ARP_GLOBAL.
2. ARP_STANDARD.INIT_STANDARD: For setting public variables in ARP_STANDARD.

Major Features

Flexibility

Per Oracle API coding standards, the various APIs in the Commitment (Deposit) API package let you specify an ID or its associated value for any attribute that is an INPUT parameter of the API.

If both an ID and value have been specified, then the ID takes precedence over the value. This provides a wide degree of flexibility when using the API, both as a base table of the form and as a server-side routine call from the PL/SQL code.

The extensive defaulting mechanism for the input parameters ensures that you will be able to achieve the basic business needs of creating a deposit by calling the relevant APIs with a minimum number of parameters. This gives you many options to achieve your requirements when you call the relevant API.

Modular Approach

The Commitment (Deposit) API has been designed in a highly modular fashion, giving you code that is:

- Easy understand
- Easy to maintain
- Easy to extend

Error Handling

The Commitment (Deposit) API provides an extensive error-handling and error-reporting mechanism whereby all errors encountered in the Defaulting and Validation phases are reported and put on the message stack. The calling program can look up all error messages, or the first error message on the stack.

If only one error exists on the message stack, then you do not need to fetch the message from the stack because the message will return as one of the output parameters of the API routine.

Robust Validation

The validations that the Commitment (Deposit) API performs are robust in nature. The APIs collect all the validation errors encountered and put them on the message stack. The relevant entity handler is called only if no errors are reported during the Defaulting and Validation phases.

Debug Messages

Extensive debug messages have been incorporated. In case of unexpected problems, use these messages to troubleshoot. This is extremely useful because APIs would be difficult to debug otherwise.

Debug messages can be written to the log file by calling the appropriate routines described in *Exception Handling and Result Messages* on page 4-7.

Solution Outline

PL/SQL APIs

To achieve the basic functionality of creating a deposit, the following API can be called:

- *AR_DEPOSIT_API_PUB.Create_deposit* on page 4-10: Creates a single deposit and completes it.
- *AR_DEPOSIT_API_PUB.insert_non_rev_salescredit* on page 4-30: Creates non revenue sales credit for a deposit.

Modular Approach

To modularize the Commitment (Deposit) API, the basic structure of the API is divided into four parts:

1. Defaulting the IDs from the values and cross validating, if you provide both the values and the IDs.
2. Defaulting all the entity level information, which the user has not entered or which the API needs internally.
3. Validating the entity level information entered by the user.
4. Call to the entity handlers to perform the relevant task (viz. Create).

This results in easy to understand and easy to maintain code. Any new functionality can be added by a simple code plug-in at each of the four parts.

Defaulting

In general, the various parameters in the API call, if not entered, get defaulted based on the following:

- Values of the other parameters in the API call
- Values set in the AR_SYSTEM_PARAMETERS table entered through the System Options form
- Relevant profile option values

Depending on the above three factors and the exact business requirement, the minimum number of parameters that may be required to perform certain business tasks may vary.

Null values are defaulted for the parameters that could not be defaulted by the API defaulting routines.

For various attributes of the business objects, you can pass either the ID or the value of the attribute.

If you specify only the value, then the value is used to derive the ID; otherwise, the ID (if specified) is taken directly. If you specify both the ID and the value, then the ID takes precedence over the value and a warning message informs you of this.

Exception Handling and Result Messages

The Commitment (Deposit) APIs return three types of information to their calling programs:

- Overall status
- Messages describing the operations performed or errors encountered by the APIs
- Some output values that the API caller might want to use (this is different for different API routines and is described in *API Usage* on page 4-10)

Return Status

The return status (x_return_status) of the API informs the caller about the result of the operation (or operations) performed by the API. The different possible values for an API return status are:

- Success (FND_API.G_RET_STS_SUCCESS)
- Error (FND_API.G_RET_STS_ERROR)
- Unexpected error (FND_API.G_RET_STS_UNEXP_ERROR)

The following section describes the different values of return status and their meanings.

Success

A success return status means that the API was able to perform all the operations requested by its caller. A success return status may be accompanied by informative messages in the API message list.

Error

An error return status means that the API failed to perform some or all of the operations requested by its caller. An error return status is usually accompanied by messages describing the error (or errors) and how to fix it.

In most cases, you should be able to take corrective action to fix regular, expected errors such as missing attributes or invalid date ranges.

Unexpected error

An unexpected error status means that the API has encountered an error condition it did not expect or could not handle. In this case, the API is unable to continue with its regular processing. Examples of such errors are irrecoverable data inconsistency errors, memory errors, and programming errors (such as attempting a division by zero).

In most cases, only system administrators or application developers can fix these unexpected errors.

Messages

The APIs put result messages into a message list. Programs calling the APIs can then get the messages from the list and process them by issuing them, loading them into a database table, or writing them to a log file.

Messages are stored in an encoded format to let the API callers find message names using the standard functions provided by the message dictionary. It also allows the storing of these messages in database tables and reporting off these tables in different languages. See *Messages* on page 4-36 for more information.

The API message list must be initialized every time a program calls an API. API callers can either call the message list utility function FND_MSG_PUB.Initialize or request that the API do the initialization on their behalf by setting the p_init_msg_ list parameter to TRUE.

The program calling the API can retrieve messages from the message stack using the existing FND API functions FND_MSG_PUB.Count_Msg and FND_MSG_ PUB.Get.

Message Level Threshold

The message level threshold is stored in a profile option named FND_API_MSG_LEVEL_THRESHOLD. This profile option can be updated at all levels (site, application, or user). The API checks against this threshold before writing a message to the API message list.

Debug Messages

The calling program enables debugging by calling the routine arp_standard.enable_file_debug. The routine requires 2 parameters: path_name and file_name.

```
arp_standard.enable_file_debug(<pathname>, <filename>)
```

The path name can be identified by using the following select statement:

```
select value from v$parameter where name = 'utl_file_dir',
```

The file name can be any name that you choose.

Example

```
arp_standard.enable_file_debug ('/sqlcom/log','txt.log')
```

This call would write the output debug file 'txt.log' in the path '/sqlcom/log'.

Calling Program Context

The program calling these APIs should have set up the application, responsibility, and user in the context of Oracle Application.

If the calling program does not set up this context, then it can be done programmatically by calling the following FND API.

```
fnd_global.apps_initialize (      user_id in number,
                                  resp_id in number,
                                  resp_appl_id in number,
                                  security_group_id in number default 0);
```

API Usage

AR_DEPOSIT_API_PUB.Create_deposit

Description

This routine is called to create a deposit for the transactions.

Only one owner can be assigned to a commitment.

This API routine has 8 output and 136 input parameters in total. Of the output parameters, the API returns CUSTOMER_TRX_ID , CUSTOMER_TRX_LINE_ID, and new TRX_NUMBER, if generated during deposit creation.

The following is the breakdown of the parameters:

Input

Standard API parameters: 4

Deposit parameters: 132 + 2 (global descriptive flexfield parameter)

Output

Standard API parameters: 3

Deposit parameters: 5

The input global descriptive flexfield parameter is a record of type global_attr_rec_type.

```
TYPE global_attr_rec_type IS RECORD(
                global_attribute_category     VARCHAR2(30) default null,
                global_attribute1             VARCHAR2(150) default NULL,
                global_attribute2             VARCHAR2(150) DEFAULT NULL,
                global_attribute3             VARCHAR2(150) DEFAULT NULL,
                global_attribute4             VARCHAR2(150) DEFAULT NULL,
                global_attribute5             VARCHAR2(150) DEFAULT NULL,
                global_attribute6             VARCHAR2(150) DEFAULT NULL,
                global_attribute7             VARCHAR2(150) DEFAULT NULL,
                global_attribute8             VARCHAR2(150) DEFAULT NULL,
                global_attribute9             VARCHAR2(150) DEFAULT NULL,
                global_attribute10            VARCHAR2(150) DEFAULT NULL,
                global_attribute11            VARCHAR2(150) DEFAULT NULL,
                global_attribute12            VARCHAR2(150) DEFAULT NULL,
                global_attribute13            VARCHAR2(150) DEFAULT NULL,
                global_attribute14            VARCHAR2(150) DEFAULT NULL,
```

```
global_attribute15          VARCHAR2(150) DEFAULT NULL,
global_attribute16          VARCHAR2(150) DEFAULT NULL,
global_attribute17          VARCHAR2(150) DEFAULT NULL,
global_attribute18          VARCHAR2(150) DEFAULT NULL,
global_attribute19          VARCHAR2(150) DEFAULT NULL,
global_attribute20          VARCHAR2(150) DEFAULT NULL,
global_attribute21          VARCHAR2(150) DEFAULT NULL,
global_attribute22          VARCHAR2(150) DEFAULT NULL,
global_attribute23          VARCHAR2(150) DEFAULT NULL,
global_attribute24          VARCHAR2(150) DEFAULT NULL,
global_attribute25          VARCHAR2(150) DEFAULT NULL,
global_attribute26          VARCHAR2(150) DEFAULT NULL,
global_attribute27          VARCHAR2(150) DEFAULT NULL,
global_attribute28          VARCHAR2(150) DEFAULT NULL,
global_attribute29          VARCHAR2(150) DEFAULT NULL,
global_attribute30          VARCHAR2(150) DEFAULT NULL);
```

The following table lists standard API parameters that are common to all the routines in the Commitment (Deposit) API.

Note: If required parameters are not passed in a call to this API, then the call will fail. However, depending on the business scenario, you will have to pass in values for other parameters to successfully create the business object; otherwise, error messages will be reported.

Parameter	Type	Data-type	Required	Default Value	Description
p_api_version	IN	NUMBER	Yes		Used to compare version numbers of incoming calls to its current version number. Unexpected error is raised if version incompatibility exists. In the current version of the API, you should pass a value of 1.0 for this parameter.
p_init_msg_list	IN	VARCHAR2		FND_API.G_FALSE	Allows API callers to request that the API does initialization of the message list on their behalf.
p_commit	IN	VARCHAR2		FND_API.G_FALSE	Used by API callers to ask the API to commit on their behalf.

Parameter	Type	Data-type	Required	Default Value	Description
p_validation_ level	IN	NUMBER		FND_API.G_ VALID_LEVEL_ FULL	Not to be used currently as this is a public API.
x_return_status	OUT	VARCHAR2			Represents the API overall return status.
x_msg_count	OUT	NUMBER			Number of messages in the API message list.
x_msg_data	OUT	VARCHAR2			This is the message in encoded format if x_msg_count=1.

The following table lists the parameters that pertain specifically to the deposit.

Parameter	Type	Data-type	Required*	Description
p_deposit_ number	IN	VARCHAR2		The deposit number of the deposit to be created. Default: Null Validate: If AR_RA_BATCH_AUTO_NUM_FLAG set by batch source is true, then it is derived automatically; else it is required to be present. Error: AR_DAPI_DEPOSIT_NO_NULL
p_deposit_date	IN	Date		The deposit date of the entered deposit. Default: System date Validate: This field is mandatory. Error: None
p_usr_currency_ code	IN	VARCHAR2		The translated currency code. Used to derive the p_ currency_code if it is not entered. Default: None Validate: Should be a valid currency, so that the corresponding currency code can be derived. Error: AR_RAPI_USR_CURR_CODE_INVALID

Parameter	Type	Data-type	Required*	Description
p_currency_code	IN	VARCHAR2		The actual currency code that gets stored in AR tables. Default: Derived from p_usr_currency_code if entered, else defaults to the functional currency code. Validate: Validated against the currencies in FND_CURRENCIES table. Error: AR_RAPI_CURR_CODE_INVALID Warning: AR_RAPI_FUNC_CURR_DEFAULTED
p_usr_exchange_rate_type	IN	VARCHAR2		The translated exchange rate type. Used to derive the p_exchange_rate_type if it has not been entered. Default: None Validate: Should be a valid rate type. Error: AR_RAPI_USR_X_RATE_TYP_INVALID
p_exchange_rate_type	IN	VARCHAR2		Exchange rate type stored in AR tables. Default: In case of foreign currency receipt, the value is derived from p_usr_exchange_rate_type. If p_usr_exchange_rate_type is null, then the value defaults from the AR: Default Exchange Rate Type profile option. Validate: Validated against values in GL_DAILY_CONVERSION_TYPES table. Error: AR_RAPI_X_RATE_TYPE_INVALID
p_exchange_rate	IN	NUMBER		The exchange rate between the receipt currency and the functional currency. Default: Derived from the Daily Rates table for rate_type <> User in case of nonfunctional currency. If Journals: Display Inverse Rate profile option = Y, set user-entered value to 1/ p_exchange_rate. The entered value is rounded to a precision of 38. Validate: In case of nonfunctional currency, the rate should have a positive value for rate type=User For nonfunctional currency and type is <> User, do not specify any value. Error: AR_RAPI_X_RATE_INVALID AR_RAPI_X_RATE_NULL

Parameter	Type	Data-type	Required*	Description
p_exchange_rate_date	IN	DATE		The date on which the exchange rate is valid. Default: Receipt date Validate: For a nonfunctional currency and type is <>User, there should be a valid rate existing in the database for this date. This is a cross validation of type, currency, and date. Error: AR_NO_RATE_DATA_FOUND
p_batch_source_id	IN	Number		Batch source identifier for the commitment. Default: Same as ar_ra_batch_source profile option. Validation: It should be a valid batch source and it should exist in the database. This field is mandatory if not defined in profile option. Error: AR_DAPI_BS_NAME_INVALID AR_DAPI_BS_NAME_IGN AR_DAPI_BS_ID_INVALID
p_batch_source_name	IN	varchar2		Batch source name for the commitment. Default: Same as ar_ra_batch_source_name profile option. Validation: It should be a valid batch source and it should exist in the database. Error: AR_DAPI_BS_NAME_INVALID AR_DAPI_BS_NAME_IGN AR_DAPI_BS_ID_INVALID
p_cust_trx_type_id	IN	Number		Transaction Type identifier. Default: Based on the value of batch source Validation: It should be a valid transaction type. This field is mandatory. Error: AR_DAPI_TRANS_TYPE_INVALID AR_RAPI_TRANS_TYPE_IGN AR_DAPI_TRANS_TYPE_ID_INVALID
p_cust_trx_type	IN	Varchar2		Transaction Type name. Default: Based on the value of batch source Validation: It should be a valid transaction type. Error: AR_DAPI_TRANS_TYPE_INVALID AR_RAPI_TRANS_TYPE_IGN AR_DAPI_TRANS_TYPE_ID_INVALID

Parameter	Type	Data-type	Required*	Description
p_class	IN	Varchar2		Constant value = DEP. Keeping as an input for a future enhancement.
p_gl_date	In	Date		Date that this deposit will be posted to the general ledger. Default: Gets defaulted to the current date if it is a valid gl_date, otherwise: --If the most recent open period is prior to the receipt date: last date of that period. --If there is a period open after the deposit date: first date of the last open period. Validate: The gl date is valid if the following conditions are true: --The date is in an Open or Future period. --The period cannot be an Adjustment period. Error: AR_INVALID_APP_GL_DATE
p_bill_to_ customer_id	IN	Number		The CUSTOMER_ID for the bill-to customer. Default: Defaulted from customer name/number. If all name, number, and ID are null, then it is same as ship-to CUSTOMER_ID. Validate: Customer exists and has prospect code = CUSTOMER. Customer has a profile defined at the customer level. Either bill-to or ship-to customer must exist. Error: AR_RAPI_CUST_ID_INVALID AR_RAPI_CUS_NAME_INVALID AR_RAPI_CUS_NUM_INVALID AR_RAPI_CUS_NAME_NUM_INVALID AR_RAPI_CUS_NAME_NUM_IGN AR_DAPI_BILL_OR_SHIP_CUST_REQ
p_bill_to_ customer_name	IN	Varchar2		The name for the entered customer. Used to default the customer ID if not specified. Default: None Validate: None Error: AR_RAPI_CUS_NAME_INVALID

Parameter	Type	Data-type	Required*	Description
p_bill_to_ customer_ number	IN	Varchar2		The number for the entered customer. Used to default the customer ID if not specified. Default: None Validate: None Error: AR_RAPI_CUS_NAME_INVALID
p_bill_to_ location	IN	Varchar2		The location for the bill-to customer. Default: Defaulted from the primary bill-to customer location, if defined. Otherwise, null. Validate: This field is mandatory. Error: AR_DAPI_CUS_LOC_INVALID
p_bill_to_ contact_id	IN	Number		The contact identifier for the bill-to customer. Default: Defaulted from the bill-to customer site level, then customer level, if defined. Otherwise, null. Validate: Yes Error: AR_DAPI_BIll_CONTACT_NAME_INV AR_DAPI_CUS_CONTACT_INVALID
p_bill_to_ contact_first_ name	IN	Varchar2		The first name of contact for the bill-to customer. Default: Defaulted from bill-to customer site level, then customer level, if defined. Otherwise, null. Validate: This field is mandatory. Error: AR_DAPI_BIll_CONTACT_NAME_INV AR_DAPI_CUS_CONTACT_INVALID
p_bill_to_ contact_last_ name	IN	Varchar2		The last name of contact for the bill-to customer. Default: Defaulted from bill-to customer site level, then customer level, if defined. Otherwise, null. Validate: This field is mandatory. Error: AR_DAPI_BIll_CONTACT_NAME_INV AR_DAPI_CUS_CONTACT_INVALID

Parameter	Type	Data-type	Required*	Description
p_ship_to_ customer_id	IN	Number		The CUSTOMER_ID for the ship-to customer. Default: Defaulted from customer name/number. Null otherwise. Validate: Customer exists and has prospect code = CUSTOMER. Customer has a profile defined at the customer level. Either bill-to or ship-to customer must exist. Error: AR_RAPI_CUST_ID_INVALID AR_RAPI_CUS_NAME_INVALID AR_RAPI_CUS_NUM_INVALID AR_RAPI_CUS_NAME_NUM_INVALID AR_RAPI_CUS_NAME_NUM_IGN AR_DAPI_BILL_OR_SHIP_CUST_REQ
p_ship_to_ customer_name	IN	Varchar2		The name for the entered customer. Used to default the customer ID, if not specified. Default: None Validate: None Error: AR_RAPI_CUS_NAME_INVALID
p_ship_to_ customer_ number	IN	Varchar2		The number for the entered customer. Used to default the customer ID, if not specified. Default: None Validate: None Error: AR_RAPI_CUS_NAME_INVALID
p_ship_to_ location	IN	Varchar2		The location for the bill-to customer. Default: Defaulted from primary bill-to customer location, if defined. Otherwise, null. Validate: This field is mandatory. Error: AR_DAPI_CUS_LOC_INVALID
p_ship_to_ contact_id	IN	Number		The contact identifier for the bill-to customer. Default: Defaulted from bill-to customer site level, then from customer level, if it is defined. If not defined, then it is not defaulted. Validate: Yes Error: AR_DAPI_BIll_CONTACT_NAME_INV AR_DAPI_CUS_CONTACT_INVALID

Parameter	Type	Data-type	Required*	Description
p_ship_to_ contact_first_ name	IN	Varchar2		The first name of contact for the bill-to customer. Default: Defaulted from bill-to customer site level, then customer level, if defined. Otherwise, null. Validate: This field is mandatory. Error: AR_DAPI_BIll_CONTACT_NAME_INV AR_DAPI_CUS_CONTACT_INVALID
p_ship_to_ contact_last_ name	IN	Varchar2		The last name of contact for the bill-to customer. Default: Defaulted from bill-to customer site level, then customer level, if defined. Otherwise, null. Validate: This field is mandatory. Error: AR_DAPI_BIll_CONTACT_NAME_INV AR_DAPI_CUS_CONTACT_INVALID
p_term_id	IN	Number		Payment terms identifier for the transactions. You can override payment terms. Default: Following hierarchy is used to default payment terms: --Customer bill-to site level --Customer address level --Customer level Transaction type Validation: It should be a valid payment term. Error: AR_DAPI_TERM_NAME_INVALID AR_DAPI_TERM_ID_INVALID
p_term_name	IN	Varchar2		Payment terms name for the transactions. You can override payment terms. Default: Following hierarchy is used to default payment terms name: --Customer bill-to site level --Customer address level --Customer level Transaction type Validation: It should be a valid payment term. Error: AR_DAPI_TERM_NAME_INVALID AR_DAPI_TERM_ID_INVALID

Parameter	Type	Data-type	Required*	Description
p_salesrep_id	IN	Number		Salesperson identifier for the transactions. You can override salesperson. Default: Default the primary ID from the bill-to customer. If salescredits are required and no ID is defaulted from the bill-to customer, then p_salesrep_id is set to -3, which means "No sales credit". Validation: It should be a valid salesperson in the system. Error: AR_DAPI_SALESREP_NAME_INVALID AR_DAPI_SALESREP_ID_INVALID
p_salesrep_name	IN	Varchar2		Salesperson name for the transactions. You can override salesperson. Default: Default the primary from the bill-to customer. If salescredits are required and no salesperson is defaulted from the bill-to customer, then p_salesrep_name is set to -3, which means "No sales credit". Validation: It should be a valid salesperson in the system. Error: AR_DAPI_SALESREP_NAME_INVALID AR_DAPI_SALESREP_ID_INVALID
p_interface_ header_context	IN	Varchar2		Interface header context. Default: Null Validation: Null Error: Null
p_interface_ header_ attribute1 to p_ interface_ header_ attribute15	IN	Varchar2		Interface header attribute value Default: Null Validation: Null Error: Null
p_attribute_ category	IN	Varchar2		Descriptive Flexfield structure defining column. Default: Null Validation: It should be a valid structure. Error: Null

Parameter	Type	Data-type	Required*	Description
p_attribute1 to p_attribute15	IN	Varchar2		Descriptive Flexfield segment column. Default: Null Validation: It should be a valid segment. Error: Validate_Desc_Flexfield
p_global_attr_ cust_rec	IN	global_ attr_ rec_ type		This is a record type that contains all the 25 global descriptive flexfield segments and one global descriptive flexfield structure defining column. Default: None Validate: None Error:
p_document_ number	IN	Number		Value assigned to document receipt. Default: Null. Validate: User should not pass the value if the current document sequence is automatic. Document sequence value should not be entered if the Sequential Numbering profile option is set to Not Used. Error: AR_RAPI_DOC_SEQ_AUTOMATIC AR_RAPI_DOC_SEQ_NOT_EXIST_A AR_RAPI_DOC_SEQ_NOT_EXIST_P
p_ussgl_ transaction_code	IN	Varchar2		Code defined by public sector accounting. Default: None Validate: None Error: None
p_printing_ option	IN	Varchar2		Printing option for the invoice. Default: Default is print option of transaction type. Validate: Can be 'PRI' or 'NOT' Error: AR_DAPI_PO_INVALID

Parameter	Type	Data-type	Required*	Description
p_default_tax_ exempt_flag	IN	Varchar2		Tax exempt flag. You can enter value for the field only if the TAX: Allow Override of Customer Exception profile option is yes. Default: 'S' i.e. Standard Validate: From lookup table for lookup_type = 'TAX_CONTROL_FLAG' Error: AR_DAPI_STATUS_TRX_INVALID
p_status_trx	IN	Varchar2		Status of the transaction. This is a user-maintainable field and it can be defined in lookup table. Default: OP, can be CL, PEN, VD Validate: from lookup table for LOOKUP_TYPE = 'INVOICE_TRX_STATUS' Error: AR_DAPI_STATUS_TRX_INVALID
p_financial_ charges	IN	Varchar2		To indicate whether financial charges are calculated. Default: Null Validate: can be null, Y, N Error: AR_DAPI_FC_INVALID
p_agreement_id	IN	Number		Agreement associated with transaction for the customer. Default: Null Validate: Null Error: Null
p_special_ instructions	In	Varchar2		Any special instruction for the transaction, up to 240 characters. Default: Null Validation: Null Error: Null
p_comments				User's comments.
p_purchase_ order	In	Varchar2		Purchase order number. Default: Null Validation: Null Error: Null

Parameter	Type	Data-type	Required*	Description
p_purchase_ order_revision	In	Varchar2		Purchase order revision number. Default: Null Validation: Null Error: Null
p_purchase_ order_date	In	date		Purchase order date. Default: Null Validation: Null Error: Null
p_remit_to_ address_id	In	Number		Remit-to address ID for the customer Default: It is remit_to_address assigned to country, state, and postal code combination for the customer's address. Validate from the view: AR_ACTIVE_REMIT_TO_ ADDRESSES_V Error: AR_DAPI_LOC_SITE_NUM_IGN AR_DAPI_REMIT_ADDR_ID_INVD
p_sold_to_ customer_id	IN	Number		The customer_id for the sold-to customer. Default: bill_to_customer_id Validate: --Customer exists and has prospect code = CUSTOMER --Customer has a profile defined at customer level --Either bill-to or ship-to customer must exist Error: AR_DAPI_SOLD_CUST_COM_INVALID AR_DAPI_SOLD_CUS_IGN AR_DAPI_SOLD_CUST_ID_INVALID
p_sold_to_ customer_name	IN	Varchar2		The name for the entered/defaulted sold-to customer. Default: none Validate: 1. Customer exists and has prospect code = CUSTOMER 2. Customer has a profile defined at customer level 3. Either bill-to or ship-to customer must exist Error: AR_DAPI_SOLD_CUST_NAME_INVALID AR_DAPI_SOLD_CUST_COM_INVALID

Parameter	Type	Data-type	Required*	Description
p_sold_to_ customer_ number	IN	Varchar2		The number for the entered/defaulted sold-to customer. Default: None Validate: Customer exists and has prospect code = CUSTOMER. Customer has a profile defined at customer level. Either bill-to or ship-to customer must exist. Error: AR_DAPI_SOLD_CUST_NUM_INVALID AR_DAPI_SOLD_CUST_COM_INVALID
p_paying_ customer_id				The customer_id associated with the customer bank account assigned to your transaction. Default: Same as bill-to customer Validate: Customer exists and has prospect code = CUSTOMER. Customer has a profile defined at customer level. Either bill-to or ship-to customer must exist. Error: AR_DAPI_CUS_NAME_NUM_IGN AR_DAPI_PAY_CUST_ID_INVALID
p_paying_ customer_name				The name for the entered/defaulted paying customer. Default: None Validate: Customer exists and has prospect code = CUSTOMER. Customer has a profile defined at customer level. Either bill-to or ship-to customer must exist. Error: AR_DAPI_PAY_CUST_NAME_INVALID AR_DAPI_PAY_CUST_COM_INVALID
p_paying_ customer_ number				The number for the entered/defaulted paying customer. Default: None Validate: Customer exists and has prospect code = CUSTOMER. Customer has a profile defined at customer level. Either bill-to or ship-to customer must exist. Error: AR_DAPI_PAY_CUST_NUM_INVALID AR_DAPI_PAY_CUST_COM_INVALID

Parameter	Type	Data-type	Required*	Description
p_paying_ location				The location for the paying customer. Default: Null Validate: This field is mandatory. Error: AR_DAPI_CUS_LOC_INVALID
p_receipt_ method_id	IN	Number		Identifies the payment method of the transactions. Default: From receipt method name. Validate: Validation detailed in *Example* on page 4-27. Error: AR_RAPI_RCPT_MD_NAME_IGN AR_RAPI_RCPT_MD_ID_INVALID
p_receipt_ method_name	IN	Varchar2		The payment method name of the transactions. Default: None Validate: None Error: AR_RAPI_RCPT_MD_NAME_INVALID
p_cust_bank_ account_id	IN	Number		Customer bank account identifier. Default: None Validate: From AP_BANK_ACCOUNTS table. Error: AR_RAPI_CUS_BK_NAME_NUM_IGN AR_RAPI_CUS_BK_AC_ID_INVALID
p_cust_bank_ account_name	IN	Varchar2		Customer bank account name. Default: None Validate: From AP_BANK_ACCOUNTS table. Error: AR_RAPI_CUS_BK_AC_NAME_INVALID AR_RAPI_CUS_BK_AC_2_INVALID
p_cust_bank_ account_number	IN	Varchar2		Customer bank account number. Default: None Validate: From AP_BANK_ACCOUNTS table. Error: AR_RAPI_CUS_BK_AC_NUM_INVALID AR_RAPI_CUS_BK_AC_2_INVALID

Parameter	Type	Data-type	Required*	Description
p_start_date_ commitment	IN	Date		Start date of commitment. Default: Sysdate Validate: Based on end date, etc. Error: AR_TW_BAD_COMMITMT_DATE_ RANGE AR_TW_COMMIT_END_TRX_DATE AR_TW_BAD_DATE_COMMITMENT
p_end_date_ commitment	IN	Date		End date of commitment. Default: Null Validate: Based on start date, etc. Error: AR_TW_BAD_COMMITMT_DATE_ RANGE AR_TW_COMMIT_END_TRX_DATE AR_TW_BAD_DATE_COMMITMENT
p_amount	IN	Number		Deposit amount. Default: Cannot be negative. Validate: Based on start date, etc. This field is mandatory. Error: AR_DAPI_COMM_AMOUNT_NULL AR_TW_COMMIT_AMOUNT_NEGATIVE
p_inventory_id	IN	Number		Item ID of commitment. You can enter memo or item ID. Default: Null Validate: Based on MTL_SYSTEM_ITEMS_B table. Error: AR_DAPI_INV_ID_INVALID AR_DAPI_INV_MEMO_COM
p_memo_line_id	IN	Number		Memo line ID. You can enter memo or item ID. Default: Null Validate: Based on AR_MEMO_LINES table. Error: AR_DAPI_MEMO_NAME_INVALID AR_DAPI_MEMO_WRG AR_DAPI_INV_MEMO_COM

Parameter	Type	Data-type	Required*	Description
p_memo_line_ name	IN	Varchar2		Deposit amount. Default: Null Validate: Based on AR_MEMO_LINES table. Error: AR_DAPI_MEMO_NAME_INVALID AR_DAPI_MEMO_WRG
p_description	IN	Varchar2		Description of deposit. Default: Null Validate: Null Error: Null
p_comm_ interface_line_ context	IN	Varchar2		Interface line context for deposit. Default: Null Validation: Null Error: Null
p_comm_ interface_line_ attr1 to p_ comm_interface_ line_ attr15	In	Varchar2	NULL	Interface line attribute value for deposit. Default: Null Validation: Null Error: Null
p_comm_attr_ category	In	Varchar2	NULL	Descriptive Flexfield structure defining column for deposit lines. Default: Null Validation: It should be a valid structure. Error: Null
p_comm_attr1 to p_comm_attr15	In	Varchar2	NULL	Descriptive Flexfield segment column for deposit lines. Default: Null Validation: It should be a valid segment. Error: Validate_Desc_Flexfield

Parameter	Type	Data-type	Required*	Description
p_global_attr_ cust_lines_ rec	In	global_attr_ rec_type	NULL	This is a record type that contains all the 25 global descriptive flexfield segments for deposit lines and one global descriptive flexfield structure defining column. Default: None Validate: None Error: None
p_owner_id	In	Number	Null	ID of the commitment owner. Default: None Validate: Yes (same as customer contact). Error: N/A
p_owners_name	In	Number	Null	Name of the commitment owner. Default: None Validate: Yes (same as customer contact) Error: N/A
X_new_trx_ number	Out	Varchar2		New transaction number, if generated.
X_new_ customer_trx_id	Out	Varchar2		New CUSTOMER_TRX_ID of the deposit created.
X_new_ customer_trx_ line_ id	Out	Varchar2		New CUSTOMER_TRX_LINE_ID of the deposit created.
X_new_rowid	Out	Varchar2		Row ID of the deposit created.
X_new_status	Out	Varchar2		Status of the deposit created.

Example

Objective:

To create a deposit using a call to ar_deposit_api_pub.Create_deposit and passing a minimum number of Input parameters.

Entered parameters:

```
p_api_version           =1.0  ,
p_init_msg_list         ='F'  ,
```

```
p_deposit_number         = 'Your Deposit Number'
p_deposit_date           = sysdate,
p_currency_code          ='USD',
p_batch_source_id        = Choose a Valid Batch source ID
p_cust_trx_type_id       = Choose a Valid Transaction Type ID of class "Deposit'
p_class                  ='DEP'  i.e. Depsoit
p_bill_to_customer_number = Choose a Valid Customer Number
p_start_date_commitment   = sysdate
p_amount                  =  Choose deposit Amount
p_description             =  Your  Deposit Description
```

Before calling the APIs you should set up the application, responsibility and the user in the context of Oracle Application by calling the following FND API.

```
fnd_global.apps_initialize ( user_id =>'Your user id', resp_id => 'Your
Responsibility id', resp_appl_id => 'Your Application id');
```

For example:

```
fnd_global.apps_initialize ( user_id => 1318, resp_id => 50559, resp_appl_id
=> 222);
```

The API Call in this case would be:

```
DECLARE
l_return_status  VARCHAR2(1);
l_msg_count      NUMBER;
l_msg_data       VARCHAR2(240);
l_count          NUMBER;
l_new_trx_number      ra_customer_trx.trx_number%type;
l_new_customer_trx_id ra_customer_trx.customer_trx_id%type;
l_new_customer_trx_line_id    ra_customer_trx_lines.customer_trx_line_id%type;
l_new_rowid      VARCHAR2(240);
l_new_status      VARCHAR2(240);

BEGIN
fnd_global.apps_initialize ( user_id => 1318, resp_id => 50559, resp_appl_id =>
222);
ar_deposit_api_pub.CREATE_DEPOSIT(
```

a. Standard API parameters.

```
p_api_version      => 1.0,
p_init_msg_list    => FND_API.G_TRUE,
p_commit           => FND_API.G_TRUE,
p_validation_level => FND_API.G_VALID_LEVEL_FULL,
x_return_status    => l_return_status,
```

```
x_msg_count        => l_msg_count,
x_msg_data         => l_msg_data,
p_deposit_number   =>    'dapi_'||userenv('SESSIONID'),
p_deposit_date        =>     sysdate,
p_currency_code        = > 'USD',
p_batch_source_id      = > 'Choose a Valid Batch source ID',
p_cust_trx_type_id     => 'Choose a Valid Transaction Type ID of class Deposit',
p_class                => 'DEP' ,
p_bill_to_customer_number => 'Choose a Valid Customer Number',
p_start_date_commitment   => sysdate,
p_amount                  => 'Choose deposit Amount',
p_description             => 'Your  Deposit Description',
X_new_trx_number            =>l_new_trx_number,
X_new_customer_trx_id       =>l_new_customer_trx_id,
X_new_customer_trx_line_id  =>l_new_customer_trx_line_id,
X_new_rowid                 =>l_new_rowid,
X_new_status                =>l_new_status ) ;
IF l_msg_count  = 1 Then
```

b. There is one message raised by the API, so it has been sent out.

c. In the parameter x_msg_data, get it.

```
                dbms_output.put_line('l_msg_data '||l_msg_data);
ELSIF l_msg_count  > 1 Then
```

d. The messages on the stack are more than one, so call them in a loop.

e. And print the messages.

```
 LOOP
  IF nvl(l_count,0) < l_msg_count THEN
  l_count := nvl(l_count,0) +1 ;
  l_msg_data := FND_MSG_PUB.Get(FND_MSG_PUB.G_NEXT,FND_API.G_FALSE);

                   IF l_count = 1 THEN
                 dbms_output.put_line('l_msg_data 1 '||l_msg_data);
                   ELSIF l_count = 2 THEN
                dbms_output.put_line('l_msg_data 2 '||l_msg_data);
                   ELSIF l_count = 3 THEN
                dbms_output.put_line('l_msg_data 3 '||l_msg_data);
                   ELSIF l_count = 4 THEN
                dbms_output.put_line('l_msg_data 4 '||l_msg_data);
                   ELSIF l_count = 5 THEN
                dbms_output.put_line('l_msg_data 5 '||l_msg_data);
                   ELSIF l_count = 6 THEN
                dbms_output.put_line('l_msg_data 6 '||l_msg_data);
                   END IF;
```

```
                    dbms_output.put_line('l_msg_data '||to_char(l_count)||':
'||l_msg_data);
                  ELSE
                    EXIT;
                  END IF;
                END LOOP;
              END IF;
Commit;
END;
```

Depending on the message level threshold set the profile option FND_API_MSG_LEVEL_THRESHOLH, the messages put on the message stack may contain both the error messages and the warnings.

AR_DEPOSIT_API_PUB.insert_non_rev_salescredit

Description

This routine is called to assign nonrevenue sales credit to salespersons for a deposit. You can create as many of the nonrevenue credit assignments as you need.

This API routine has 4 output and 22 input parameters in total.

The following is the breakdown of the parameters:

Input

Standard API parameters:	4
Owners parameters:	22

Output

Standard API parameters:	3
Owners parameters:	0

The following table lists the API parameters.

Parameter	Type	Data-type	Required	Default Value	Description
p_api_version	IN	NUMBER	Yes		Used to compare version numbers of incoming calls to its current version number. Unexpected error is raised if version incompatibility exists. In the current version of the API, you should pass in a value of 1.0 for this parameter.
p_init_msg_ list	IN	VARCHAR2		FND_API.G_FALSE	Allows API callers to request that the API does initialization of the message list on their behalf.
p_commit	IN	VARCHAR2		FND_API.G_FALSE	Used by API callers to ask the API to commit on their behalf.
p_validation_ level	IN	NUMBER		FND_API.G_ VALID_LEVEL_ FULL	Not to be used currently as this is a public API.
x_return_ status	OUT	VARCHAR2			Represents the API overall return status.
x_msg_count	OUT	NUMBER			Number of messages in the API message list.
x_msg_data	OUT	VARCHAR2			This is the message in encoded format if x_msg_count=1.

The following table lists the parameters relevant to the deposit.

Parameter	Type	Data-type	Required	Description
p_deposit_number	IN	Varchar2	Null	Deposit number, same as trx_number for the transaction number. Default: None Validate: Yes Error: N/A

Parameter	Type	Data-type	Required	Description
p_customer_trx_id	IN	Number		Customer_trx_id of the deposit created. Default: None Validate: Yes Error: N/A
p_salesrep_number	IN	Number	Null	Salesperson number. Default: None Validate: Yes (same as customer contact). Error: N/A
p_salesrep_id	IN	Number		Salesrep_id of the salesperson. Default: None Validate: Yes Error: N/A
p_non_revenue_amount_split	IN	Number		Nonrevenue credit amount associated with salesperson. Default: None Validate: Yes Error: N/A
p_non_revenue_percent_split	IN	Number		Nonrevenue credit percent associated with salesperson. Default: None Validate: Yes Error: N/A
p_attribute_category	IN	Varchar2		Descriptive Flexfield structure defining column. Default: Null Validation: It should be a valid structure. Error: Null
p_attribute1 to p_attribute15	IN	Varchar2		Descriptive Flexfield segment column. Default: Null Validation: It should be a valid segment. Error: Validate_Desc_Flexfield

Example

Objective:

To create owner assignment using ar_deposit_api_pub. insert_non_rev_salescredit and passing a minimum number of Input parameters.

Entered parameters:

```
p_api_version                    =1.0 ,
p_init_msg_list                  ='F',
,p_customer_trx_id=>' Valid Customer Trx ID , Must be a deposit'
,p_salesrep_id=>-3 , means no Sales Rep
p_non_revenue_percent_split=>300
```

Before calling the APIs you should set up the application, responsibility and the user in the context of Oracle Application by calling the following FND API.

```
fnd_global.apps_initialize ( user_id =>'Your user id', resp_id => 'Your
Responsibility id', resp_appl_id => 'Your Application id');
```

For example:

```
fnd_global.apps_initialize ( user_id => 1318, resp_id => 50559, resp_appl_id
=> 222);
```

The API Call in this case would be:

```
DECLARE
l_return_status  VARCHAR2(1);
l_msg_count      NUMBER;
l_msg_data       VARCHAR2(240);
l_count          NUMBER;

BEGIN
fnd_global.apps_initialize ( user_id => 1318, resp_id => 50559, resp_appl_id =>
222);
  ar_deposit_api_pub.insert_non_rev_salescredit(
```

a. Standard API parameters.

```
p_api_version      => 1.0,
p_init_msg_list    => FND_API.G_TRUE,
p_commit           => FND_API.G_TRUE,
p_validation_level => FND_API.G_VALID_LEVEL_FULL,
x_return_status    => l_return_status,
```

```
                x_msg_count        => l_msg_count,
                x_msg_data         => l_msg_data,
                p_customer_trx_id  =>  ' Valid Customer Trx ID , Must be a
deposit',
                p_salesrep_id      =>  -3,
                p_non_revenue_amount_split  =>  300);

          dbms_output.put_line('return status '||l_return_status);
          dbms_output.put_line('l_msg_count '||to_char(l_msg_count));

IF l_msg_count  = 1 Then
```

b. There is one message raised by the API, so it has been sent out.

c. In the parameter x_msg_data, get it.

```
                dbms_output.put_line('l_msg_data '||l_msg_data);
ELSIF l_msg_count  > 1 Then
```

d. The messages on the stack are more than one, so call them in a loop.

e. And print the messages.

```
 LOOP
  IF nvl(l_count,0) < l_msg_count THEN
  l_count := nvl(l_count,0) +1 ;
  l_msg_data := FND_MSG_PUB.Get(FND_MSG_PUB.G_NEXT,FND_API.G_FALSE);

                   IF l_count = 1 THEN
                  dbms_output.put_line('l_msg_data 1 '||l_msg_data);
                   ELSIF l_count = 2 THEN
                 dbms_output.put_line('l_msg_data 2 '||l_msg_data);
                   ELSIF l_count = 3 THEN
                 dbms_output.put_line('l_msg_data 3 '||l_msg_data);
                   ELSIF l_count = 4 THEN
                 dbms_output.put_line('l_msg_data 4 '||l_msg_data);
                   ELSIF l_count = 5 THEN
                 dbms_output.put_line('l_msg_data 5 '||l_msg_data);
                   ELSIF l_count = 6 THEN
                 dbms_output.put_line('l_msg_data 6 '||l_msg_data);
                   END IF;
                   dbms_output.put_line('l_msg_data '||to_char(l_count)||':
'||l_msg_data);
                  ELSE
                    EXIT;
                  END IF;
```

```
            END LOOP;
          END IF;
Commit;
END;
```

Depending on the message level threshold set the profile option FND_API_MSG_LEVEL_THRESHOLD, the messages put on the message stack may contain both the error messages and the warnings.

Messages

Messages play an important role in the effectiveness of your API calls. The right message is raised at the right point to convey to you the exact error that has occurred or any warnings that have been raised.

In the Commitment (Deposit) API, all error messages and warnings raised during the execution are put on the message stack and can be retrieved by the user as described in *Exception Handling and Result Messages* on page 4-7.

WARNINGS AND ERRORS

The following table contains the list of all the error messages raised by the Commitment (Deposit) API.

Message Number	Message Code	Message Text	Type
294849	AR_DAPI_COMM_AMOUNT_NULL	The commitment amount requires a value.	E
294850	AR_DAPI_CUS_LOC_INVALID	The customer location is invalid.	E
294851	AR_DAPI_CUS_SITE_DFT_INVALID	The customer site use ID could not be defaulted.	E
294852	AR_DAPI_CUS_CONTACT_INVALID	The customer contact is invalid.	E
294853	AR_DAPI_CUST_NULL	A value for the customer ID, name, or number is required.	E
294854	AR_DAPI_COMM_BATCH_INVALID	The batch name or ID is invalid.	E
294855	AR_DAPI_TRANS_TYPE_ID_INVALID	The transaction type ID is invalid.	E
294856	AR_DAPI_TRANS_TYPE_INVALID	The transaction type is invalid.	E
294857	AR_DAPI_TERM_NAME_INVALID	The term name is invalid.	E
294858	AR_DAPI_TERM_ID_INVALID	The term ID is invalid.	E
294859	AR_DAPI_SALESREP_NAME_INVALID	The sales representative name is invalid.	E

Message Number	Message Code	Message Text	Type
294860	AR_DAPI_SALESREP_ID_ INVALID	The sales representative ID is invalid.	E
294861	AR_DAPI_BS_NAME_INVALID	The batch source name is invalid.	E
294862	AR_DAPI_BS_ID_INVALID	The batch source ID is invalid.	E
	AR_DAPI_BS_NAME_IGN	The batch source name has been ignored.	W
294863	AR_DAPI_SOLD_CUST_NAME_ INVALID	The sold-to customer name is invalid.	E
294864	AR_DAPI_SOLD_CUST_COM_ INVALID	The combination of sold-to customer name and number must be valid.	E
294865	AR_DAPI_PAY_CUST_NAME_ INVALID	The paying customer name is invalid.	E
	AR_DAPI_SOLD_CUST_DFT	The sold-to customer defaulted to the bill-to customer.	W
294866	AR_DAPI_PAY_CUST_COM_ INVALID	The combination of paying customer name and number must be valid.	E
294867	AR_DAPI_PAY_CUST_NUM_ INVALID	The paying customer number is invalid.	E
	AR_DAPI_CUS_NAME_NUM_ IGN	The paying customer name and number have been ignored.	W
294868	AR_DAPI_PAY_CUST_ID_ INVALID	The paying customer ID is invalid.	E
294869	AR_DAPI_SOLD_CUST_ID_ INVALID	The sold-to customer ID is invalid.	E
	AR_DAPI_SOLD_CUS_IGN	The sold-to customer name and number have been ignored.	W
	AR_DAPI_PO_INVALID	The printing option is invalid.	E
294871	AR_DAPI_STATUS_TRX_ INVALID	The transaction status is invalid.	E
294872	AR_DAPI_TAX_FLAG_INVALID	The default tax flag is invalid.	E
	AR_DAPI_NO_BATCH	A batch or a batch in the profile is required.	E
294874	AR_DAPI_MEMO_NAME_ INVALID	The memo name is invalid.	E

Message Number	Message Code	Message Text	Type
	AR_DAPI_MEMO_WRG	The memo ID, not the provided memo name, has been used.	W
	AR_DAPI_TRANS_TYPE_IGN	The type ID, not the provided type, has been used.	W
	AR_DAPI_INV_ID_INVALID	The inventory item ID is invalid.	E
	AR_DAPI_INV_MEMO_COM	Enter either a memo or inventory item ID.	E
294877	AR_DAPI_BILL_OR_SHIP_CUST_REQ	A bill-to or ship-to customer is required.	E
294878	AR_DAPI_BILL_CONTACT_NAME_INV	Both a first and last name are required for the bill-to contact.	E
294879	AR_DAPI_SHIP_CONTACT_NAME_INV	Both a first and last name are required for the ship-to contact.	E
	AR_DAPI_DEPOSIT_NO_NULL	A deposit number is required.	E
294881	AR_DAPI_FC_INVALID	The finance charges are invalid.	E
	AR_DAPI_LOC_SITE_NUM_IGN	The location site number has been ignored.	W
294882	AR_DAPI_REMIT_ADDR_ID_INVD	The remit-to address ID is invalid.	E
294883	AR_DAPI_CUST_LOC_SITE_NUM_INV	The customer location site number is invalid.	E
294884	AR_DAPI_REMIT_ADDRESS_DFT_ERR	The remit-to address did not successfully default.	E
294885	AR_DAPI_TRANS_TYPE_NULL	A value for either the transaction type or ID is required.	E
294886	AR_DAPI_BIll_CONTACT_COM_INV	The combination of the bill-to contact's first and last name must be valid.	E
294887	AR_DAPI_SHIP_CONTACT_COM_INV	The combination of the ship-to contact's first and last name must be valid.	E
294888	AR_DAPI_POST_COMMIT_ST	The deposit did not successfully post.	E
294889	AR_DAPI_INSERT_HEADER_ST	The header was not successfully inserted for the deposit.	E
	AR_DAPI_BILL_VAL_SHIP_IGN	The bill-to customer was defaulted from the ship-to customer because a value for the bill-to customer did not exist.	W
294890	AR_DAPI_LOC_INV	The location is invalid.	E

Message Number	Message Code	Message Text	Type
294891	AR_DAPI_SALESREP_ST	The salesperson was not successfully inserted for the deposit.	E
294892	AR_DAPI_SALESREP_NO_ID_NAME	The salesperson ID and name are required.	E
294893	AR_DAPI_NON_REV_AMT_PCT	A percentage or amount of nonrevenue sales credit is required.	E
294894	AR_DAPI_DEP_NO_ID_REQ	A deposit number or customer transaction ID is required.	
	AR_DAPI_DEP_NO_ING	The deposit number has been ignored.	W
294895	AR_DAPI_DEP_ID_INVALID	The customer transaction ID is invalid.	E
294896	AR_DAPI_DEP_NO_INVALID	The deposit number is invalid.	E
	AR_DAPI_REV_AMT_IGN	The nonrevenue sales credit amount has been ignored.	W

5

Invoice Creation API User Notes

Overview

This document outlines the use of Invoice Creation API. This API allows users to create an invoice using simple calls to PL/SQL functions.

The Invoice Creation API is not intended to replace the existing Transaction workbench, AutoInvoice, or the Transaction API program.

You can access this API in two ways:

- As standard PL/SQL servers-side routine calls
- Through Forms, utilizing the capability of Forms6 to have a procedure as its underlying base table

Before you begin....

Initialization of ARP_STANDARD and ARP_GLOBAL

Custom code that uses AR or HZ APIs will set the ORG_ID via dbms_application_info.set_client_info() and then call the APIs. The APIs in turn might access either ARP_STANDARD and ARP_GLOBAL, which initialize the global variables that are used across Oracle Receivables when the package is first called. Most of these global variable values are organization dependent, and the first such call sets the global variables based on the current ORG_ID.

If additional custom code then changes the ORG_ID via another call to dbms_application_info.set_client_info(), then the ORG context changes, ***but the ARP_STANDARD and ARP_GLOBAL context does not.***

In such cases, you should explicitly re-initialize the global variables by a call to these two public procedures:

1. ARP_GLOBAL.INIT_GLOBAL: For setting public variables in ARP_GLOBAL.
2. ARP_STANDARD.INIT_STANDARD: For setting public variables in ARP_STANDARD.

Major Features

Flexibility

Per Oracle API coding standards, this API lets you specify an ID or its associated value for any attribute that is an INPUT parameter of the API.

If both an ID and value have been specified, then the ID takes precedence over the value. This provides a wide degree of flexibility when using the API, both as a base table of the form and as a server-side routine call from the PL/SQL code.

The extensive defaulting mechanism for the input parameters ensures that you will be able to achieve the basic business needs of creating a invoice by calling the API with a minimum number of parameters. This gives you many options to achieve your requirements when you call the relevant API.

Modular Approach

The Invoice Creation API has been designed in a highly modular fashion, giving you code that is:

- Easy understand
- Easy to maintain
- Easy to extend
- Bulk enabled

Error Handling

The Invoice Creation API provides an extensive error-handling and error-reporting mechanism whereby all errors encountered in the Defaulting and Validation phases are reported in a global temporary error table. The calling program can get all the error messages from the error table.

Robust Validation

The validations that the Invoice Creation API performs are robust in nature. The APIs collect all the validation errors encountered and put them on the global temporary error table.

Debug Messages

The Invoice Creation API uses the Oracle Applications Logging Framework to log all debug messages in a central repository. Please query using module name, ar.plsql.InvoiceAPI.

See also: *Oracle Applications Logging Framework Guide*

Solution Outline

PL/SQL APIs

To create an invoice, you can call the following APIs:

- AR_INVOICE_API_PUB.CREATE_INVOICE: Creates multiple invoices in a batch.
- AR_INVOICE_API_PUB.CREATE_SINGLE_INVOICE: Create a single invoice and return customer_trx_id.

See *AR_INVOICE_API_PUB* on page 5-10.

Modular Approach

To modularize the Invoice Creation API, the basic structure of the API is divided into four parts:

1. Get all the default values from profiles and AR_SYSTEM_PARAMETERS table.
2. Populate four global temporary tables for Header, Lines, Distributions and Sales Credits from PL/SQL tables and Default values (if user has not entered).
3. Validate all the parameters entered by the user.
4. Call the entity handlers to perform the relevant task (such as Create).

This results in easy to understand and easy to maintain code. Any new functionality can be added by a simple code plug-in at each of the four parts.

Defaulting

In general, the various parameters in the API call, if not entered, get defaulted based on the following:

- Values set in the AR_SYSTEM_PARAMETERS table entered through the System Options form
- Relevant profile option values
- Values set through various setup forms.

Depending on the above three factors and the exact business requirement, the minimum number of parameters that may be required to perform certain business tasks may vary.

Null values are defaulted for the parameters that could not be defaulted by the API defaulting routines.

For various attributes of the business objects, you can pass either the ID or the value of the attribute.

If you specify only the value, then the value is used to derive the ID; otherwise, the ID (if specified) is taken directly. If you specify both the ID and the value, then the ID takes precedence over the value and a warning message informs you of this.

Exception Handling and Result Messages

The Invoice Creation APIs return these types of information to their calling programs:

- Overall status
- Messages describing the operations performed or errors encountered by the APIs
- Some output values that the API caller might want to use (this is different for different API routines and is described in *API Usage* on page 5-10)

Return Status

The return status (x_return_status) of the API informs the caller about the result of the operation (or operations) performed by the API. The different possible values for an API return status are:

- Success (FND_API.G_RET_STS_SUCCESS)
- Unexpected error (FND_API.G_RET_STS_UNEXP_ERROR)

The following section describes the different values of return status and their meanings.

Success

A Success return status means that the API was able to perform all the operations requested by its caller.

However, a Success status does *not* mean that an invoice is created successfully. You must check the error table AR_TRX_ERRORS_GT for any validation errors. The invoice will not be created if any related records exist in this table.

Unexpected error

An unexpected error status means that the API has encountered an error condition it did not expect or could not handle. In this case, the API is unable to continue with

its regular processing. Examples of such errors are irrecoverable data inconsistency errors, memory errors, and programming errors (such as attempting a division by zero).

In most cases, only system administrators or application developers can fix these unexpected errors.

Messages

The APIs put all error messages in the global temporary table AR_TRX_ERRORS_GT. Programs calling these APIs can then get the messages from this table and process them by issuing them, loading them into a database table, or writing them to a log file.

Debug Messages

The API uses Oracle Applications Logging Framework to log debug messages in a central repository.

The debugging can be enabled by the setting the following profile options:

1. FND: Debug Log Enabled(AFLOG_ENABLED) to 'Y'.
2. FND: Debug Log Level (AFLOG_LEVEL) to 'Statement'.

Once the above parameters are set, the message will be logged in the FND repository. The API to log accepts log level, module name, and the actual text.

An example is given below:

```
FND_LOG.STRING(P_LOG_LEVEL,P_MODULE_NAME, P_MESSAGE);
```

All Invoice Creation API debug messages use a module name of 'ar.plsql.InvoiceAPI'.

See also: *Oracle Applications Logging Framework Guide*

Calling Program Context

The program calling these APIs should have set up the application, responsibility, and user in the context of Oracle Application.

If the calling program does not set up this context, then it can be done programmatically by calling the following FND API.

```
fnd_global.apps_initialize (      user_id,
                                  resp_id,
                                  resp_appl_id,
                                  security_group_id);
```

API Usage

AR_INVOICE_API_PUB

The API contains 2 public procedures to create invoice either in batch mode with multiple invoices or a single invoice. The input parameters are same for both the procedures and explained in the following section.

- CREATE_INVOICE procedure needs to be called to create multiple invoices in a batch. The procedure gives back a global record type structure which contains batch_id to retrieve the necessary data from transaction tables. The structure is defined in the package specification of ar_invoice_api_pub. Please refer to *Example for Creating Multiple Invoices in a Batch* on page 5-22 for usage.

```
TYPE api_outputs_type IS RECORD
(
batch_id NUMBER DEFAULT NULL
        );
```

- CREATE_SINGLE_INVOICE procedure needs to be called to create a single invoice. The procedure return customer_trx_id as out parameter. Please refer to *Example for Creating a Single Invoice* on page 5-25 for usage.

API Parameters

The API accepts the following parameters:

```
p_api_version                  IN                 NUMBER,
p_init_msg_list                IN                 VARCHAR2 := FND_API.G_FALSE,
p_commit                       IN                 VARCHAR2 := FND_API.G_FALSE,
p_batch_source_rec             IN                 batch_source_rec_type,
p_trx_header_tbl               IN                 trx_header_tbl_type,
p_trx_lines_tbl                IN                 trx_line_tbl_type,
p_trx_dist_tbl                 IN                 trx_dist_tbl_type,
p_trx_salescredits_tbl         IN                 trx_salescredits_tbl_type,
x_customer_trx_id              OUT NOCOPY         NUMBER,
x_return_status                OUT NOCOPY         VARCHAR2,
x_msg_count                    OUT NOCOPY         NUMBER,
x_msg_data                     OUT NOCOPY         VARCHAR2,
```

The following table shows the list of standard API parameters.

Parameter	Type	Data Type	Required	Default Value	Description
p_api_version	IN	NUMBER	Yes	1.0	Compare version numbers of incoming calls to its current versions
p_init_msg_list	IN	VARCHAR2		FND_API.G_FALSE	Allow API callers to request that API does initialize the message list on their behalf.
p_commit	IN	VARCHAR2		FND_API.G_FALSE	Used by API callers to ask the API to commit on their behalf.
x_customer_trx_id	OUT	NUMBER			Returns customer_trx_id in case it is called for creating a single invoice. This parameter works only with CREATE_SINGLE_INVOICE procedure.
x_return_status	OUT	VARCHAR2			Represent the API status.
x_msg_count	OUT	NUMBER			Number of messages in the PI message list (not used by this API).
x_message_data	OUT	VARCHAR2			Message in case API encounters any unexpected error.

P_BATCH_SOURCE_REC Parameter

The P_BATCH_SOURCE_REC parameter is of PL/SQL record type, and has the following attributes, as described in this table:

Attribute Name	Data Type	Required	Default Value	Description
batch_source_id	NUMBER		Null	If batch_source_id is null then value will be derived from AR_RA_BATCH_SOURCE profile option. In case the value is passed then it will be validated against ra_batch_sources. Only 'Manual' batch sources are allowed.

Attribute Name	Data Type	Required	Default Value	Description
default_date	DATE		Null	If the value is null then Sysdate will be taken.

P_TRX_HEADER_TBL Parameter

The P_TRX_HEADER_TBL parameter is of PL/SQL table type TRX_HEADER_REC_TYPE.

TRX_HEADER_REC_TYPE has the following attributes, as described in this table:

Attribute Name	Data Type	Required	Default Value	Description
trx_header_id	NUMBER	Yes		Identifier for the Invoice header record. This must be unique for each record. This column can be generated based on a sequence or any number value. The value does not get recorded into any table.
trx_number	VARCHAR2(30)		Null	This is the transaction number for the invoice. This field should not be populated if the batch source has Copy Document Sequence Number to Transaction Number checked or if Automatic Transaction Numbering is enabled.
trx_date	DATE		Null	Invoice Date. If no value is passed then p_batch_source_rec.default_date is used. If that too is not passed then sysdate is used.
gl_date	DATE		Null	General ledger Date. If no date is passed then p_batch_source_rec.default_date is used. If that too is not passed then sysdate is used.
trx_currency	VARCHAR2(30)		Null	Transaction Currency. If not populated then ar_system_parameters is used to retrieve it. The currency if populated must be active as of the transaction date.
cust_trx_type_id	NUMBER		Null	Transaction Type Identifier. Only 'INV' type is allowed. Validated against ra_cust_trx_types. If not populated, then it is retrieved from the batch source.
bill_to_customer_id	NUMBER	Yes		Bill To Customer ID. This must exist in hz_cust_accounts table. The customer must be an active ('A') customer. Validated against hz_cust_accounts.cust_account_id.

Attribute Name	Data Type	Required	Default Value	Description
bill_to_account_number	VARCHAR2(30)		Null	Bill To Customer Number. If both Bill To Customer ID and Bill To Customer Number are passed, then the former will take precedence.Validated against hz_cust_accounts.account_number.
bill_to_customer_name	VARCHAR2 (260)		Null	Bill To Customer Name. If all three are passed, the precedence is as follows: Customer ID, Customer Number, then Customer Name.
bill_to_contact_id	NUMBER		Null	Bill To Customer Contact ID. This must exist for the Bill To Customer and Bill To Address combination.
bill_to_address_id	NUMBER		Null	Bill To Address ID. This must exist in hz_cust_acct_sites for the populated Bill To Customer ID
bill_to_site_use_id	NUMBER		Null	Bill To Site use ID. The site use ID must exist in combination with Ship To Customer ID, Ship To Address ID.
ship_to_customer_id	NUMBER			Ship To Customer ID. This must exist in hz_cust_accounts table.
ship_to_account_number	VARCHAR2(30)		Null	Ship To Customer Number. If both Bill To Customer ID and Ship To Customer Number are passed, then the former will take precedence.
ship_to_customer_name	VARCHAR2 (260)		Null	Ship To Customer Name. If all three are passed, the precedence is as follows: Customer ID, Customer Number, then Customer Name.
ship_to_contact_id	NUMBER		Null	Ship To Customer Contact ID. This must exist for the Ship To Customer and Ship To Address combination.
ship_to_address_id	NUMBER		Null	Ship To Address ID. This must exist in hz_cust_acct_sites for the populated Ship To Customer ID.
ship_to_site_use_id	NUMBER		Null	Ship To Site use ID. The site use ID must exist in combination with Ship To Customer ID, Ship To Address ID.
sold_to_customer_id	NUMBER		Null	Ship To Customer ID. This must exist in hz_cust_accounts table.

Attribute Name	Data Type	Required	Default Value	Description
term_id	NUMBER		Null	Payment Terms Identifier. The Term ID must be valid for the transaction date. If not populated, then it is retrieved from ra_terms based on bill_to_customer_id and bill_to_site_use_id.
primary_salesrep_id	NUMBER		Null	Primary Salesrep ID. This is required if Salesperson check box is checked in the System Options form. If not populated, then it is derived based on bill-to_customer_id and bill_to_site_use_id.
primary_salesrep_name	VARCHAR2 (240)		Null	Primary Salesrep name. If both salesrep ID and name are passed, then Salesrep ID will take precedence.
exchange_rate_type	VARCHAR2(60)		Null	Exchange Rate Type. This must exist in gl_daily_conversion_types. Required if trx_currency is different from functional currency. If not populated, then it will derive from gl.
exchange_date	DATE		Null	Exchange Date. Required if trx_currency is different from functional currency. If not populated, then it will derive from gl.
exchange_rate	NUMBER		Null	Exchange Rate. This should be entered only if transaction currency is different from the functional currency and exchange rate type is 'User'.
territory_id	NUMBER		Null	Territory ID. If not populated, then it is defaulted based on the following hierarchy: -The Bill To site use -The Ship To Site Use -The Primary Salesrep's territory depending on the value of the DEFAULT_TERRITORY system option
remit_to_address_id	NUMBER		Null	Remit To Address ID. If not populated, then it is defaulted based on country, state, and postal code of bill_to_site_use_id. If populated, then validated against ar_active_remit_to_addresses_v.
invoicing_rule_id	NUMBER		Null	Invoicing Rule ID. Valid values are -2 and -3. If you enter a value here, then you must populate accounting rule for line type = 'LINE'.

Attribute Name	Data Type	Required	Default Value	Description
printing_option	VARCHAR2(20)		Null	Revenue Accounting lookup code for INVOICE_PRINT_OPTIONS. Valid codes are PRI - Print and NOT - Do not Print.
purchase_order	VARCHAR2(50)		NULL	Purchase Order Number for this transaction.
purchase_order_revision	VARCHAR2(50)		Null	Purchase Order Revision. This must not be entered if purchase order is not populated.
purchase_order_date	DATE		Null	Purchase Order date. This must not be entered if purchase order is not populated.
comments	VARCHAR2 (240)		Null	Comments. Value can be printed on an invoice using the Print Invoice view.
internal_notes	VARCHAR2 (240)		Null	Stores the special instruction. Value can be printed on an invoice using the Print Invoice view.
finance_charges	VARCHAR2(1)		Null	Indicates if finance charges are included. Y for yes, N otherwise.
receipt_method_id	NUMBER		Null	This is the payment identifier for this transaction. If not populated, then it is defaulted based on the following hierarchy: 1.primary receipt method of parent primary bill to site 2. Primary receipt method of the parent customer 3. Primary receipt method of the bill to site 4. Primary receipt method of the bill-to customer
related_customer_trx_id	NUMBER		Null	Customer transaction ID of the document to which this transaction is related. Validated against ra_customer_trx_all.customer_trx_id. Not required for on-account credit memos.
agreement_id	NUMBER		Null	Customer Agreement identifier for this transaction. If not populated, then it will be defaulted from the commitment. Must exist in SO_AGREEMENTS. (For future use.)
ship_via	VARCHAR2(30)		Null	Ship Via Code. If populated, then validated against org_freight.
ship_date_actual	DATE		Null	Ship Date
waybill_number	VARCHAR2(50)		Null	Waybill Number

Attribute Name	Data Type	Required	Default Value	Description
fob_point	VARCHAR2(30)		Null	Free on Board Point. Validated against AR_ LOOKUPS.LOOKUP_TYPE='FOB'.
customer_bank_ account_id	NUMBER		Null	Customer bank account ID. If the payment method is Automatic, then it is required. If not populated, then it will be default using the following hierarchy. 1.primary bank account assigned to the primary site. 2. primary bank assigned to parent customer. 3. primary bank assigned to bill to site use. 4. primary bank assigned to bill to customer.
default_ussgl_ transaction_code	VARCHAR2(30)		Null	Default value for the USSGL Transaction Code Flexfield (for future use)
status_trx	VARCHAR2(30)		Null	The status of the transaction. If not populated, then defaulted from Transaction Type. Valid values are 'OP', 'CL', 'PEN', 'VD.
paying_ customer_id	NUMBER		Null	This column is required when the RECEIPT_ METHOD_ID column is an automatic payment method.
paying_site_use_ id	NUMBER		Null	This column is required when the RECEIPT_ METHOD_ID column is an automatic payment method.
doc_sequence_ value	NUMBER(15)		Null	Document Number. Must not exist in Oracle Receivables.
attribute_ category	VARCHAR2(30)		Null	Descriptive flexfield structure definition column.
attribute1 - 10	VARCHAR2 (150)		Null	Descriptive flexfield segment.
global_attribute_ category	VARCHAR2(30)		Null	Reserved for country-specific functionality. (For future use.)
global_ attribute1-30	VARCHAR2 (150)		Null	Reserved for country-specific functionality. (For future use.)
interface_header_ context	VARCHAR2(30)		Null	Interface header context.
interface_header_ attribute1 - 15	VARCHAR2(30)		Null	Interface header attribute value.

P_TRX_LINES_TBL Parameter

The P_TRX_LINES_TBL parameter is of PL/SQL table type TRX_LINE_REC_TYPE.

TRX_LINE_REC_TYPE has the following attributes, as described in this table:

Attribute Name	Data Type	Required	Default Value	Description
trx_header_id	NUMBER	Yes		Identifier for the Invoice header record. This column can be generated based on a sequence or any number value. The value does not get recorded into any table. This column ties back with P_TRX_HEADER_TBL.
trx_line_id	NUMBER	Yes		Identifier for the Invoice lines record. This column can be generated based on a sequence or any number value. The value does not get recorded into any table.
link_to_trx_line_id	NUMBER			This column is required only if line type is 'TAX' and 'FREIGHT' (if it is associated with any line).
line_number	NUMBER	Yes		Line number of the invoice
reason_code	VARCHAR2(30)			Reason code. Validated against AR_ LOOKUPS.LOOKUP_TYPE = 'INVOICING_ REASON'.
inventory_item_id	NUMBER			Inventory item identifier. Mutually exclusive with the column MEMO_LINE_ID. Validated against mtl_system_items.inventory_item_id and invoice_enabled_flag = 'Y'.
description	VARCHAR2 (240)			Line description. Required if inventory_item_id or memo_line_id is not provided.
quantity_ordered	NUMBER			Quantity of an order
quantity_invoiced	NUMBER			Quantity of invoice line. Required for Invoices.
unit_standard_ price	NUMBER			List price per unit.
unit_selling_price	NUMBER			Selling price per unit for a transaction line. Required for Invoices.
sales_order	VARCHAR2(50)			Sales order number for this transaction.
sales_order_line	VARCHAR2(30)			Sales order line number for this transaction.
sales_order_date	DATE			Sales order date for this transaction.

Attribute Name	Data Type	Required	Default Value	Description
accounting_rule_id	NUMBER			Accounting rule identifier. Must provide a value for invoice with Rule ID. Validated against RA_ RULES.
line_type	VARCHAR2(20)	Yes		Receivables lookup code for STD_LINE_TYPE.
attribute_category	VARCHAR2(30)			Descriptive flexfield structure definition column.
attribute1-15	VARCHAR2 (150)			Descriptive flexfield segment.
rule_start_date	DATE			First GL date of the invoice. Only used for invoice with rules.
interface_line_ context	VARCHAR2(30)			Interface line context.
interface_line_ attribute1-15	VARCHAR2(30)			Interface line attribute value.
sales_order_source	VARCHAR2(50)			The source of the sales order.
amount	NUMBER			Transaction line revenue amount. If line type = 'FREIGHT' or 'TAX', then amount must be populated.
tax_precedence	NUMBER			Tax precedence for a tax line. Used to compute tax compounding. **Note:** Required for line type = 'LINE' and 'FREIGHT'.
tax_rate	NUMBER			Tax rate for a line. Required for TAX line in case amount is not populated.
memo_line_id	NUMBER			Memo line description identifier. Mutually exclusive with the column INVENTORY_ITEM_ ID. Not required for 'TAX' and 'FREIGHT' lines.
uom_code	VARCHAR2(3)			Unit of measure code. Required for line type of 'LINE' and has a item on the line. Not required for 'TAX' and 'FREIGHT' lines.
default_ussgl_ transaction_code	VARCHAR2(30)			Default value for the USSGL Transaction Code Flexfield. (For future use.)
default_ussgl_trx_ code_context	VARCHAR2(30)			Default context value for the USSGL Transaction Code Flexfield. (For future use.)
vat_tax_id	NUMBER			Unique identifier for AR_VAT_TAX. Required for 'TAX' Lines.

Attribute Name	Data Type	Required	Default Value	Description
tax_exempt_flag	VARCHAR2(1)			Tax Lines are controlled by the lookup (TAX_ CONTROL_FLAG), which allows for standard tax, exempt tax, and required tax.
tax_exempt_ number	VARCHAR2(80)			Exemption certificate number for item lines that have TAX_EXEMPT_FLAG set to E for exempt.
tax_exempt_ reason_code	VARCHAR2(30)			Tax Exempt Reason, for item lines that have tax_ exempt_flag set to "E" (exempt).
movement_id	NUMBER			Intrastate movement ID number that is tied to the shipment information.
global_ attribute1-20 -20	VARCHAR2 (150)			Reserved for country-specific functionality. (For future use.)
global_attribute_ category	VARCHAR2(30)			Reserved for country-specific functionality. (For future use.)
amount_includes_ tax_flag	VARCHAR2(1)			Y indicates tax is inclusive. N indicates tax is exclusive. NULL for lines indicates tax cannot be overridden or tax is a tax group. Cannot be NULL for tax types. Must be NULL for other types.
warehouse_id	NUMBER			Foreign key to the HR_ORGANIZATIONS table. The warehouse identifies the ship-from location and can be used to control taxation. Within the US, the Warehouse ID is important when calculating tax on the origin/modified origin state sales tax. Outside the US you can use tax groups and conditions to build a schedule of multiple conditional taxes based on both the ship-from and ship-to county/county/state or provinces.
contract_line_id	NUMBER			Identifies the contract line from Oracle Contracts Core that is associated with this line.
source_data_ key1-5	VARCHAR2 (150)			Identifies source data from original system.
invoiced_line_ acctg_level	VARCHAR2(15)			Identifies accounting level for invoiceable lines in original system.

P_TRX_DIST_TBL Parameter

The P_TRX_DIST_TBL parameter is of PL/SQL table type TRX_DIST_REC_TYPE.

TRX_DIST_REC_TYPE has the following attributes, as described in this table:

Attribute Name	Data Type	Required	Default Value	Description
trx_line_id	NUMBER	Yes		Identifier for the Invoice lines record. This column can be generated based on a sequence or any number value. The value does not get recorded into any table.
trx_header_id	NUMBER			Identifier for the Invoice header record. This column can be generated based on a sequence or any number value. The value does not get recorded into any table. This column ties back with P_TRX_HEADER_TBL. Required in case of 'REC' distribution type.
trx_dist_id	NUMBER	Yes		Identifier for the Distribution record. This column can be generated based on a sequence or any number value. The value does not get recorded into any table.
account_class	VARCHAR2(20)	Yes		Account Class for this distribution. Freight, Receivable, Revenue, AutoInvoice Clearing, Tax, Unbilled Receivable, Unearned Revenue, or Charges account type.
amount	NUMBER			Amount of this record in the foreign currency. Required if percentage is not passed.
acctd_amount	NUMBER			Amount of this record in the functional currency. If not populated, then it will be populated based on amount passed.
percent	NUMBER			Percent of the line amount represented by this record. Required if amount is not passed.
code_ combination_id	NUMBER	Yes		Code combination ID for Accounting Flexfield. Validated against gl_code_combinations.code_ combination_id.
attribute_ category	VARCHAR2(30)			Descriptive flexfield structure definition column.
attribute1-15	VARCHAR2(150)			Descriptive flexfield segment.
comments	VARCHAR2(240)			Comment about the revenue distribution.

P_TRX_ SALESCREDITS _TBL Parameter

The P_TRX_SALESCREDITS_TBL parameter is of PL/SQL table type TRX_ SALESCREDITS_REC_TYPE.

TRX_ SALESCREDITS _REC_TYPE has the following attributes, as described in this table:

Attribute Name	Data Type	Required	Default Value	Description
trx_salescredit_ id	NUMBER	Yes		Identifier for the Salesperson on the lines record. This column can be generated based on a sequence or any number value. The value does not get recorded into any table.
trx_line_id	NUMBER	Yes		Identifier for the Invoice lines record. This column can be generated based on a sequence or any number value. The value does not get recorded into any table.
salesrep_id	NUMBER	Yes		Identifies the salesperson for this sales credit assignment. Validated against ra_ salesreps.salesrep_id.
salesrep_ number	VARCHAR2(30)			Salesrep Number assignment. Validated against ra_salesreps.salesrep_number. If both number and ID is passed, then ID will take precedence.
sales_credit_ type_name	VARCHAR2(30)			Sales Credit Type Name. Validated against so_ sales_credit_types.name.
sales_credit_ type_id	NUMBER	Yes		Sales Credit Type Identifier. Validated against so_sales_credit_types.sales_credit_type_id. If both ID and name are passed, then ID will take precedence.
salescredit_ amount_split	NUMBER			The amount of revenue/non-revenue credit for this salesperson/customer. Required if salescredit_percent_split is not passed.
salescredit_ percent_split	NUMBER			The percent of revenue/non-revenue credit for this salesperson/customer. Required if salescredit_amount_split is not passed.
attribute_ category	VARCHAR2(30)			Descriptive flexfield structure definition column.
attribute1-15	VARCHAR2(150)			Descriptive flexfield segment.

Example for Creating Multiple Invoices in a Batch

Objective:

To create an Invoice using a call to ar_invoice_api_pub.Create_invoive and passing a minimum number of Input parameters.

1. DECLARE

```
l_return_status         varchar2(1);
l_msg_count             number;
l_msg_data              varchar2(2000);
l_batch_id              number;


l_batch_source_rec      ar_invoice_api_pub.batch_source_rec_type;
l_trx_header_tbl        ar_invoice_api_pub.trx_header_tbl_type;
l_trx_lines_tbl         ar_invoice_api_pub.trx_line_tbl_type;
l_trx_dist_tbl          ar_invoice_api_pub.trx_dist_tbl_type;
l_trx_salescredits_tbl  ar_invoice_api_pub.trx_salescredits_tbl_type;

CURSOR cBatch IS
      select customer_trx_id
      from ra_customer_trx_all
      where batch_id = l_batch_id;

CURSOR cValidTxn IS
       SELECT trx_header_id
       From ar_trx_header_gt
       WHERE trx_header_id not in (
             SELECT trx_header_id
             FROM ar_trx_errors_gt);
```

2. BEGIN

   a. Set applications context if not already set.

```
fnd_global.apps_initialize(1318, 50559, 222,0);
```

   b. Populate header information.

```
l_trx_header_tbl(1).trx_header_id := 101;
l_trx_header_tbl(1).trx_number := 'Test Invoice API';
l_trx_header_tbl(1).bill_to_customer_id := 1006;
l_trx_header_tbl(1).cust_trx_type_id := 2376;
```

   c. Populate batch source information.

```
l_batch_source_rec.batch_source_id := 1188;
```

d. Populate line 1 information.

```
l_trx_lines_tbl(1).trx_header_id := 101;
l_trx_lines_tbl(1).trx_line_id := 101;
l_trx_lines_tbl(1).line_number := 1;
l_trx_lines_tbl(1).memo_line_id := 8;
l_trx_lines_tbl(1).quantity_invoiced := 10;
l_trx_lines_tbl(1).unit_selling_price := 12;
l_trx_lines_tbl(1).line_type := 'LINE';
```

e. Populate line 2 information.

```
l_trx_lines_tbl(2).trx_header_id := 101;
l_trx_lines_tbl(2).trx_line_id := 102;
l_trx_lines_tbl(2).line_number := 2;
l_trx_lines_tbl(2).description := 'Test';
l_trx_lines_tbl(2).quantity_invoiced := 12;
l_trx_lines_tbl(2).unit_selling_price := 12;
l_trx_lines_tbl(2).line_type := 'LINE';
```

f. Populate freight information and link it to line 1.

```
l_trx_lines_tbl(3).trx_header_id := 101;
l_trx_lines_tbl(3).trx_line_id := 103;
l_trx_lines_tbl(3).link_to_trx_line_id := 101;
l_trx_lines_tbl(3).line_number := 1;
l_trx_lines_tbl(3).line_type := 'FREIGHT';
l_trx_lines_tbl(3). amount := 25;
```

g. Call the invoice api to create multiple invoices in a batch.

```
AR_INVOICE_API_PUB.create_invoice(
  p_api_version           => 1.0,
  p_batch_source_rec      => l_batch_source_rec,
  p_trx_header_tbl        => l_trx_header_tbl,
  p_trx_lines_tbl         => l_trx_lines_tbl,
  p_trx_dist_tbl          => l_trx_dist_tbl,
  p_trx_salescredits_tbl  => l_trx_salescredits_tbl,
  x_return_status         => l_return_status,
  x_msg_count             => l_msg_count,
  x_msg_data              => l_msg_data);

IF l_return_status = fnd_api.g_ret_sts_error OR
   l_return_status = fnd_api.g_ret_sts_unexp_error THEN
```

```
    dbms_output.put_line('unexpected errors found!');
  ELSE
```

h. Check if there are record exist in error table. If no records exist for a trx_header_id, then only Invoice will create in the system; otherwise not.

```
For cValidTxnRec IN cvalidTxn
loop
       IF (ar_invoice_api_pub.g_api_outputs.batch_id IS NOT NULL) THEN
       dbms_output.put_line('Invoice(s) suceessfully created!') ;
       dbms_output.put_line('Batch ID: ' || ar_invoice_api_pub.g_api_
outputs.batch_id);
       l_batch_id :=  ar_invoice_api_pub.g_api_outputs.batch_id;
```

i. To see all customer_trx_id for this batch:

```
for cBatchRec in cBatch
    loop
        dbms_output.put_line ( 'Cust Trx Id '||  cBatchRec.customer_trx_id
);
    end loop;
      ELSE
       dbms_output.put_line('Errors found!');
      END IF;
   End loop;
   END IF;

END;
/
```

j. See all the validation errors.

```
SET LINESIZE 200
COLUMN trx_header_id HEADING 'Header ID'
COLUMN trx_line_id   HEADING 'Line ID'
COLUMN error_message HEADING 'Message'
COLUMN invalid_value HEADING 'Invalid Value'
COLUMN trx_header_id FORMAT 9999999
COLUMN trx_line_id   FORMAT 9999999
COLUMN error_message FORMAT a30
COLUMN invalid_value FORMAT a20
SELECT trx_header_id, trx_line_id, error_message, invalid_value
FROM ar_trx_errors_gt;
```

Example for Creating a Single Invoice

Objective:

To create an Invoice using a call to ar_invoice_api_pub.Create_single_invoive and passing a minimum number of Input parameters.

1. DECLARE

```
l_return_status         varchar2(1);
l_msg_count             number;
l_msg_data              varchar2(2000);
l_batch_id              number;
l_cnt                   number := 0;


l_batch_source_rec      ar_invoice_api_pub.batch_source_rec_type;
l_trx_header_tbl        ar_invoice_api_pub.trx_header_tbl_type;
l_trx_lines_tbl         ar_invoice_api_pub.trx_line_tbl_type;
l_trx_dist_tbl          ar_invoice_api_pub.trx_dist_tbl_type;
l_trx_salescredits_tbl  ar_invoice_api_pub.trx_salescredits_tbl_type;
l_customer_trx_id        number;
```

2. BEGIN

   a. Set applications context if not already set.

```
fnd_global.apps_initialize(1318, 50559, 222,0);
```

   b. Populate header information.

```
l_trx_header_tbl(1).trx_header_id := 101;
l_trx_header_tbl(1).trx_number := 'Test Invoice API';
l_trx_header_tbl(1).bill_to_customer_id := 1006;
l_trx_header_tbl(1).cust_trx_type_id := 2376;
```

   c. Populate batch source information.

```
l_batch_source_rec.batch_source_id := 1188;
```

   d. Populate line 1 information.

```
l_trx_lines_tbl(1).trx_header_id := 101;
l_trx_lines_tbl(1).trx_line_id := 101;
l_trx_lines_tbl(1).line_number := 1;
l_trx_lines_tbl(1).memo_line_id := 8;
l_trx_lines_tbl(1).quantity_invoiced := 10;
```

```
l_trx_lines_tbl(1).unit_selling_price := 12;
l_trx_lines_tbl(1).line_type := 'LINE';
```

e. Populate line 2 information.

```
l_trx_lines_tbl(2).trx_header_id := 101;
l_trx_lines_tbl(2).trx_line_id := 102;
l_trx_lines_tbl(2).line_number := 2;
l_trx_lines_tbl(2).description := 'Test';
l_trx_lines_tbl(2).quantity_invoiced := 12;
l_trx_lines_tbl(2).unit_selling_price := 12;
l_trx_lines_tbl(2).line_type := 'LINE';
```

f. Populate freight information and link it to line 1.

```
l_trx_lines_tbl(3).trx_header_id := 101;
l_trx_lines_tbl(3).trx_line_id := 103;
l_trx_lines_tbl(3).link_to_trx_line_id := 101;
l_trx_lines_tbl(3).line_number := 1;
l_trx_lines_tbl(3).line_type := 'FREIGHT';
l_trx_lines_tbl(3). amount := 25;
```

g. Call the invoice api to create multiple invoices in a batch.

```
AR_INVOICE_API_PUB.create_single_invoice(
  p_api_version           => 1.0,
  p_batch_source_rec      => l_batch_source_rec,
  p_trx_header_tbl        => l_trx_header_tbl,
  p_trx_lines_tbl         => l_trx_lines_tbl,
  p_trx_dist_tbl          => l_trx_dist_tbl,
  p_trx_salescredits_tbl  => l_trx_salescredits_tbl,
  x_customer_trx_id       => l_customer_trx_id,
  x_return_status         => l_return_status,
  x_msg_count             => l_msg_count,
  x_msg_data              => l_msg_data);

IF l_return_status = fnd_api.g_ret_sts_error OR
   l_return_status = fnd_api.g_ret_sts_unexp_error THEN
  dbms_output.put_line('unexpected errors found!');
ELSE
```

h. Check whether any record exist in error table

```
SELECT count(*)
Into      cnt
From ar_trx_errors_gt;
```

```
IF cnt = 0
THEN
dbms_output.put_line ( 'Customer Trx id '|| l_customer_trx_id);
ELSE
dbms_output.put_line ( 'Transaction not Created, Please check ar_trx_errors_
gt table');
 END IF;

END;
/
```

i. See all the validation errors.

```
SET LINESIZE 200
COLUMN trx_header_id HEADING 'Header ID'
COLUMN trx_line_id   HEADING 'Line ID'
COLUMN error_message HEADING 'Message'
COLUMN invalid_value HEADING 'Invalid Value'
COLUMN trx_header_id FORMAT 9999999
COLUMN trx_line_id   FORMAT 9999999
COLUMN error_message FORMAT a30
COLUMN invalid_value FORMAT a20
SELECT trx_header_id, trx_line_id, error_message, invalid_value
FROM ar_trx_errors_gt;
```

Note: In the above examples, we did not pass distribution and sales credits. Note, however, that you *can* create an invoice passing distributions and sales credits.

6

Prepayments API User Notes

Overview

This document outlines the specifications and the methodology for using the Prepayments API.

Use the Prepayments API to:

- Generate a unique payment grouping identifier (payment_set_id)
- Create a prepayment receipt flagged with this payment_set_id
- Apply the prepayment receipt to a receivable activity of type Prepayment

You can access this API:

- As standard PL/SQL server-side routine calls
- Through forms, utilizing the capability of Forms6 to have a procedure as its underlying base table

Basic Business Needs

The Prepayments API addresses the following business needs:

- Enables the creation of a receipt in advance of the invoicing event
- Provides a mechanism of matching a prepayment receipt to a prepaid invoice

The Prepayments API lets you model down payments, deposits, or prepayments as receipts created in Oracle Receivables in advance of the invoice creation event.

It is not intended for the purpose of creating receipts for existing invoices, simply before the invoices.

Before you begin....

Initialization of ARP_STANDARD and ARP_GLOBAL

Custom code that uses AR or HZ APIs will set the ORG_ID via dbms_application_info.set_client_info() and then call the APIs. The APIs in turn might access either ARP_STANDARD and ARP_GLOBAL, which initialize the global variables that are used across Oracle Receivables when the package is first called. Most of these global variable values are organization dependent, and the first such call sets the global variables based on the current ORG_ID.

If additional custom code then changes the ORG_ID via another call to dbms_application_info.set_client_info(), then the ORG context changes, ***but the ARP_STANDARD and ARP_GLOBAL context does not.***

In such cases, you should explicitly re-initialize the global variables by a call to these two public procedures:

1. ARP_GLOBAL.INIT_GLOBAL: For setting public variables in ARP_GLOBAL.
2. ARP_STANDARD.INIT_STANDARD: For setting public variables in ARP_STANDARD.

Major Features

Flexibility

Per Oracle API coding standards, the various APIs in the Prepayment API package let you specify an ID or its associated value for any attribute that is an INPUT parameter of the API.

If both an ID and value have been specified, then the ID takes precedence over the value. This provides a wide degree of flexibility when using the API, both as a base table of the form and as a server-side routine call from the PL/SQL code.

The extensive defaulting mechanism for the input parameters ensures that you will be able to achieve the basic business needs of creating a prepayment receipt by calling the relevant APIs with a minimum number of parameters. This gives you many options to achieve your requirements when you call the relevant API.

Modular Approach

The API has been designed in a highly modular fashion, resulting in code that is:

- Easy to understand
- Easy to maintain
- Easy to expand

Error Handling

The Prepayment API provides an extensive error-handling and error-reporting mechanism whereby all errors encountered in the Defaulting and Validation phases are reported and put on the message stack. The calling program can look up all error messages, or the first error message on the stack.

If only one error exists on the message stack, then you do not need to fetch the message from the stack because the message will return as one of the output parameters of the API routine.

Robust Validation

The validations that the Prepayment API performs are robust in nature. The APIs collect all the validation errors encountered and put them on the message stack. The relevant entity handler is called only if no errors are reported during the Defaulting and Validation phases.

Debug Messages

Extensive debug messages have been incorporated to simplify the troubleshooting process when problems are encountered with the API.

Debug messages can be written to the log file by calling the appropriate routines described in *Exception Handling and Result Messages* on page 6-7.

Solution Outline

PL/SQL API

To create, apply, and refund a prepayment receipt, you can call the following routine:

- *AR_PREPAYMENTS_PUB.Create_Prepayment* on page 6-10: Use this routine to create a prepayment receipt.

Modular Approach

To modularize the Prepayments API, the basic structure of the API is divided into four parts:

1. Defaulting the IDs from the values and cross validating, if you provide both the values and the IDs.
2. Defaulting all the entity level information, which you have not entered or which the API needs internally.
3. Validating the entity level information that you entered.
4. Calling to the Receipt API to create a prepaid receipt.

This results in code that is easy to understand and easy to maintain. Any new functionality can be added by a simple code plug-in at each of the four parts.

Defaulting

In general, the various parameters in the API call, if not entered, get defaulted based on the following:

- Values of the other parameters in the API call
- Values set in the AR_SYSTEM_PARAMETERS table entered through the System Options form
- Relevant profile option values

Depending on the above three factors and the exact business requirement, the minimum number of parameters required to perform certain business tasks may vary.

Null values are defaulted for the parameters that could not be defaulted by the API defaulting routines.

For various attributes of the business objects, you can pass either the ID or the value of the attribute.

If you specify only the value, then the value is used to derive the ID; otherwise, the ID (if specified) is taken directly. If you specify both the ID and the value, then the ID takes precedence over the value and a warning message informs you of this.

Exception Handling and Result Messages

The Prepayments API returns three types of information to its calling programs:

- Overall status
- Messages describing the operations performed or errors encountered by the APIs
- Some output values that the API caller might want to use (this is different for different API routines and is described in *API Usage* on page 6-10)

Return Status

The return status (x_return_status) of the API informs the caller about the result of the operation (or operations) performed by the API. The different possible values for an API return status are:

- Success (FND_API.G_RET_STS_SUCCESS)
- Error (FND_API.G_RET_STS_ERROR)
- Unexpected error (FND_API.G_RET_STS_UNEXP_ERROR)

The following section describes the different values of return status and their meanings.

Success

A success return status means that the API was able to perform all the operations requested by its caller. A success return status may be accompanied by informative messages in the API message list.

Error

An error return status means that the API failed to perform some or all of the operations requested by its caller. An error return status is usually accompanied by messages describing the error (or errors) and how to fix it.

In most cases, you should be able to take corrective action to fix regular, expected errors such as missing attributes or invalid date ranges.

Unexpected error

An unexpected error status means that the API has encountered an error condition it did not expect or could not handle. In this case, the API is unable to continue with its regular processing. Examples of such errors are irrecoverable data inconsistency errors, memory errors, and programming errors (such as attempting a division by zero).

In most cases, only system administrators or application developers can fix these unexpected errors.

Messages

The APIs put result messages into a message list. Programs calling the APIs can then get the messages from the list and process them by issuing them, loading them into a database table, or writing them to a log file.

Messages are stored in an encoded format to let the API callers find message names using the standard functions provided by the message dictionary. It also allows the storing of these messages in database tables and reporting off these tables in different languages.

The API message list must be initialized every time a program calls an API. API callers can either call the message list utility function FND_MSG_PUB.Initialize or request that the API do the initialization on their behalf by setting the p_init_msg_ list parameter to TRUE.

The program calling the API can retrieve messages from the message stack using the existing FND API functions FND_MSG_PUB.Count_Msg and FND_MSG_ PUB.Get.

Message Level Threshold

The message level threshold is stored in a profile option named FND_API_MSG_ LEVEL_THRESHOLD. This profile option can be updated at all levels (site, application, or user). The API checks against this threshold before writing a message to the API message list.

Debugging

You must enable debugging by calling the routine arp_standard.enable_file_debug. The routine requires 2 parameters: path_name and file_name.

```
arp_standard.enable_file_debug(<pathname>, <filename>)
```

The path name can be identified by using the following select statement:

```
select value from v$parameter where name = 'utl_file_dir',
```

The file name can be any name that you choose.

Example

```
arp_standard.enable_file_debug ('/sqlcom/log','txt.log')
```

This call would write the output debug file 'txt.log' in the path '/sqlcom/log'.

Calling Program Context

The program calling these APIs should have set up the application, responsibility, and user in the context of Oracle Application.

If the calling program does not set up this context, then it can be done programmatically by calling the following FND API.

```
fnd_global.apps_initialize (      user_id in number,
                                  resp_id in number,
                                  resp_appl_id in number,
                                  security_group_id in number default 0);
```

API Usage

This section describes how to use the Prepayments API to:

- Create a prepayment receipt
- Apply the prepayment receipt to the prepayment activity
- Calculate the amount of all the installments of a particular payment term

AR_PREPAYMENTS_PUB.Create_Prepayment

Description

This routine is called to create a prepayment receipt.

This API routine has 5 output, 8 input-output, and 56 input parameters. Of the output parameters, the API returns 5.

Input

Standard API parameters: 4

Prepayment parameters: 48 + 8 (INOUT) parameters

4 (global descriptive flexfield parameters)

Output

Standard API parameters: 3

Prepayment parameters: 2 + 8 (INOUT) parameters

The input descriptive flexfield parameter is a record of type *attribute_rec_type*.

```
TYPE attribute_rec_type IS RECORD(
                        attribute_category    VARCHAR2(30) DEFAULT NULL,
                        attribute1            VARCHAR2(150) DEFAULT NULL,
                        attribute2            VARCHAR2(150) DEFAULT NULL,
                        attribute3            VARCHAR2(150) DEFAULT NULL,
                        attribute4            VARCHAR2(150) DEFAULT NULL,
                        attribute5            VARCHAR2(150) DEFAULT NULL,
                        attribute6            VARCHAR2(150) DEFAULT NULL,
                        attribute7            VARCHAR2(150) DEFAULT NULL,
                        attribute8            VARCHAR2(150) DEFAULT NULL,
                        attribute9            VARCHAR2(150) DEFAULT NULL,
                        attribute10           VARCHAR2(150) DEFAULT NULL,
```

```
                    attribute11             VARCHAR2(150) DEFAULT NULL,
                    attribute12             VARCHAR2(150) DEFAULT NULL,
                    attribute13             VARCHAR2(150) DEFAULT NULL,
                    attribute14             VARCHAR2(150) DEFAULT NULL,
                    attribute15             VARCHAR2(150) DEFAULT NULL);
```

The input global descriptive flexfield parameter is a record of type *global_attr_rec_type*.

```
TYPE global_attribute_rec_type IS RECORD(
                global_attribute_category     VARCHAR2(30) default null,
                global_attribute1             VARCHAR2(150) default NULL,
                global_attribute2             VARCHAR2(150) DEFAULT NULL,
                global_attribute3             VARCHAR2(150) DEFAULT NULL,
                global_attribute4             VARCHAR2(150) DEFAULT NULL,
                global_attribute5             VARCHAR2(150) DEFAULT NULL,
                global_attribute6             VARCHAR2(150) DEFAULT NULL,
                global_attribute7             VARCHAR2(150) DEFAULT NULL,
                global_attribute8             VARCHAR2(150) DEFAULT NULL,
                global_attribute9             VARCHAR2(150) DEFAULT NULL,
                global_attribute10            VARCHAR2(150) DEFAULT NULL,
                global_attribute11            VARCHAR2(150) DEFAULT NULL,
                global_attribute12            VARCHAR2(150) DEFAULT NULL,
                global_attribute13            VARCHAR2(150) DEFAULT NULL,
                global_attribute14            VARCHAR2(150) DEFAULT NULL,
                global_attribute15            VARCHAR2(150) DEFAULT NULL,
                global_attribute16            VARCHAR2(150) DEFAULT NULL,
                global_attribute17            VARCHAR2(150) DEFAULT NULL,
                global_attribute18            VARCHAR2(150) DEFAULT NULL,
                global_attribute19            VARCHAR2(150) DEFAULT NULL,
                global_attribute20            VARCHAR2(150) DEFAULT NULL);
```

The following table lists the parameters that pertain specifically to the Create Prepayment routine:

Parameter	Type	Mandatory/ Optional	Data-type	Default Value	Description
p_api_version	IN	M	Number		Constant 1.0
p_init_msg_list	IN	O	Varchar2		Default FND_API.G_FALSE
p_commit	IN	O	Varchar2		Default FND_API.G_FALSE
p_validation_level	IN	O	Number		Default FND_API.G_VALID_LEVEL_ FULL

Parameter	Type	Mandatory/ Optional	Data-type	Default Value	Description
x_return_status	OUT	M	Varchar2		Return status of the prepayment call
x_msg_count	OUT	M	Number		Message counts in message stack
x_msg_data	OUT	M	Varchar2		Message text in message stack.
p_usr_currency_code	IN	O	Varchar2		Translated currency code
p_currency_code	IN	M	Varchar2		Currency of the receipt
p_usr_exchange_rate_type	IN	O	Varchar2		User exchange rate type
p_exchange_rate_type	IN	O	Varchar2		Exchange rate type, if other than functional currency (if functional currency is different than receipt)
p_exchange_rate_date	IN	O	Date		Exchange rate date
p_exchange_rate	IN	O	Number		Exchange rate
p_amount	IN	M	Number		Receipt amount
p_factor_discount_amount	IN	O	Number		Factor discount amount
p_receipt_number	INOUT	O	Varchar2		Receipt number, need to pass if doc sequence is not enabled
p_receipt_date	IN	O	Date		Receipt creation Date
p_gl_date	IN	O	Date		GL date of the receipt
p_maturity_date	IN	O	Date		Maturity date of the receipt
p_postmark_date	IN	O	Date		Postmark date of receipt
p_customer_id	IN	M	Number		Customer ID of the receipt
p_customer_name	IN	O	Varchar2		Customer Name
p_customer_number	IN	O	Number		Customer Number
p_customer_bank_account_id	IN	M	Number		Customer bank account ID
p_customer_bank_account_num	IN	O	Varchar2		Customer bank account number
p_customer_bank_account_name	IN	O	Varchar2		Customer bank account name
p_location	IN	O	Varchar2		Location

Parameter	Type	Mandatory/ Optional	Data-type	Default Value	Description
p_customer_site_use_id	IN	M	Number		Site use ID
p_customer_receipt_ reference	IN	O	Varchar2		Reference information on receipt header
p_override_remit_ account_flag	IN	O	Varchar2		Remittance account override flag
p_remittance_bank_ account_id	IN	M	Varchar2		Remittance bank account ID
p_remittance_bank_ account_num	IN	O	Varchar2		Remittance bank account number
p_remittance_bank_ account_name	IN	O	Varchar2		Remittance bank account name
p_deposit_date	IN	O	Date		Deposit date
p_receipt_method_id	IN	M	Number		Remittance method ID (Payment Method)
p_receipt_method_ name	IN	O	Varchar2		Receipt method name
p_doc_sequence_value	IN	O	Number		Doc sequence value, if doc sequence is enabled (mandatory if doc sequence is enabled)
p_ussgl_transaction_ code	IN	O	Number		USSGL transaction code, if exists, on receipt header
p_anticipated_clearing_ date	IN	O	Date		Anticipated receipt clearing date
p_called_from	IN	M	Number		Which program called this routine?
p_attribute_rec	IN	O	Record type		Receipt Header attributes
p_global_attribute_rec	IN	O	Record type		Global attributes on receipt header (GDF)
p_receipt_comments	IN	O	Varchar2		Receipt header comments
p_issuer_name	IN	O	Varchar2		AR Notes Issuer name
p_issue_date	IN	O	Date		AR Notes Issue Date
p_issuer_bank_branch_ id	IN	O	Number		AR Notes Issuer bank branch ID

Parameter	Type	Mandatory/ Optional	Data-type	Default Value	Description
p_cr_id	OUT	M	Number		Cash receipt ID
p_applied_payment_ schedule_id	IN	M	Number		For prepayment, it will be -7
p_amount_applied	IN	O	Number		Specify amount which needs to be put in prepayment out of the receipt amount
p_application_ref_type	IN	O	Varchar2		Prepayment application reference from a lookup code for lookup type AR_ PREPAYMENT_TYPE to indicate where it is created from. For example, OM.
p_application_ref_id	IN OUT	M	Number		Application reference ID. For example, order ID.
p_application_ref_num	IN OUT	M	Varchar2		Reference number. For example, order number.
p_secondary_ application_ref_id	IN OUT	O	Number		Additional reference, if exists
p_receivable_trx_id	IN	O	Number		Receivable activity ID, default if not passed for prepayment.
p_amount_applied_ from	IN	O	Number		Amount applied in functional currency
p_apply_date	IN	O	Date		If null, takes sysdate
p_apply_gl_date	IN	O	Date		Application GL date
app_ussgl_transaction_ code	IN	O	Varchar2		USSGL transaction type code on application
p_show_closed_ invoices	IN	O	Varchar2		Default FALSE
p_move_deferred_tax	IN	O	Varchar2		Default Y
app_attribute_rec	IN	O	Record type		Application attributes
app_global_attribute_ rec	IN	O	Record Type		Global application attributes (GDF)
app_comments	IN	O	Varchar2		comments on application
p_payment_server_ order_num	IN OUT	M	Varchar2		Payment server order number

Parameter	Type	Mandatory/ Optional	Data-type	Default Value	Description
p_call_payment_ processor	IN	O	Varchar2		Decides whether to call *i*Payment. DEFAULT FND_API.G_FALSE
p_payment_response_ error_code	IN OUT	M	Varchar2		*i*Payment return error code
p_approval_code	IN OUT	M	Varchar2		Credit Card Approval code
p_receivable_ application_id	OUT	M	Number		Receivable applications ID of the application
p_payment_set_id	IN OUT	M	Number		If passed, it will take the passed payment_set_id while creating prepayment application. Otherwise, generate a new number and pass it back.

Example

The following is a test case for creating a prepayment.

Objective:

To create a prepayment, passing the minimum number of parameters.

Entered parameters:

- p_api_version
- p_currency_code
- p_amount
- p_customer_id
- p_customer_bank_account_id
- p_customer_site_use_id
- p_remittance_bank_account_id
- p_receipt_method_id
- p_called_from
- p_applied_payment_schedule_id
- p_application_ref_id

- p_application_ref_num

The API call in this case would be:

```
AR_PREPAYMENTS_PUB.create_prepayment(
      p_api_version       => 1.0,
      p_commit            => FND_API.G_FALSE,
      x_return_status     => x_return_status,
      x_msg_count         => x_msg_count,
      x_msg_data          => x_msg_data,
      p_init_msg_list     => FND_API.G_TRUE,
      p_receipt_number    => l_receipt_number,
      p_currency_code     => l_currency_code,
      p_amount            => p_payment_amount,
      p_receipt_method_id => l_receipt_method_id,
      p_customer_id       => p_customer_id,
      p_customer_site_use_id => l_site_use_id,
      p_customer_bank_account_id => p_bank_account_id,
      p_currency_code     => l_receipt_currency_code,
      p_exchange_rate     => l_receipt_exchange_rate,
      p_exchange_rate_type => l_receipt_exchange_rate_type,
      p_exchange_rate_date => l_receipt_exchange_rate_date,
      p_applied_payment_schedule_id => p_payment_schedule_id,
      p_application_ref_type      =>  l_application_ref_type , --Order type
      p_application_ref_num => l_application_ref_num, --Order Number
      p_application_ref_id => l_application_ref_id, --Order Id
      p_cr_id             => l_cr_id --OUT,
      p_receivable_application_id => l_receivable_application_id --OUT
      p_call_payment_processor => l_call_payment_processor
      p_payment_response_error_code=> l_payment_response_error_code
      p_payment_set_id  => l_payment_set_id -If not passed generate a new number
      );
```

AR_PREPAYMENTS_PUB.Get_Installment

Description

This routine is called to calculate the amount of all the installments of a given payment term.

This API routine has 4 output and 3 input parameters. Of the output parameters, the API returns 5.

Input

Standard API parameters: 0

Prepayment parameters: 3

Output

Standard API parameters: 3

Prepayment parameters: 1

The following table lists the parameters that pertain specifically to the Get Installment routine:

Parameter	Type	Mandatory /Optional	Data- Type	Default Value	Details
p_term_id	IN	M	Number		Payment term ID
p_amount	IN	M	Varchar2		Input amount for which the installment amount needs to be calculated
p_currency_code	IN	M	Varchar2		Currency code for calculating the installment amount
p_installment_tbl	OUT	O	Number		A table consisting of installment number and installment amount
x_return_status	OUT	M	Varchar2		Return status of the API call
x_msg_count	OUT	M	Number		Message counts in message stack
x_msg_data	OUT	M	Varchar2		Message text in message stack.

Example

The following is a test case for get_installment.

Objective:

To get the installment amount given an amount, payment term and currency code.

Entered parameters:

- p_term_id
- p_amount
- p_currency_code

```
AR_PREPAYMENTS_PUB.get_installment(
     p_term_id     => l_term_id     ,
     p_amount => l_amount,
     p_currency_code => l_currency_code,
     p_installment_tbl=>  l_installment_tbl ,  --OUT
     x_return_status     => x_return_status,
     x_msg_count         => x_msg_count,
     x_msg_data          => x_msg_data);
```

Messages

Messages play an important role in the effectiveness of your API calls. The right message is raised at the right point to convey to you the exact error that has occurred or any warnings that have been raised.

In the Prepayments API, all error messages and warnings raised during the execution are put on the message stack and can be retrieved by the user as described in *Robust Validation* on page 6-5.

The following is the list of all error messages raised by the Prepayments API.

Message Number	Message Name	Message Description
96735	AR_RAPI_CUS_BK_AC_2_ INVALID	Invalid combination of customer bank account name and number.
294347	AR_RAPI_PREPAY_SEQ_ FAILED	The prepayment sequence generation has failed. Please contact your system administrator.
	AR_PPAY_PAY_TERM_ INVALID	Payment term ID is invalid.
	AR_PPAY_BASE_AMOUNT_ INVALID	The amount can not be null, 0, or negative.
96734	AR_RAPI_CURR_CODE_ INVALID	Currency code is invalid.

Since this API also calls the Receipt API AR_RECEIPT_API_PUB, it could also throw messages raised by the Receipt API.

Please refer to messages listed in *Chapter 7, Receipt API User Notes* on page 7-1.

7

Receipt API User Notes

Overview

This document outlines the specifications and the methodology for using the various Receipt APIs. These APIs provide an extension to existing functionality of creating and manipulating receipts through standard AR Receipts forms and lockboxes.

You can access these APIs:

- As standard PL/SQL server-side routine calls
- Through forms, utilizing the capability of Forms6 to have a procedure as its underlying base table

Basic Business Needs

The Receipt API caters to the following basic functionality via different API calls:

- Creating a cash receipt
- Applying a cash receipt to a debit item
- Creating a cash receipt and applying it to a debit item in one pass
- On-account application
- Unapplying the on-account application
- Unapplying the receipt application to a particular transaction
- Reversing the receipt
- Activity application, such as Receipt Write-off
- Creating a miscellaneous receipt
- Other account application, such as Claim Investigation
- Receipt-to-receipt application
- Creating a cash receipt and an on-account application in one pass

Before you begin....

Initialization of ARP_STANDARD and ARP_GLOBAL

Custom code that uses AR or HZ APIs will set the ORG_ID via dbms_application_info.set_client_info() and then call the APIs. The APIs in turn might access either ARP_STANDARD and ARP_GLOBAL, which initialize the global variables that are used across Oracle Receivables when the package is first called. Most of these global variable values are organization dependent, and the first such call sets the global variables based on the current ORG_ID.

If additional custom code then changes the ORG_ID via another call to dbms_application_info.set_client_info(), then the ORG context changes, ***but the ARP_STANDARD and ARP_GLOBAL context does not.***

In such cases, you should explicitly re-initialize the global variables by a call to these two public procedures:

1. ARP_GLOBAL.INIT_GLOBAL: For setting public variables in ARP_GLOBAL.
2. ARP_STANDARD.INIT_STANDARD: For setting public variables in ARP_STANDARD.

Major Features

Flexibility

Per Oracle API coding standards, the various APIs in the Receipt API package provide the option of specifying an ID or its associated value for any attribute which is an input parameter of the API. If both the ID and value are specified, then the ID takes precedence over the value. This provides a wide degree of flexibility for using the API, both as a base table of the form and as a server-side routine call from the PL/SQL code.

The extensive defaulting mechanism for the input parameters ensures that you can create, apply, unapply, and reverse a receipt by calling the relevant APIs with a minimum number of parameters. This gives you many options when you call the relevant API to achieve your requirements.

Modular Approach

The Receipt API has been designed in a highly modular fashion, giving you code that is:

- Easy to understand
- Easy to maintain
- Easy to extend

Error Handling

The Receipt API provides an extensive error-handling and error-reporting mechanism whereby all errors encountered in the Defaulting and Validation phases are reported and put on the message stack. The calling program has the option of looking at all the error messages or only the first error message on the stack. If only one error is in the message stack, then the message comes out as one of the output parameters of the API routine; you do not need to fetch the message from the stack.

Robust Validation

The validations performed by the Receipt API are robust in nature. The APIs collect all the validation errors encountered and put them on the message stack. The relevant entity handler is called only if there are no errors reported during the Defaulting and Validation phases.

Debug Messages

Extensive debug messages have been incorporated. In the case of unexpected problems, these will be used to troubleshoot. This is extremely useful because APIs are otherwise difficult to debug.

Debug messages can be written to the log file by calling the appropriate routines described in *Exception Handling and Result Messages* on page 7-8.

Solution Outline

PL/SQL APIs

To achieve the basic functionality of creating a cash receipt, applying it, unapplying it, or reversing it, you can call the following APIs:

- *Ar_receipt_api_pub.Create_cash*: Creates a single cash receipt, as in the case of manually created cash receipts.
- *Ar_receipt_api_pub.Apply*: Applies a cash receipt to a particular installment of a debit item. The application can also be a cross currency application.
- *Ar_receipt_api_pub.Create_and_apply*: Creates a cash receipt and applies it to a specified installment of a debit item in one pass. Application fails if the creation fails due to some reason.
- *Ar_receipt_api_pub.Apply_on_account*: Creates an on-account application for a cash receipt.
- *Ar_receipt_api_pub.Unapply_on_account*: Unapplies the on-account application on the specified receipt.
- *Ar_receipt_api_pub.Unapply*: Unapplies the application of a particular installment of a debit item against the specified cash receipt.
- *Ar_receipt_api_pub.Reverse*: Reverses the specified receipt.
- *Ar_receipt_api_pub.activity_application*: Applies to an activity, such as Receipt Write-off.
- *Ar_receipt_api_pub.activity_unapplication*: Unapplies from an activity, such as a Receipt Write-off.
- *Ar_receipt_api_pub.Create_misc*: Creates a single miscellaneous receipt.
- *Ar_receipt_api_pub.apply_other_account*: Applies to other account activities, such as Claim Investigation (for Trade Management customers only).
- *Ar_receipt_api_pub.unapply_other_account*: Unapplies from other account activities, such as Claim Investigation.
- *Ar_receipt_api_pub.apply_open_receipt*: Creates a receipt-to-receipt application (payment netting).
- *Ar_receipt_api_pub.unapply_open_receipt*: Unapplies a receipt-to-receipt application.

- *Ar_receipt_api_pub.Create_apply_on_acc*: Creates a cash receipt and an on-account application in one pass. If the receipt creation fails, then the application fails as well.

Modular Approach

To modularize the Receipt API, the basic structure of the API is divided into four parts. The API:

1. Defaults the IDs from the values and cross validates if both the values and the IDs are entered by the user.
2. Defaults all the entity level information which the user has not entered or which the API needs internally.
3. Validates the entity level information entered by the user.
4. Calls the entity handlers to perform the relevant task, such as Create, Apply, Unapply, or Reverse.

This results in easy to understand and easy to maintain code. Any new functionality can be added by a simple code plug-in at each of the four parts.

Defaulting

In general, if the various parameters in the API call are not entered, then they get defaulted based on the following:

- Values of the other parameters in the API call
- Values set in the AR_SYSTEM_PARAMETERS table entered through the System Options form
- Relevant profile option values

Depending on the above three factors and the exact business requirement, the minimum number of parameters that may be required to perform certain business tasks might vary.

Null values are defaulted for the parameters which could not be defaulted by the API defaulting routines.

For various attributes of the business objects, you can pass either the ID or the value of the attribute. If you specify only the value, then the value is used to derive the ID; otherwise, the ID (if specified) is taken directly. If you specify both the ID and the value, then the ID takes precedence over the value and a warning message informs you of this.

Exception Handling and Result Messages

The Receipt API gives back three types of information to its calling programs:

- Overall status
- Messages describing the operations performed or errors encountered by the APIs
- Some output values that the API caller might want to use (this is different for different API routines and is described in *API Usage* on page 7-11)

Return Status

The return status (x_return_status) of the API informs the caller about the result of the operation (or operations) performed by the API. The different possible values for an API return status are:

- Success (FND_API.G_RET_STS_SUCCESS)
- Error (FND_API.G_RET_STS_ERROR)
- Unexpected error (FND_API.G_RET_STS_UNEXP_ERROR)

The following section describes the different values of return status and their meanings.

Success

A success return status means that the API was able to perform all the operations requested by its caller. A success return status may be accompanied by messages in the API message list which will be informative.

Error

An error return status means that the API failed to perform some or all of the operations requested by its caller. An error return status is usually accompanied by messages describing the error (or errors) and how to fix it.

In most cases, you should be able to take corrective action to fix regular, expected errors such as missing attributes or invalid date ranges.

Unexpected error

An unexpected error status means that the API has encountered an error condition it did not expect or could not handle. In this case, the API is unable to continue with its regular processing. Examples of such errors are irrecoverable data inconsistency errors, memory errors, and programming errors (such as attempting a division by zero).

In most cases, only system administrators or application developers can fix these unexpected errors.

Messages

The APIs put result messages into a message list. Programs calling these APIs can then get the messages from the list and process them by issuing them, loading them into a database table, or writing them to a log file.

Messages are stored in an encoded format to let the API callers find message names using the standard functions provided by the message dictionary. It also allows the storing of these messages in database tables and reporting off these tables in different languages. See *Messages* on page 7-131 for more information.

The API message list must be initialized every time a program calls an API. API callers can either call the message list utility function FND_MSG_PUB.Initialize or request the API to do the initialization on their behalf by setting the p_init_msg_list parameter to TRUE.

The program calling the API can retrieve messages from the message stack using the existing FND API functions FND_MSG_PUB.Count_Msg and FND_MSG_PUB.Get.

Message Level Threshold

The message level threshold is stored in a profile option named FND_API_MSG_LEVEL_THRESHOLD. This profile option can be updated at all levels (site, application, responsibility, or user). The API checks against this threshold before writing a message to the API message list.

Debug Messages

The calling program enables debugging by calling the routine *arp_standard.enable_file_debug (<pathname>, <filename>)*.

This routine takes in two parameters, path_name and file_name. To find the path name, use the following select statement:

```
select value from v$parameter where name = 'utl_file_dir',
```

The file name can be any name that you choose.

Example:

```
arp_standard.enable_file_debug ('/sqlcom/log', 'txt.log');
```

This call would write the output debug file 'txt.log' in the path '/sqlcom/log'.

Calling Program Context

The programs calling these APIs should have set up the application, responsibility, and the user in the context of Oracle Applications. If the calling program does not set up this context, then it can be done programmatically by calling the following FND API:

```
fnd_global.apps_initialize(        user_id in number,
                                   resp_id in number,
                                   resp_appl_id in number,
                                   security_group_id in number default 0);
```

Integration with Oracle *i*Payment

The following table illustrates the integration between Oracle *i*Payment and the Receipt API routines that create receipts:

Receipt API Routine	Calls Oracle *i*Payment?
Ar_receipt_api_pub.Create_cash	No
Ar_receipt_api_pub.Create_and_apply	Yes
Ar_receipt_api_pub.Create_misc	No
Ar_receipt_api_pub.Create_apply_on_acc	Yes

API Usage

Ar_receipt_api_pub.Create_cash

Description

This routine is called to create cash receipts for the payment received in the form of a check or cash. Cash receipts can be created as identified (with a customer) or as unidentified (without a customer).

> **Note:** This routine does ***not*** call Oracle iPayment directly. See *Integration with Oracle iPayment* on page 7-10.

This API routine has 4 output and 44 input parameters in total. As one of the output parameters, the API returns the cash_receipt_id of the cash receipt created. The following is the breakdown of the parameters:

Input

Standard API parameters: 4

Cash Receipt parameters: 38 + 1 (descriptive flexfield parameter)
+ 1 (global descriptive flexfield parameter)

Output

Standard API parameters: 3

Cash Receipt parameters: 1

The input descriptive flexfield parameter is a record of type *attribute_rec_type*.

```
TYPE attribute_rec_type IS RECORD
    (p_attribute_category    IN VARCHAR2,
    p_attribute1            IN VARCHAR2,
    p_attribute2            IN VARCHAR2,
    p_attribute3            IN VARCHAR2,
    p_attribute4            IN VARCHAR2,
    p_attribute5            IN VARCHAR2,
    p_attribute6            IN VARCHAR2,
    p_attribute7            IN VARCHAR2,
    p_attribute8            IN VARCHAR2,
    p_attribute9            IN VARCHAR2,
```

```
    p_attribute10           IN VARCHAR2,
    p_attribute11           IN VARCHAR2,
    p_attribute12           IN VARCHAR2,
    p_attribute13           IN VARCHAR2,
    p_attribute14           IN VARCHAR2,
    p_attribute15           IN VARCHAR2);
```

The input global descriptive flexfield parameter is a record of type *global_ attribute_ rec_type*.

```
TYPE global_attribute_rec_type IS RECORD
    (p_global_attribute_category     IN VARCHAR2,
    p_global_attribute1             IN VARCHAR2,
    p_global_attribute2             IN VARCHAR2,
    p_global_attribute3             IN VARCHAR2,
    p_global_attribute4             IN VARCHAR2,
    p_global_attribute5             IN VARCHAR2,
    p_global_attribute6             IN VARCHAR2,
    p_global_attribute7             IN VARCHAR2,
    p_global_attribute8             IN VARCHAR2,
    p_global_attribute9             IN VARCHAR2,
    p_global_attribute10           IN VARCHAR2,
    p_global_attribute11           IN VARCHAR2,
    p_global_attribute12           IN VARCHAR2,
    p_global_attribute13           IN VARCHAR2,
    p_global_attribute14           IN VARCHAR2,
    p_global_attribute15           IN VARCHAR2,
    p_global_attribute16           IN VARCHAR2,
    p_global_attribute17           IN VARCHAR2,
    p_global_attribute18           IN VARCHAR2,
    p_global_attribute19           IN VARCHAR2,
    p_global_attribute20           IN VARCHAR2);
```

The following table lists standard API parameters that are common to all the routines in the Receipt API.

Parameter	Type	Data-type	Required	Default Value	Description
p_api_version	IN	NUMBER	Yes		Used to compare version numbers of incoming calls to its current version number. Unexpected error is raised if version incompatibility exists. In the current version of the API, you should pass in a value of 1.0 for this parameter.

Parameter	Type	Data-type	Required	Default Value	Description
p_init_msg_list	IN	VARCHAR2		FND_API.G_FALSE	Allows API callers to request that the API does initialization of the message list on their behalf.
p_commit	IN	VARCHAR2		FND_API.G_FALSE	Used by API callers to ask the API to commit on their behalf.
p_validation_level	IN	NUMBER		FND_API.G_VALID_LEVEL_FULL	Not to be used currently as this is a public API.
x_return_status	OUT	VARCHAR2			Represents the API overall return status. Detailed in *Return Status* on page 7-8.
x_msg_count	OUT	NUMBER			Number of messages in the API message list
x_msg_data	OUT	VARCHAR2			This is the message in encoded format if x_msg_count=1

The following table lists the parameters that pertain specifically to the cash receipt routine.

Note: If required parameters are not passed in a call to this API, then the call will fail. However, depending on the business scenario, you will have to pass in values for other parameters to successfully create the business object. Otherwise, error messages will be reported.

Parameter	Type	Data-type	Required	Description
p_usr_currency_code	IN	VARCHAR2		The translated currency code. Used to derive the p_currency_code if it is not entered. Default: None Validate: Should be a valid currency, so that the corresponding currency code can be derived. Error: AR_RAPI_USR_CURR_CODE_INVALID

Parameter	Type	Data-type	Required	Description
p_currency_code	IN	VARCHAR2		The actual currency code that gets stored in AR tables. Default: 1. Derived from p_usr_currency_code if entered, else 2. Defaults to the functional currency code Validate: 1. Validated against the currencies in fnd_currencies table. Error: AR_RAPI_CURR_CODE_INVALID Warning: AR_RAPI_FUNC_CURR_ DEFAULTED
p_usr_exchange_rate_ type	IN	VARCHAR2		The translated exchange rate type. Used to derive the p_exchange_rate_type if it has not been entered. Default: None Validate: Should be a valid rate type. Error: AR_RAPI_USR_X_RATE_TYP_ INVALID
p_exchange_rate_type	IN	VARCHAR2		Exchange rate type stored in AR tables. Default: 1. In case of foreign currency receipt, derived from p_usr_exchange_rate_type. 2. In case of foreign currency receipt, defaults from profile option 'AR: Default Exchange Rate Type' Validate: 1. Validated against values in gl_ daily_conversion_types table. Error: AR_RAPI_X_RATE_TYPE_INVALID

Parameter	Type	Data-type	Required	Description
p_exchange_rate	IN	NUMBER		The exchange rate between the receipt currency and the functional currency. Default: 1. Derived from the Daily Rates table for rate_type <>'User' in case of non-functional currency 2. If profile option Journals: Display Inverse Rate = 'Y', set user entered value to 1/ p_exchange_rate 3. The entered value is rounded to a precision of 38. Validate: 1. In case of non-functional currency the rate should have a positive value for rate type= 'User' 2. For non-functional currency and type is <> 'User', do not specify any value. Error: AR_RAPI_X_RATE_INVALID AR_RAPI_X_RATE_NULL
p_exchange_rate_date	IN	DATE		The date on which the exchange rate is valid. Default: Receipt date Validate: 1. For a non-functional currency and type is <>'User' there should be a valid rate existing in the database for this date. This is a cross validation of type, currency, and date. Error: AR_NO_RATE_DATA_FOUND
p_amount	IN	NUMBER	Yes	The cash receipt amount. Default: Null Validate: >0 Error: AR_RAPI_REC_AMT_NEGATIVE AR_RAPI_RCPT_AMOUNT_NULL

Parameter	Type	Data-type	Required	Description
p_factor_discount_amount	IN	NUMBER		The bank charges on the cash receipt. Default: None Validate: 1. Bank charges not allowed if profile option AR: Create Bank Charges = 'No'. 2. Bank charges not allowed if the receipt state, derived from the receipt class of the receipt method <> 'CLEARED'. 3. If allowed then >=0 Error: AR_JG_BC_AMOUNT_NEGATIVE AR_BK_CH_NOT_ALLWD_IF_NOT_CLR
p_receipt_number	IN	VARCHAR2(30)		The receipt number of the receipt to be created. Default: If not specified, the receipt number is defaulted from the document sequence value. Validate: Receipt number should not be null. Error: AR_RAPI_RCPT_NUM_NULL
p_receipt_date	IN	DATE		The receipt date of the entered cash receipt. Default: System date Validate: None Error: None

Parameter	Type	Data-type	Required	Description
p_gl_date	IN	DATE		Date that this receipt will be posted to the General Ledger. Default: Gets defaulted to the receipt date if it is a valid gl_date. Validate: The date is valid if the following conditions are true: ■ The date is in an Open or Future period ■ The period cannot be an Adjustment period If the date is invalid, then: ■ If the most recent open period is prior to the receipt date: last date of that period ■ If there is a period open after the receipt date: first date of the last open period Error: AR_INVALID_APP_GL_DATE
p_maturity_date	IN	DATE		Receipt maturity date. Default: Deposit date Validate: >= p_receipt_date Error: AR_RW_MAT_BEFORE_RCT_DATE
p_postmark_date	IN	DATE		The postmark date Default: None Validate: None Error: None
p_customer_id	IN	NUMBER(15)		The customer_id for the paying customer. Default: Defaulted from customer name/number Validate: 1. Customer exists and has prospect code = 'CUSTOMER' 2. Customer has a profile defined at the customer level Error: AR_RAPI_CUST_ID_INVALID

Parameter	Type	Data-type	Required	Description
p_customer_name	IN	VARCHAR2(50)		The name for the entered customer. Used to default the customer id if not specified. Default: None Validate: None Error: AR_RAPI_CUS_NAME_INVALID
p_customer_number	IN			The customer number. Used to default the customer_id if not specified Default: None Validate: None Error: AR_RAPI_CUS_NUM_INVALID
p_customer_bank_account_id	IN	NUMBER(15)		The customer bank account id. Default: From bank account id/number Validate: 1. It must be a valid Bank Account of the paying customer 2. The inactive date (if defined) of the Bank Account, should be greater than the receipt_date 3. The receipt date has to be within the Start date and the End date of the Bank Account Error: AR_RAPI_CUS_BK_AC_2_INVALID AR_RAPI_CUS_BK_AC_ID_INVALID
p_customer_bank_account_num	IN	VARCHAR2(30)		The customer bank account number. Used to default the customer bank account id, if not specified Default: None Validate: None Error: None
p_customer_bank_account_name	IN	VARCHAR2(80)		The customer bank account name. Used to default the customer bank account id, if not specified Default: None Validate: None Error: None

Parameter	Type	Data-type	Required	Description
p_location	IN	VARCHAR2(40)		The Bill_To location for the customer. Used to derive the p_customer_site_use_id Default: None Validate: None Error: AR_RAPI_CUS_LOC_INVALID
p_customer_site_use_id	IN	NUMBER(15)		The Bill_To site_use_id for the customer Default: 1. Defaulted from customer location, else 2. Primary Bill_To customer site_use_id of the customer. Validate: It should be a valid Bill_To site of the paying customer. Error: AR_RAPI_CUS_SITE_USE_ID_INVALID
p_customer_receipt_reference	IN	VARCHAR2(30)		This column is used to store a customer receipt reference value supplied by the customer at the confirmation time. Default: None Validate: None Error: None
p_override_remit_bank_account_flag	IN	VARCHAR2(1)		The flag value decides when the remittance bank account can be overridden by the remittance selection process. Default: 'Y' Validate: valid values 'Y' and 'N' Error: AR_RAPI_INVALID_OR_REMIT_BK_AC

Parameter	Type	Data-type	Required	Description
p_remittance_bank_account_id	IN	NUMBER(15)		Identifies the user's bank account for depositing the receipt. Default: 1. From remittance bank account number 2. From the receipt method based on logic mentioned in *Defaulting* on page 7-24. Validate: Validation logic detailed in *Validation* on page 7-23. Error: AR_RAPI_REM_BK_AC_ID_INVALID AR_RAPI_REM_BK_AC_ID_NULL
p_remittance_bank_account_num	IN	VARCHAR2(30)		The remittance bank account number. Used to default the remittance bank account id if not specified. Default: None Validate: None Error: AR_RAPI_REM_BK_AC_NUM_INVALID
p_remittance_bank_account_name	IN	VARCHAR2(50)		The remittance bank account name. Used to default the remittance bank account id if not specified Default: None Validate: None Error: AR_RAPI_REM_BK_AC_NAME_INVALID
p_deposit_date	IN	DATE		The deposit date. Default: receipt date Validate: None Error: None
p_receipt_method_id	IN	NUMBER(15)		Identifies the payment method of the receipt Default: From receipt method name Validate: Validation detailed in *Validation* on page 7-23 Error: AR_RAPI_INVALID_RCT_MD_ID

Parameter	Type	Data-type	Required	Description
p_receipt_method_name	IN	VARCHAR2(30)		The payment method name of the receipt. Used to default the receipt method id if not specified. Default: None Validate: None Error: AR_RAPI_RCPT_MD_NAME_INVALID
p_doc_sequence_value	IN	NUMBER		Value assigned to document receipt. Default: Detailed in *Defaulting* on page 7-24 Validate: ■ User should not pass in the value if the current document sequence is automatic. ■ Document sequence value should not be entered if profile option Sequential Numbering is set to Not Used Error: AR_RAPI_DOC_SEQ_AUTOMATIC AR_RAPI_DOC_SEQ_VAL_INVALID
p_ussgl_transaction_code	IN	VARCHAR2(30)		Code defined by public sector accounting. Default: None Validate: None Error: None
p_anticipated_clearing_date	IN	DATE		Date the receipt is expected to be cleared. Default: None Validate: >= gl_date Error: AR_RW_EFFECTIVE_BEFORE_GL_DATE
p_event	IN	VARCHAR2		The event that resulted in the creation of the receipt. Currently used only by Bills Receivable. Default: None Validate: None Error: None

Parameter	Type	Data-type	Required	Description
p_called_from	IN	VARCHAR2(20)		This parameter is used to identify the calling routine. Currently used to identify only the 'BR_REMIT' program. Default: None Validate: None Error: None
p_attribute_record	IN	attribute_rec_ type (PL/SQL defined record type)		This is a record type which contains all the 15 descriptive flexfield segments and one descriptive flexfield structure defining column. It represents the Receipt Information flexfield. Default: DFF APIs used to do the defaulting and validation Validate: DFF APIs used to do the defaulting and validation Error: AR_RAPI_DESC_FLEX_INVALID
p_global_attribute_ record	IN	global_ attribute_rec_ type		This is a record type which contains all the 20 global descriptive flexfield segments and one global descriptive flexfield structure defining column. Default: None Validate: None Error:
p_comments	IN	VARCHAR2 (240)		User's comments
p_issuer_name	IN	VARCHAR2(50)		Issuer name of Notes Receivable (Asia Pacific Requirement) Default: None Validate: None Error:
p_issue_date	IN	DATE		Date Notes receivable was issued (Asia Pacific Requirement) Default: None Validate: None Error: None

Parameter	Type	Data-type	Required	Description
p_issuer_bank_ branch_id	IN	NUMBER(15)		Bank/ Branch issuing the Notes Receivable (Asia Pacific Requirement) Default: None Validate: None Error: None
p_cr_id	OUT	NUMBER(15)	Yes	The cash receipt id of the receipt created by the API call.

Validation

This section explains the validation mechanisms for the various parameters of this API which are relatively more complex and could not be explained in the Description column of the preceding table.

Validating Receipt Method ID

The receipt method ID is validated per the following conditions:

- It must be a valid receipt method ID in the AR_RECEIPT_METHOD table.
- Receipt date must lie between the receipt method start date and end date (if not null).
- The creation method code for the receipt class of this particular receipt method ID should be 'AUTOMATIC,' the remit flag ='Y,' and the confirm flag = 'N' or 'MANUAL.'
- At least one remittance bank account associated with this receipt method ID must have either the multi-currency flag set to 'Y' or the same currency as the receipt currency. In addition, this should have a bank account type = 'INTERNAL' and its inactive date (if specified) greater than the receipt_date.

Validating Remittance Bank Account ID

A remittance bank account ID, which is associated with a particular receipt method, is validated after validating the receipt method ID. If the receipt method ID is invalid, then the validation for the remittance bank account ID is not completed. An error message raised for an invalid value is AR_RAPI_INVALID_REMIT_BK_AC_ ID.

The remittance bank account ID must:

- Be a valid remittance bank account ID for the current receipt method.

- Have the multi-currency flag set to 'Y' or the same currency as the receipt currency. In addition, this should have a bank account type = 'INTERNAL' and its inactive date (if specified) greater than the receipt_date.

Validating for Duplicate Receipt

If the combination of the receipt_date, receipt_number, and amount on this receipt matches any existing receipts which have not been reversed, then the error message AR_RW_CASH_DUPLICATE_RECEIPT is raised.

Defaulting

This section explains the defaulting mechanisms for the various parameters of this API which are relatively more complex and could not be explained in the Description column of the preceding table.

Defaulting the Remittance Bank Account ID

In addition to being defaulted from the remittance bank account name and/or remittance bank account number, the remittance bank account identifier is defaulted from the receipt method that is specified for the cash receipt. If only one remittance bank account is associated with the specified receipt method that has the multi-currency flag = 'Y' or has same currency as the receipt currency, and the receipt date is within its start date and end date range, then that remittance bank account is used as the default value.

Example

Objective:

To create an identified cash receipt using a call to *Ar_receipt_api_pub.Create_cash* and passing a minimum number of input parameters.

This table lists the entered parameters:

Parameter	Entered Value	Default Value
p_api_version	1.0	
p_init_msg_list	FND_API.G_TRUE	
p_receipt_number	'aj_test_api_1'	
p_amount	1000	

Parameter	Entered Value	Default Value
p_receipt_method_id	1001	
p_customer_name	'Computer Service and Rentals'	

This table lists the defaulted input parameters, which were not entered:

Parameter	Entered Value	Default Value
p_customer_id		1006
p_currency_code		USD
p_receipt_date		10-FEB-2000
p_gl_date		10-FEB-2000
p_deposit_date		10-FEB-2000
p_customer_site_use_id		1025
p_override_remit_bank_ account_flag		'Y'
p_remittance_bank_ account_id		10001
p_maturity_date		10-FEB-2000

The API call in this case would be:

```
Ar_receipt_api_pub.Create_cash(
    p_api_version => 1.0,
    p_init_msg_list => FND_API.G_TRUE,
    p_receipt_number => 'aj_test_api_1',
    p_amount => 1000,
    p_receipt_method_id => 1001,
    p_customer_name => 'Computer Service and Rentals',
    p_cr_id => l_cr_id,
    x_return_status => l_return_status,
    x_msg_count => l_msg_count,
    x_msg_data => l_msg_data);
```

The warnings and the error messages that the API puts on the message stack are retrieved after execution of this API by the calling program in the following manner:

```
IF l_msg_count  = 1 Then
  --there is one message raised by the API, so it has been sent out
  --in the parameter x_msg_data, get it.
 l_msg_data_out := l_msg_data;
ELSIF l_msg_count  > 1 Then
--the messages on the stack are more than one so call them in a loop
-- and put the messages in a PL/SQL table.
   loop
    count := count +1 ;
    l_mesg := FND_MSG_PUB.Get;
     If l_mesg  IS NULL Then
      EXIT;
    else
      Mesg_tbl(count).message := l_mesg;
    End if;
 end loop;
END IF;
```

Depending on the message level threshold set by the profile option FND_API_MSG_LEVEL_THRESHOLD, the messages put on the message stack may contain both the error messages and the warnings.

Result:

We were able to create an identified cash receipt by specifying only six input parameters in our call to this API.

Similarly, without initializing the message stack (p_init_msg_list not passed and defaulted), you can create an unidentified cash receipt (without a customer) by passing only four input parameters to this API call.

Ar_receipt_api_pub.Apply

Description

Call this routine to apply the cash receipts of a customer (identified cash receipt) to a debit item. This debit item could be of the same customer or related customer, or an unrelated customer, depending on the value of the Allow Payment of Unrelated Transactions system option. This API routine has 3 output and 34 input parameters in total. Based on the type, the following is the breakdown of the parameters:

Input

Standard API parameters:	4	
Application parameters:	28	+ 1 (descriptive flexfield record parameters)
		+ 1 (global descriptive flexfield record parameters)

Output

Standard API parameters:	3
Application parameters:	0

The description of the standard API parameters is the same as that already given in *Ar_receipt_api_pub.Create_cash* on page 7-11.

The following table lists the parameters that pertain specifically to the Apply routine.

> **Note:** If required parameters are not passed in a call to this API, then the call will fail. However, depending on the business scenario, you will have to pass in values for other parameters to successfully create the business object. Otherwise, error messages will be reported.

Parameter	Type	Data-type	Required	Description
p_cr_id	IN	NUMBER(15)		The cash_receipt_id of the receipt which needs to be applied to a given debit item. Default: None Validate: 1. Type must be 'CASH' 2. Status must not be Reversed or Approved 3. The receipt must not be Unidentified Error: AR_RAPI_CASH_RCPT_ID_INVALID AR_RAPI_CASH_RCPT_ID_NULL
p_receipt_number	IN	VARCHAR2(30)		The receipt number of the receipt to be applied. Used to default the cash_receipt_id. Default: None Validate: None Error: AR_RAPI_RCPT_NUM_INVALID
p_customer_trx_id	IN	NUMBER(15)		The customer_trx_id of the debit item to which the receipt is to be applied Default: None Validate: Detailed in *Validation* on page 7-34 Error: Detailed in *Validation* on page 7-34
p_trx_number	IN	VARCHAR2(20)		The trx_number of the debit item to which the receipt is to be applied. Used to default the customer_trx_id Default: None Validate: None Error: AR_RAPI_TRX_NUM_INVALID
p_installment	IN	NUMBER(15)		The installment (or term_sequence_number) of the debit item. Used in conjunction with customer_trx_id to derive the applied payment schedule id if not specified. Default: 1, if only one installment exists for the debit item Validate: 1) >0; 2) valid installment of transaction. Also see *Validation* on page 7-34 Error: AR_RAPI_INSTALL_NULL

Parameter	Type	Data-type	Required	Description
p_applied_ payment_schedule_ id	IN	NUMBER(15)		The payment schedule id of the debit item. Also used to derive the customer_trx_id if not specified Default: Defaulted based on the installment and the customer_trx_id Validation:1. > 0; 2. It must correspond to Customer trx id and installment specified. 3. It must have the status <> 'CL' if the show closed invoices flag <> 'Y' Error: AR_RAPI_APP_PS_ID_INVALID
p_amount_applied	IN	NUMBER		The transaction amount to which the receipt is to be applied. This in the transaction currency. Default: Depending on the profile option AR: Cash-Default Amount Applied, it is defaulted either to: ■ the open amount of the transaction, or ■ the unapplied amount of the receipt. Discounts, if applicable, are taken into account by the discounts routine which calculates the amount applied. Validate: Detailed in *Validation* on page 7-34 Error: Detailed in *Validation* on page 7-34
p_amount_applied_ from	IN	NUMBER		The allocated receipt amount in receipt currency. Default: ■ For a same currency application defaults to the amount applied ■ For the cross currency application defaults to trans_to_receipt_rate * amount_applied Validate: Detailed in *Validation* on page 7-34 Error: Detailed in *Validation* on page 7-34

Parameter	Type	Data-type	Required	Description
p_trans_to_receipt_rate	IN	NUMBER		For cross currency receipts, the exchange rate used to convert an amount from a foreign currency to functional currency Default: Detailed in *Defaulting* on page 7-34 Validate: Detailed in *Validation* on page 7-34 Error: Detailed in *Validation* on page 7-34
p_discount	IN	NUMBER		Discount on the debit item, entered in the invoice currency Default: Detailed in *Defaulting* on page 7-34 Validate: Detailed in *Validation* on page 7-34 Error: Detailed in *Validation* on page 7-34
p_apply_date	IN	DATE		Date the application was applied. Default: 1. Receipt date, if receipt date >= system date 2. System date, if receipt date < system date Validate: apply date >= transaction date apply date >= receipt date Error:AR_APPLY_BEFORE_TRANSACTION AR_APPLY_BEFORE_RECEIPT
p_gl_date	IN	DATE		Date that this application will be posted to the General Ledger Default: Detailed in *Defaulting* on page 7-34 Validate: 1. Validated as per standard gl date validation described for the gl date in Create_cash routine 2. >= transaction gl date 3. >= receipt gl date Error:1. AR_INVALID_APP_GL_DATE 2. AR_VAL_GL_INV_GL 3. AR_RW_GL_DATE_BEFORE_REC_GL
p_ussgl_transaction_code	IN	VARCHAR2(30)		Code defined by public sector accounting. Default: None Validate: None Error: None

Parameter	Type	Data-type	Required	Description
p_customer_trx_line_id	IN	NUMBER(15)		The customer trx line id of the debit item to which the payment is applied. Default: From the line number if specified Validate: This should be a valid line id for the specified customer trx id. Error: AR_RAPI_TRX_LINE_ID_INVALID
p_line_number	IN	NUMBER		The line number of the debit item to which the payment is applied. Default: None Validate: None Error: AR_RAPI_TRX_LINE_NO_INVALID
p_show_closed_invoices	IN	VARCHAR2(1)		This flag decides whether to do the receipt application against closed invoices. The valid values are 'Y' and 'N' Default: 'N' Validate: Any other value is treated as 'N'. Error: None
p_event	IN	VARCHAR2(50)		The event that resulted in the creation of the receipt. Currently used only by Bills Receivables. Default: None Validate: None Error: None
p_move_deferred_tax	IN	VARCHAR2(1)		Depending on maturity date, this flag indicates when deferred tax should be moved on the accounting event. Default: None Validate: None Error: None

Parameter	Type	Data-type	Required	Description
p_attribute_record	IN	attribute_rec_ type		This is a record type which contains all the 15 descriptive flexfield segments and one descriptive flexfield structure defining column. It represents the Receipt Application Information flexfield. Default: DFF APIs used to do the defaulting and validation Validate: DFF APIs used to do the defaulting and validation Error: AR_RAPI_DESC_FLEX_INVALID
p_global_attribute_ record	IN	global_ attribute_rec_ type		This is a record type which contains all the 20 global descriptive flexfield segments and One global descriptive flexfield structure defining column. Default: None Validate: None Error: None
p_comments	IN	VARCHAR2 (240)		User's comments
p_payment_set_id	IN	NUMBER(15)		Payment set ID is populated only for a prepayment receipt that needs to be applied to a given debit item. Default: None Validate: None
p_application_ref_ type	IN	VARCHAR2(30)		Application reference type – this determines the context of the application reference fields. Default: None Validate: Must be Null or, if a Trade Management deduction is being created, then must be 'CLAIM' (Trade Management must be installed). Error: AR_RAPI_INVALID_APP_REF
p_application_ref_id	IN	NUMBER(15)		Must be NULL.

Parameter	Type	Data-type	Required	Description
p_application_ref_num	IN	VARCHAR2(30)		The reference number relating to the application reference type. If application reference type is 'CLAIM', then this would be a deduction number. Default: None Validate: If populated, then must be an existing deduction number in Trade Management. Error: AR_RAPI_INVALID_CLAIM_NUM
p_secondary_application_ref_id	IN	NUMBER(15)		The secondary application reference ID related to the application reference type. Default: None Validate: If populated, and if the application reference type is 'CLAIM', then this must contain a valid claim ID in Trade Management. Error: AR_RW_INVALID_CLAIM_ID
p_application_ref_reason	IN	VARCHAR2(30)		The reason code related to the application reference type. Default: None Validate: If populated, and if the application reference type is 'CLAIM', then this must contain a valid reason code ID in Trade Management. Error: AR_RAPI_INVALID_REF_REASON
p_customer_reference	IN	VARCHAR2(100)		Reference supplied by customer.
p_customer_reason	IN	VARCHAR2(30)		Reason code supplied by customer, in the context of an application reference type of 'CLAIM'. Default: None Validate: None in Oracle Receivables (the attempt to match to an Oracle reason code is made in Trade Management).

Defaulting

This section explains the defaulting mechanisms for the various parameters of this API, which are relatively more complex in nature and could not be explained in the Description column of the preceding table.

Trans to receipt rate

For a cross currency application, the transaction to receipt rate is defaulted by the following rules:

- Check if a fixed rate exists (using the GL APIs) between the transaction currency and the receipt currency. If yes, then get it and use it as the default.
- If there is no fixed rate relationship between the transaction currency and the receipt currency, and the profile option AR: Cross Currency Exchange Rate Type has a value, then try to derive a rate from the database using the profile option value and the cash receipt date as the exchange rate date. If you get a rate from the database, then use it as default.
- If the amount_applied and the amount_applied_from are specified, then derive the transaction to receipt rate using the following equation: *trans_to_receipt_rate=amount_applied_from/amount_applied*.

GL Date

The application GL date is defaulted to the greater of the Receipt GL date or, depending on the value of the profile option AR: Application GL Date Default, the system date or transaction GL date.

Discount

Defaults to the maximum discount available on the transaction, as of the date of application, which is internally calculated by the discounts routine.

Validation

This section explains the validation mechanisms for the various parameters of this API which are relatively more complex in nature and could not be explained in the Description column of the preceding table.

Customer Trx ID

The customer_trx_id is validated using the conditions mentioned below:

- If the Show Closed Invoices flag is set to 'Y,' then the current transaction + installment can have a payment schedule status of Closed ('CL'). Otherwise, the payment schedule status must be Open ('OP').
- If the Allow Payment of Unrelated Transactions system option = 'Y,' then the current transaction can be for a customer who is not related to the customer on the receipt. Otherwise, the transaction must be for the same or related customer on the receipt.
- The transaction must be an Invoice, Credit Memo, Debit Memo, Deposit, or Chargeback.

> **Note:** This transaction can be in a currency that is different from the receipt currency.

Depending on the specified input parameters, one of the following error messages is raised for an invalid transaction:

- AR_RAPI_TRX_ID_INST_INVALID
- AR_RAPI_TRX_NUM_INST_INVALID
- AR_RAPI_CUST_TRX_ID_INVALID
- AR_RAPI_TRX_NUM_INVALID
- AR_RAPI_APP_PS_ID_INVALID

For details of these messages, refer to *Messages* on page 7-131.

Amount Applied

- The amount applied cannot be null. The error message raised for an invalid value is AR_RAPI_APPLIED_AMT_NULL.
- The amount applied must not be greater than the line amount for the given customer_trx_line ID (if specified). The error message raised for an invalid value is AR_RW_APPLIED_GREATER_LINE.
- Depending on the creation sign, natural application flag, allow overapplication flag, and the amount due remaining of the specified transaction installment, the amount applied is validated to check for overapplication and natural application. The error messages raised for invalid values are AR_CKAP_OVERAPP, AR_CKAP_NATURALAPP, and AR_CKAP_CT_SIGN. For details of the messages, refer to *Messages* on page 7-131.

- For a cross currency application, the following equation should always be valid:
  *amount applied * trans to receipt rate = amount applied from*

  The error message raised is AR_RAPI_INVALID_CC_AMTS.

Amount Applied From

- The amount applied from cannot be null. The error message raised for an invalid value is AR_RAPI_AMT_APP_FROM_NULL.
- The amount applied from cannot be greater than the unapplied amount available on the receipt. The error message raised for invalid values is AR_RW_APP_NEG_UNAPP.
- If the transaction currency and the receipt currency are the same, then the amount applied from must always be equal to the amount applied. The error message raised for an invalid value is AR_RAPI_AMT_APP_FROM_INVALID.
- As mentioned previously for a cross currency application, the following equation must always be valid:
  *amount applied * trans to receipt rate = amount applied from*

Trans to Receipt Rate

- For a cross currency application, the trans to receipt rate should have a positive value. The error message raised for an invalid value is AR_RW_CC_RATE_POSITIVE.
- If the transaction currency and the receipt currency are the same, then the rate should not have any value specified. The error message raised for an invalid value is AR_RAPI_INVALID_CC_RATE.
- For a cross currency application, the following equation should always be valid:
  *amount applied * trans to receipt rate = amount applied from*

  If this condition is violated, then the error raised is AR_RAPI_CC_RATE_AMTS_INVALID.

Discount

- If the amount due original on the transaction (debit item) is negative, then discount = 0 or null. The error message raised for an invalid value is AR_RW_NO_DISCNT.
- If amount applied > 0, then the discount cannot be negative. The error message raised for an invalid value is AR_RW_VAL_NEG_DISCNT.

- If partial discount flag = 'N' and the transaction has not been completely paid off by the receipt application, then the discount = 0 or null. The error message raised for an invalid value is AR_NO_PARTIAL_DISC.
- The discount must not be greater than the maximum discount allowed on the transaction, which is internally calculated in the API by the discounts routine. The error message raised for an invalid value is AR_RW_VAL_DISCOUNT.

If the Allow Unearned Discounts system option = 'N,' then the discount must be less than or equal to the allowed earned discount, which gets internally calculated in the API by the discounts routine for the given transaction. The error message raised for an invalid value is AR_RW_VAL_UNEARNED_DISCOUNT.

Application Ref Num

If p_application_ref_type is 'CLAIM', then the application reference number can be populated with a valid deduction number from Trade Management. This deduction/overpayment must be in the same currency as the debit item being applied to. Otherwise, the error message raised is AR_RAPI_INVALID_CLAIM_NUM.

Secondary Application Ref ID

If p_application_ref_type is 'CLAIM', then the secondary application reference ID can be populated with a valid claim ID from Trade Management. This deduction/overpayment must be in the same currency as the debit item being applied to. Otherwise, the error message raised is AR_RAPI_INVALID_CLAIM_NUM.

If both the application reference number and the secondary application reference ID are left null, and p_application_ref_type is 'CLAIM', then a new claim will be created in Trade Management.

Example

Objective:

To apply a cash receipt in functional currency to an invoice in functional currency having only one installment using a call to the API *Ar_receipt_api_pub.Apply* and passing a minimum number of input parameters.

This table lists the entered parameters:

Parameter	Entered Value	Default Value
p_api_version	1.0	
p_trx_number	'aj_test_trx_1'	
p_receipt_number	'aj_test_cr_2'	

This table lists the defaulted input parameters, which were not entered:

Parameter	Entered Value	Default Value
p_customer_trx_id		187807
p_installment		1
p_cr_id		23927
p_gl_date		10-FEB-2000
p_applied_payment_ schedule_id		36271
p_apply_date		10-FEB-2000
p_amount_applied		98
p_amount_applied_from		98
p_discount		2
p_show_closed_invoices		'N'

Result:

We were able to apply the cash receipt against the specified transaction by specifying only three input parameters in our call to this API. The retrieval and handling of the warnings and the error messages, put on the message stack by the API during execution, are the same as described in *Defaulting* on page 7-24.

Ar_receipt_api_pub.Create_and_apply

Description

Call this routine to create a cash receipt and apply it to a specified installment of a debit item. This debit item could be for the same customer or related customer, or for an unrelated customer, depending on the Allow Payment of Unrelated Transactions system option.

This is essentially a superset of the *ar_receipt_api_pub.Create_cash* and *Ar_receipt_api_pub.Apply* APIs, and contains the same parameters as contained in those two APIs. During the call to this API, if the creation of the receipt is successfully completed but its application to the debit item fails, then the receipt creation is also rolled back.

This routine calls Oracle *i*Payment, where required. See *Integration with Oracle iPayment* on page 7-10.

> **Note:** To create credit card receipts that need to be processed by *i*Payment APIs, you must pass the p_call_payment_processor parameter as fnd_api.g_true. Additionally, you must specify the p_customer_bank_account_id parameter.

This API routine has 3 output and 57 input parameters in total. Based on the type, the following is the breakdown of the parameters:

Input

Standard API parameters:	4	
Application parameters:	45	+ 2 (descriptive flexfield record parameter)
		+ 2 (global descriptive flexfield record parameter)

Output

Standard API parameters:	3
Application parameters:	0

The description of the seven standard API parameters is the same as already given in *Description* on page 7-11.

The following table lists the parameters that are relevant to the receipt creation and application for the API.

Note: If required parameters are not passed in a call to this API, then the call will fail. However, depending on the business scenario, you will have to pass in values for other parameters to successfully create the business object. Otherwise, error messages will be reported.

Parameter	Type	Data-type	Required	Description
p_usr_currency_code	IN	VARCHAR2		The translated currency code. Used to derive the p_currency_code if it is not entered. Default: None Validate: Should be a valid currency, so that we can derive the corresponding currency code. Error: AR_RAPI_USR_CURR_CODE_INVALID
p_currency_code	IN	VARCHAR2(15)		The actual currency code that gets stored in AR tables. Default: 1. Derived from p_usr_currency_code if entered. Otherwise, 2. Defaulted to the functional currency code. Validate: Validated against the currencies in fnd_currencies table. Error: AR_RAPI_CURR_CODE_INVALID Warning: AR_RAPI_FUNC_CURR_DEFAULTED
p_usr_exchange_rate_type	IN	VARCHAR2		The translated exchange rate type. Used to derive the p_exchange_rate_type if it has not been entered. Default: None Validate: Should be a valid rate type. Error: AR_RAPI_USR_X_RATE_TYP_INVALID

Parameter	Type	Data-type	Required	Description
p_exchange_rate_type	IN	VARCHAR2(30)		Exchange rate type stored in AR tables. Default: 1. In case of foreign currency receipt, derived from p_usr_exchange_rate_type 2. If p_usr_exchange_rate_type is null, then defaulted from AR: Default Exchange Rate Type profile option 3. Should be left null, if the receipt is in the same denomination as functional currency Validate: 1. Validated against values in gl_daily_conversion_types table Error: AR_RAPI_X_RATE_TYPE_INVALID
p_exchange_rate	IN	NUMBER		The exchange rate between the receipt currency and the functional currency. Default: 1. Derived from the Daily Rates table for rate_type <>'User' in case of non-functional currency 2. If profile option Journals: Display Inverse Rate = 'Y', set user entered value to 1/ p_exchange_rate 3. The entered value is rounded to a precision of 38. Validate: 1. In case of non-functional currency the rate should have a positive value for rate type= 'User' 2. For non-functional currency and type <> 'User' the user should not specify any value. Error: AR_RAPI_X_RATE_INVALID AR_RAPI_X_RATE_NULL
p_exchange_rate_date	IN	DATE		The date on which the exchange rate is valid. Default: Receipt date Validate: 1. For a non-functional currency and type <> 'User' there should be a valid rate existing in the database for this date. This is a cross validation of type, currency and date Error: AR_NO_RATE_DATA_FOUND

Parameter	Type	Data-type	Required	Description
p_amount	IN	NUMBER	Yes	The cash receipt amount. Default: Null Validate: >0 Error: AR_RAPI_REC_AMT_NEGATIVE AR_RAPI_RCPT_AMOUNT_NULL
p_factor_discount_ amount	IN	NUMBER		The bank charges on the cash receipt. Default: None Validate: 1. Bank charges are not allowed if profile option AR: Create Bank Charges = 'No'. 2. Bank charges not allowed if the receipt state, derived from the receipt class of the receipt method, <> 'CLEARED'. 3. If allowed, then >= 0. Error: AR_BK_CH_NOT_ALLWD_IF_NOT_CLR AR_JG_BC_AMOUNT_NEGATIVE
p_receipt_number	IN	VARCHAR2(30)		The receipt number of the receipt to be created. Default: If not specified, the receipt number is defaulted from the document sequence value. Validate: Receipt number should not be null Error: AR_RAPI_RCPT_NUM_NULL
p_receipt_date	IN	DATE		The receipt date of the entered cash receipt. Default: System date Validate: None Error: None

Parameter	Type	Data-type	Required	Description
p_gl_date	IN	DATE		Date that this receipt will be posted to the General Ledger. Default: Gets defaulted to the receipt date if it is a valid gl_date. Validate: The date is valid if the following conditions are true: ■ The date is in an Open or Future period ■ The period cannot be an Adjustment period If the date is invalid, then: ■ If the most recent open period is prior to the receipt date: last date of that period ■ If there is a period open after the receipt date: first date of the last open period Error: AR_INVALID_APP_GL_DATE
p_maturity_date	IN	DATE		Receipt maturity date. Default: Deposit date Validate: >= p_receipt_date Error: AR_RW_MAT_BEFORE_RCT_DATE
p_customer_id	IN	NUMBER(15)		The customer_id for the paying customer. Default: Refer to *Defaulting* on page 7-53 Validate: 1. Customer exists and has prospect code = 'CUSTOMER' 2. Customer has a profile defined a customer level Error: AR_RAPI_CUST_ID_INVALID
p_customer_name	IN	VARCHAR2(50)		The name for the entered customer. Used to default the customer id if not specified. Default: None Validate: None Error: AR_RAPI_CUS_NAME_INVALID

Parameter	Type	Data-type	Required	Description
p_customer_number	IN	VARCHAR2(30)		The customer number. Used to default the customer_id if not specified. Default: None Validate: None Error: AR_RAPI_CUS_NUM_INVALID
p_customer_bank_account_id	IN	NUMBER(15)		The customer bank account ID. Default: From bank account ID/number. Validate: 1. It must be a valid bank account of the paying customer. 2. The inactive date (if defined) of the bank account should be greater than the receipt_date. 3. The receipt date must be within the Start date and the End date of the bank account uses. Error: AR_RAPI_CUS_BK_AC_2_INVALID AR_RAPI_CUS_BK_AC_ID_INVALID
p_customer_bank_account_num	IN	VARCHAR2(30)		The customer bank account number. Used to default the customer bank account id, if not specified. Default: None Validate: None Error: None
p_customer_bank_account_name	IN	VARCHAR2(80)		The customer bank account name. Used to default the customer bank account id, if not specified. Default: None Validate: None Error: None
p_customer_location	IN	VARCHAR2(40)		The Bill_To location for the customer. Used to derive the p_customer_site_use_id. Default: None Validate: None Error: AR_RAPI_CUS_LOC_INVALID

Parameter	Type	Data-type	Required	Description
p_customer_site_use_id	IN	NUMBER(15)		The Bill_To site_use_id for the customer. Default: 1. Defaulted from customer location. Otherwise, 2. Primary Bill_To customer site_use_id of the customer. Validate: It should be a valid Bill_To site of the paying customer. Error: AR_RAPI_CUS_SITE_USE_ID_INVALID
p_customer_receipt_reference	IN	VARCHAR2(30)		This column is used to store a customer receipt reference value that the customer supplies at the confirmation time. Default: None Validate: None Error: None
p_override_remit_bank_account_flag	IN	VARCHAR2(1)		The flag value decides when the remittance bank account can be overridden by the remittance selection process. Default: 'Y' Validate: valid values 'Y' and 'N' Error: AR_RAPI_INVALID_OR_REMIT_BK_AC
p_remittance_bank_account_id	IN	NUMBER(15)		Identifies the user's bank account for depositing the receipt. Default: 1. From remittance bank account number 2. From the receipt method based on logic mentioned in *Defaulting* on page 7-24 Validate: Validation logic detailed in *Validation* on page 7-23 Error: AR_RAPI_REM_BK_AC_ID_INVALID AR_RAPI_REM_BK_AC_ID_NULL

Parameter	Type	Data-type	Required	Description
p_remittance_bank_ account_num	IN	VARCHAR2(30)		The remittance bank account number. Used to default the remittance bank account id if not specified. Default: None Validate: None Error: AR_RAPI_REM_BK_AC_NUM_INVALID
p_remittance_bank_ account_name	IN	VARCHAR2(50)		The remittance bank account name. Used to default the remittance bank account id if not specified. Default: None Validate: None Error: AR_RAPI_REM_BK_AC_NAME_INVALID
p_deposit_date	IN	DATE		The deposit date. Default: receipt date Validate: None Error: None
p_receipt_method_id	IN	NUMBER(15)		Identifies the payment method of the receipt. Default: From receipt method name Validate: Validation detailed in *Validation* on page 7-23 Error: AR_RAPI_INVALID_RCT_MD_ID

Parameter	Type	Data-type	Required	Description
p_receipt_method_name	IN	VARCHAR2(30)		The payment method name of the receipt. Used to default the receipt method id if not specified. Default: None Validate: None Error: None **Note:** To use credit card refund functionality, ensure that remittance of the original receipt is performed within Oracle Receivables. Do this by setting the remittance method on the payment method's associated receipt class to *Standard*. **Caution:** If you use this API to both authorize and capture credit card payments, then set the remittance method to *None*. Note, however, that with this setting, you cannot use standard credit card refund functionality. Instead, you must refund such payments *outside* Receivables.
p_doc_sequence_value	IN	NUMBER		Value assigned to document receipt. Default: Detailed in *Defaulting* on page 7-24. Validate: ■ You should not pass a value, if the current document sequence is automatic. ■ Document sequence value should not be entered if profile option Sequential Numbering is set to Not Used. Error: AR_RAPI_DOC_SEQ_AUTOMATIC AR_RAPI_DOC_SEQ_VAL_INVALID
p_ussgl_transaction_ code	IN	VARCHAR2(30)		Code defined by public sector accounting. Default: None Validate: None Error: None

Parameter	Type	Data-type	Required	Description
p_anticipated_clearing_date	IN	DATE		Date the receipt is expected to be cleared. Default: None Validate: >= gl_date Error: AR_RW_EFFECTIVE_BEFORE_GL_DATE
p_event	IN	VARCHAR2		The event that resulted in the creation of the receipt. Currently used only by Bills Receivables. Default: None Validate: None Error: None
p_called_from	IN	VARCHAR2(20)		This parameter is used to identify the calling routine. Currently used to identify only the 'BR_REMIT' program. Default: None Validate: None Error: None
p_attribute_record	IN	attribute_rec_type		This is a record type which contains all the 15 descriptive flexfield segments and one descriptive flexfield structure defining column. It represents the Receipt Information flexfield. Default: DFF APIs used to do the defaulting and validation Validate: DFF APIs used to do the defaulting and validation Error: AR_RAPI_DESC_FLEX_INVALID
p_global_attribute_record	IN	global_attribute_rec_type		This is a record type which contains all the 20 global descriptive flexfield segments and one global descriptive flexfield structure defining column. Default: None Validate: None Error:

Parameter	Type	Data-type	Required	Description
p_issuer_name	IN	VARCHAR2(50)		Issuer name of Notes Receivable (Asia Pacific Requirement). Default: None Validate: None Error: None
p_issue_date	IN	DATE		Date when the note receivable was issued (Asia Pacific Requirement). Default: None Validate: None Error: None
p_customer_trx_id	IN	NUMBER(15)		The customer_trx_id of the debit item to which the receipt is to be applied. Default: None Validate: Detailed in *Validation* on page 7-34 Error: Detailed in *Validation* on page 7-34
p_trx_number	IN	VARCHAR2(20)		The trx_number of the debit item to which the receipt is to be applied. Used to default the customer_trx_id. Default: None Validate: None Error: AR_RAPI_TRX_NUM_INVALID
p_installment	IN	NUMBER(15)		The installment (or term_sequence_ number) of the debit item. Used in conjunction with customer_trx_id to derive the applied payment schedule id if not specified. Default: 1, if only one installment exists for the debit item Validate: 1) >0; 2) valid installment of transaction. Also see *Validation* on page 7-34 Error: AR_RAPI_INSTALL_NULL

Parameter	Type	Data-type	Required	Description
p_applied_payment_schedule_id	IN	NUMBER(15)		The payment schedule id of the debit item. Also used to derive the customer_trx_id if not specified. Default: Defaulted based on the installment and the customer_trx_id Validation: 1. > 0; 2. It must correspond to Customer trx id and installment specified. 3. It must have the status <> 'CL' if the show closed invoices flag <> 'Y' Error: AR_RAPI_APP_PS_ID_INVALID
p_amount_applied	IN	NUMBER		The transaction amount to which the receipt is to be applied. This in the transaction currency. Default: Depending on the profile option AR: Cash-Default Amount Applied, it is defaulted either to: ■ the open amount of the transaction, or ■ the unapplied amount of the receipt. Discounts, if applicable, are taken into account by the discounts routine which calculates the amount applied. Validate: Detailed in *Validation* on page 7-34. Error: Detailed in *Validation* on page 7-34.
p_amount_applied_from	IN	NUMBER		The allocated receipt amount in receipt currency. Default: ■ For a same currency application, defaults to the amount applied. ■ For the cross currency application, defaults to trans_to_receipt_rate * amount_applied. Validate: Detailed in *Validation* on page 7-34. Error: Detailed in *Validation* on page 7-34.

Parameter	Type	Data-type	Required	Description
p_trans_to_receipt_rate	IN	NUMBER		For cross currency receipts, the exchange rate used to convert an amount from a foreign currency to functional currency. Default: Detailed in *Defaulting* on page 7-34 Validate: Detailed in *Validation* on page 7-34 Error: Detailed in *Validation* on page 7-34
p_discount	IN	NUMBER		Discount on the debit item, entered in the invoice currency. Default: Detailed in *Defaulting* on page 7-34 Validate: Detailed in *Validation* on page 7-34 Error: Detailed in *Validation* on page 7-34
p_apply_date	IN	DATE		Date the application was applied. Default: 1. Receipt date, if receipt date >= system date 2. System date, if receipt date < system date Validate: apply date >= transaction date apply date >= receipt date Error: AR_APPLY_BEFORE_TRANSACTION AR_APPLY_BEFORE_RECEIPT
p_apply_gl_date	IN	DATE		Date that this application will be posted to the General Ledger. Default: Detailed in *Defaulting* on page 7-34 Validate: 1. Validated as per standard gl date validation described for the gl date in Create_cash routine 2. Greater than or equal to transaction gl date 3. Greater than or equal to receipt gl date Error: 1. AR_INVALID_APP_GL_DATE 2. AR_VAL_GL_INV_GL 3. AR_RW_GL_DATE_BEFORE_REC_GL

Parameter	Type	Data-type	Required	Description
p_app_ussgl_ transaction_code	IN	VARCHAR2(30)		Code defined by public sector accounting. Default: None Validate: None Error:
p_customer_trx_line_id	IN	NUMBER(15)		The customer trx line id of the debit item to which the payment is applied. Default: From the line number if specified Validate: This should be a valid line id for the specified customer trx id. Error: AR_RAPI_TRX_LINE_ID_INVALID
p_line_number	IN	NUMBER		The line number of the debit item to which the payment is applied. Default: None Validate: None Error: AR_RAPI_TRX_LINE_NO_INVALID
p_show_closed_invoices	IN	VARCHAR2(1)		This flag decides whether to do the receipt application against closed invoices. The valid values are 'Y' and 'N'. Default: 'N' Validate: Check for the valid values. Error: AR_RAPI_INVALID_SHOW_CL_INV
p_event	IN	VARCHAR2(50)		The event that resulted in the creation of the receipt. Currently used only by Bills Receivables. Default: None Validate: None Error: None

Parameter	Type	Data-type	Required	Description
p_move_deferred_tax	IN	VARCHAR2(1)		Depending on maturity date, this flag indicates when deferred tax should be moved on the accounting event. Default: None Validate: None Error: None
p_app_attribute_record	IN	attribute_rec_ type		This is a record type which contains all the 15 descriptive flexfield segments and one descriptive flexfield structure defining column. It represents the Receipt Application Information flexfield. Default: DFF APIs used to do the defaulting and validation Validate: DFF APIs used to do the defaulting and validation Error: AR_RAPI_DESC_FLEX_INVALID
p_app_global_attribute_ record	IN	global_attribute_ rec_type		This is a record type which contains all the 20 global descriptive flexfield segments and one global descriptive flexfield structure defining column. Default: None Validate: None Error:
p_comments	IN	VARCHAR2(240)		User's comments for the application.
p_call_payment_ processor	IN	VARCHAR2 (1)	FND_ API.G_ FALSE	This is the payment processing indicator flag. Pass as FND_API.G_TRUE, if you want to call *i*Payment payment APIs for credit card processing.

Defaulting

This section explains the defaulting mechanisms for the various parameters of this API which are relatively more complex in nature and could not be explained in the Description column of the preceding table.

Customer ID

The p_customer_id is required for the create_and_apply routine because an unidentified receipt cannot be applied to a transaction. If not specified, then the customer ID gets defaulted from one of the following:

- Customer number, customer name, or both
- Bill_to customer on the transaction or drawee customer on the bill (for receipt application against a bill)

If the customer ID is not defaulted by one of the above, then the AR_RAPI_CUST_ID_NULL error is raised.

Example

Objective:

To create a cash receipt in the functional currency against an invoice in USD having only one installment, using a call to the API *Ar_receipt_api_pub.Create_and_Apply* and passing a minimum number of input parameters.

This table lists the entered parameters:

Parameter	Entered Value	Default Value
p_api_version	1.0	
p_receipt_number	'aj_test_api_3'	
p_amount	1000	
p_receipt_method_id	1001	
p_customer_name	'Computer Service and Rentals'	
p_trx_number	'aj_test_trx_3'	

This table lists the defaulted input parameters, which were not entered:

Parameter	Entered Value	Default Value
p_customer_id		1006
p_currency_code		USD
p_receipt_date		10-FEB-2000

Parameter	Entered Value	Default Value
p_gl_date		10-FEB-2000
p_deposit_date		10-FEB-2000
p_customer_site_use_id		1025
p_override_remit_ bank_account_flag		'Y'
p_remittance_bank_ account_id		10001
p_maturity_date		10-FEB-2000
p_customer_trx_id		187809
p_installment		1
p_apply_gl_date		10-FEB-2000
p_applied_payment_ schedule_id		36277
p_apply_date		10-FEB-2000
p_amount_applied		1000
p_amount_applied_ from		1000
p_discount		0
p_show_closed_ invoices		'N'

Result:

We were able to create the cash receipt 'aj_test_api_3' and then apply it against the invoice 'aj_test_trx_3' by specifying only six input parameters in our call to this API. Both the receipt and the invoice are in the functional currency. The retrieval and handling of the warnings and the error messages, put on the message stack by the API during execution, are the same as described in *Defaulting* on page 7-24.

Ar_receipt_api_pub.Unapply

Description

Call this routine to unapply a cash receipt application against a specified installment of a debit item or payment_schedule_id. This API routine has 3 output and 14 input parameters in total. Based on the type, the following is the breakdown of the parameters:

Input

Standard API parameters: 4

Application parameters: 10

Output

Standard API parameters: 3

Application parameters: 0

The description of the seven standard API parameters is the same as given in *Description* on page 7-11.

The following table lists the parameters that are specific to the unapplication for the API.

> **Note:** If required parameters are not passed in a call to this API, then the call will fail. However, depending on the business scenario, you will have to pass in values for other parameters to successfully create the business object. Otherwise, error messages will be reported.

Parameter	Type	Data-type	Required	Description
p_cr_id	IN	NUMBER(15)		The cash_receipt_id of the receipt whose application has to be unapplied. Default: None Validate: 1. Status must not be Reversed or Approved 2. The receipt should have an application on it. Error: AR_RAPI_CASH_RCPT_ID_NULL

Parameter	Type	Data-type	Required	Description
p_receipt_number	IN	VARCHAR2(30)		The receipt number of the receipt whose application is to be unapplied. Used to default the cash_receipt_id. Default: None Validate: None Error: AR_RAPI_RCPT_NUM_INVALID AR_RAPI_TRX_NUM_INST_INVALID
p_customer_trx_id	IN	NUMBER(15)		The customer_trx_id of the debit item against which the specified receipt has an application. Default: None Validate: The transaction must have an application against the specified receipt. Error: AR_RAPI_CUST_TRX_ID_INVALID AR_RAPI_TRX_ID_INST_INVALID
p_trx_number	IN	VARCHAR2(20)		The trx_number of the debit item against which the specified receipt has an application. Used to default the customer_trx_id. Default: None Validate: None Error: None
p_installment	IN	NUMBER(15)		The installment (or term_sequence_number) of the debit item. Used in conjunction with customer_trx_id to derive the applied payment schedule id if not specified. Default: 1, if only one installment exists for the debit item Validate: 1) >0; 2) valid installment of transaction Error: AR_RAPI_INSTALL_NULL AR_RAPI_TRX_ID_INST_INVALID AR_RAPI_TRX_NUM_INST_INVALID

Parameter	Type	Data-type	Required	Description
p_applied_ payment_ schedule_id	IN	NUMBER(15)		The payment schedule id of the debit item. Also used to derive the customer_trx_id, if not specified. Default: Derived from the installment and the customer_trx_id. Validation: 1. > 0 2. It must correspond to Customer trx id and installment, if specified. 3. For applications with Bills Receivables installed, you cannot unapply a bill that is in the process of remittance. Error: AR_RAPI_APP_PS_ID_INVALID
p_receivable_ application_id	IN	NUMBER(15)		Identifies the receivable application. Used to derive the customer_trx_id, cash_receipt_id, and the applied_payment_schedule_id, if not specified. Default: Defaulted from the specified transaction and the receipt. Validate: 1. Application type must be 'CASH'. 2. Display flag = 'Y' (latest application). 3. The applied payment schedule id of the receivable application record must correspond to the p_applied_payment_schedules_id, if specified. 4. The cash receipt id must correspond to the cash receipt id specified. 5. For applications with Bills Receivables installed, you cannot unapply the application of a bill that is in the process of remittance. Error: AR_RAPI_REC_APP_ID_NULL AR_RAPI_REC_APP_ID_INVALID

Parameter	Type	Data-type	Required	Description
p_reversal_gl_date	IN	DATE		The reversal gl date. Default: Gets defaulted to the application gl date if it is a valid gl_date. Validate: 1. It is valid if the following conditions are true: --The date is in an Open or Future period. --The period cannot be an Adjustment period. 2. The reversal GL date >= application GL date. 3. The reversal GL date >= receipt GL date. If the date is invalid, then: ■ If the most recent open period is prior to the receipt date: last date of that period ■ If there is a period open after the receipt date: first date of the last open period Error: AR_INVALID_APP_GL_DATE AR_RW_BEFORE_APP_GL_DATE AR_RW_BEFORE_RECEIPT_GL_DATE
p_called_from	IN	VARCHAR2(20)		This parameter is used to identify the calling routine. Default: None Validate: None Error: None
p_cancel_claim_flag	IN	VARCHAR2(1)		Not used – leave null.

Defaulting

This section explains the defaulting mechanisms for the various parameters of this API which are relatively more complex and could not be explained in the Description column of the preceding table.

Receivable Application ID

If not specified, then the receivable application ID can be defaulted by one of the following:

- Using the specified installment and p_customer_trx_id (derived from p_trx_number if not specified) and p_cr_id (derived from the receipt number if not specified).

- Using the specified value of p_applied_payment_schedule_id and p_cr_id (derived from the receipt number if not specified).

Validation

This section explains the cross validations for the various parameters of this API which are relatively more complex and could not be explained in the Description column of the preceding table.

Cross validation between customer_trx_id, applied_payment_schedule_id, cash_receipt_id, and receivable_application_id

- If p_customer_trx_id, p_installment, and p_applied_payment_schedule_id are specified and the two do not point to the same transaction, then the error AR_RAPI_TRX_PS_ID_X_INVALID is raised.
- If the combination of the specified p_applied_payment_schedule_id (or derived from the p_customer_trx_id and p_installment) and the specified p_receivable_application_id is invalid, then the error AR_RAPI_APP_PS_RA_ID_X_INVALID or AR_RAPI_TRX_RA_ID_X_INVALID is raised, depending on the input parameters.

Example

Objective:

To unapply the receipt application against an invoice using the call to API *Ar_receipt_api_pub.Unapply* and passing a minimum number of input parameters.

This table lists the entered parameters:

Parameter	Entered Value	Default Value
p_api_version	1.0	
p_receipt_number	'aj_test_api_4'	
p_applied_payment_schedule_id	1001	

This table lists the defaulted input parameters, which were not entered:

Parameter	Entered Value	Default Value
p_cr_id		1006

Parameter	Entered Value	Default Value
p_customer_trx_id		USD
p_reversal_gl_date		10-FEB-2000
p_receivable_application_id		29711

The retrieval and handling of the warnings and the error messages, put on the message stack by the API during execution, are the same as described in *Defaulting* on page 7-24.

Ar_receipt_api_pub.Apply_on_account

Description

Call this routine to apply an on-account application of the specified cash receipt. This API routine has 3 output and 21 input parameters in total. Based on the type, the following is the breakdown of the parameters:

Input

Standard API parameters:	4		
Application parameters:	14	+	1 (descriptive flexfield record type)
		+	1 (global descriptive flexfield record type)

Output

Standard API parameters:	3
Application parameters:	0

The description of the seven standard API parameters is the same as given in *Description* on page 7-11.

The following table lists the descriptions of the on-account application-related parameters of the API.

> **Note:** If required parameters are not passed in a call to this API, then the call will fail. However, depending on the business scenario, you will have to pass in values for other parameters to successfully create the business object. Otherwise, error messages will be reported.

Parameter	Type	Data-type	Required	Description
p_cr_id	IN	NUMBER(15)		The cash_receipt_id of the receipt which is to be applied on account. Default: None Validate: 1. Type must be 'CASH' 2. Status must not be Reversed or Approved 3. The receipt must not be Unidentified Error: 1. AR_RAPI_CASH_RCPT_ID_INVALID 2. AR_RAPI_CASH_RCPT_ID_NULL
p_receipt_number	IN	VARCHAR2(30)		The receipt number of the receipt to be applied on account. Used to default the cash_receipt_id. Default: None Validate: None Error: AR_RAPI_RCPT_NUM_INVALID
p_amount_applied	IN	NUMBER		The amount on the cash receipt that is to be applied on account. Default: Amount due remaining on the receipt. Validate: 1. Greater than or equal to 0. 2. Less than or equal to the amount due remaining on the receipt. Error: 1. AR_RAPI_APPLIED_AMT_NULL 2. AR_RW_APP_NEG_UNAPP 3. AR_RW_AMOUNT_LESS_THAN_APP
p_apply_date	IN	DATE		Date the application was applied. Default: 1. Receipt date, if receipt date >= system date 2. System date, if receipt date < system date Validate: apply date >= receipt date Error: AR_APPLY_BEFORE_RECEIPT

Parameter	Type	Data-type	Required	Description
p_apply_gl_date	IN	DATE		Date that this application will be posted to the General Ledger. Default: Defaulted to greater of the receipt date and the system date. Validate: 1. Validated as per standard gl date validation described for the gl date in Create_cash routine. 2. >= receipt gl date. Error: 1. AR_INVALID_APP_GL_DATE 2. AR_RW_GL_DATE_BEFORE_REC_GL
p_ussgl_transaction_code	IN	VARCHAR2(30)		Code defined by public sector accounting. Default: None Validate: None Error: None
p_attribute_rec	IN	attribute_rec_type		This is a record type which contains all the 15 descriptive flexfield segments and one descriptive flexfield structure defining column. It represents the Receipt Application Information flexfield. Default: DFF APIs used to do the defaulting and validation Validate: DFF APIs used to do the defaulting and validation Error: AR_RAPI_DESC_FLEX_INVALID
p_global_attribute_rec	IN	global_attribute_rec_type		This is a record type which contains all the global descriptive flexfields: One global descriptive flexfield structure defining column and 20 segments. Default: None Validate: None Error: None
p_comments	IN	VARCHAR2(240)		User comments.
p_application_ref_num	IN	VARCHAR2(30)		Deduction number, if resulting from Trade Management claim settlement.

Parameter	Type	Data-type	Required	Description
p_secondary_ application_ref_id	IN	NUMBER(15)		Claim ID, if resulting from Trade Management claim settlement.
p_customer_reference	IN	VARCHAR2(100)		Reference supplied by customer.
p_called_from	IN	VARCHAR2(20)		This parameter is used to identify the calling routine. Default: None Validate: None Error: None
p_customer_reason	IN	VARCHAR2(30)		Reason code supplied by customer.
p_secondary_app_ ref_type	IN	VARCHAR2(30)		Used for automated receipt handling. Leave null.
p_secondary_app_ ref_num	IN	VARCHAR2(30)		Used for automated receipt handling. Leave null.

Note: With an on-account application, you cannot apply a negative amount, as you can do in a regular application of a receipt to a debit item.

Example

Objective:

To apply a cash receipt in the functional currency to an invoice in the functional currency having only one installment, using a call to the API *Ar_receipt_api_pub.Apply_on_account* and passing a minimum number of input parameters.

This table lists the entered parameters:

Parameter	Entered Value	Default Value
p_api_version	1.0	
p_receipt_ number	'aj_test_cr_2'	

This table lists the defaulted input parameters, which were not entered:

Parameter	Entered Value	Default Value
p_cr_id		23927
p_gl_date		01-JUN-2000
p_apply_date		01-JUN-2000
p_amount_applied		100

The retrieval and handling of the warnings and error messages, put on the message stack by the API during execution, are the same as described in *Defaulting* on page 7-24.

Ar_receipt_api_pub.Unapply_on_account

Description

Call this routine to unapply an on-account application on the specified cash receipt. This API routine has 3 output and 8 input parameters in total. Based on the type, the following is the breakdown of the parameters:

Input

Standard API parameters: 4

Application parameters: 4

Output

Standard API parameters: 3

Application parameters: 0

The description of the seven standard API parameters is the same as already given in *Description* on page 7-11.

The following table lists the parameters that are relevant to the on-account unapplication for the API.

> **Note:** If required parameters are not passed in a call to this API, then the call will fail. However, depending on the business scenario, you will have to pass in values for other parameters to successfully create the business object. Otherwise, error messages will be reported.

Parameter	Type	Data-type	Required	Description
p_cr_id	IN	NUMBER(15)		The cash_receipt_id of the receipt whose application has to be unapplied. Default: None Validate: 1. Status must not be Reversed or Approved. 2. The receipt must have an on-account application on it. Error: AR_RAPI_CASH_RCPT_ID_INVALID

Parameter	Type	Data-type	Required	Description
p_receipt_number	IN	VARCHAR2 (30)		The receipt number of the receipt which is to be unapplied. Used to default the cash_receipt_id. Default: None Validate: None Error: AR_RAPI_RCPT_NUM_INVALID
p_receivable_ application_id	IN	NUMBER(15)		Identifies the receivable application. Used to derive the customer trx id, cash_receipt_id and the applied_ps_id, if not specified. Default: Refer to *Validation* on page 7-73. Validate: 1. Application type = 'CASH'. 2. Display flag = 'Y' (latest application) and status = 'ACC'. 3. The applied payment schedule id of the receivable application record must correspond to the p_applied_payment_schedules_id, if specified. 4. The cash receipt id must correspond to the cash receipt id specified. Error: AR_RAPI_REC_APP_ID_INVALID
p_reversal_gl_date	IN	DATE		The reversal gl date. Default: Gets defaulted to the application gl date if it is a valid gl_date. Validate: 1. It is valid if the following conditions are true: --The date is in an Open or Future period. --The period cannot be an Adjustment period. 2. The reversal GL date >= application GL date. 3. The reversal GL date >= receipt GL date. If the date is invalid, then: ■ If the most recent open period is prior to the receipt date: last date of that period ■ If there is a period open after the receipt date first date of the last open period Error: AR_INVALID_APP_GL_DATE AR_RW_BEFORE_APP_GL_DATE AR_RW_BEFORE_RECEIPT_GL_DATE

Defaulting

This section explains the defaulting mechanisms for the various parameters of this API which could not be explained in the Description column of the preceding table.

Receivable Application ID

The value for p_receivable_application_id, if not specified, is defaulted from the p_cr_id (or p_receipt_number). If the receipt does not have an on-account application, then the error AR_RAPI_CASH_RCPT_ID_INVALID is raised. If there is more than one on-account application on the receipt and the value for p_receivable_application_id has not been specified, then the error AR_RAPI_MULTIPLE_ON_AC_APP is raised.

Example

Objective:

To unapply the receipt application using the call to API *Ar_receipt_api_pub.Unapply_on_account* and passing a minimum number of input parameters.

This table lists the entered parameters:

Parameter	Entered Value	Default Value
p_api_version	1.0	
p_receipt_number	'aj_test_api_6'	

This table lists the defaulted input parameters, which were not entered:

Parameter	Entered Value	Default Value
p_cr_id		20338
p_reversal_gl_date		01-JUN-2000

The retrieval and handling of the warnings and error messages, put on the message stack by the API during execution, are the same as described in *Defaulting* on page 7-24.

Ar_receipt_api_pub.Reverse

Description

Call this routine to reverse cash as well as miscellaneous receipts. This API routine has 3 output and 14 input parameters in total. Based on the type, the following is the breakdown of the parameters:

Input

Standard API parameters: 4

Application parameters: 11 + 1 (descriptive flexfield record type)

1 (global descriptive flexfield record type)

Output

Standard API parameters: 3

Application parameters: 0

The description of the seven standard API parameters is the same as given in *Description* on page 7-11.

The following table lists the descriptions of the reversal-related parameters of the API.

> **Note:** If required parameters are not passed in a call to this API, then the call will fail. However, depending on the business scenario, you will have to pass in values for other parameters to successfully create the business object. Otherwise, error messages will be reported.

Parameter	Type	Data-type	Required	Description
p_cr_id	IN	NUMBER(15)		The cash_receipt_id of the receipt which needs to be reversed. Default: None Validate: Detailed in *Defaulting* on page 7-69. Error:

Parameter	Type	Data-type	Required	Description
p_receipt_number	IN	VARCHAR2(30)		The receipt number of the receipt to be reversed. Used to default the cash_receipt_id. Default: None Validate: None Error: AR_RAPI_RCPT_NUM_INVALID
p_reversal_ category_code	IN	VARCHAR2(20)		Identifies the reason why the payment entry was reversed. Default: None Validate: Validated against the values in ar_ lookups for lookup_type = 'REVERSAL_ CATEGORY_TYPE Error: 1. AR_RAPI_REV_CAT_CD_NULL 2. AR_RAPI_REV_CAT_CD_INVALID
p_reversal_ category_name	IN	VARCHAR2(80)		This is the translated lookup meaning for the reversal category code. Used to default the reversal category code if not specified. Default: None Validate: None Error: AR_RAPI_REV_CAT_NAME_INVALID
p_reversal_gl_date	IN	DATE		The General Ledger Date that is used to credit the Account CCID for the reversed receipt. Default: System date Validate: 1. Validated as per standard gl date validation described for the gl date in Create_cash routine 2. Greater than or equal to receipt gl date Error: 1. AR_INVALID_APP_GL_DATE 2. AR_RW_BEFORE_RECEIPT_GL_DATE
p_reversal_date	IN	DATE		Date on which the payment entry reversed Default: ■ System date if system date >= receipt date, else ■ Receipt date if receipt date > system date Validate: Greater than or equal to receipt date Error: AR_RW_REV_BEFORE_RCT_DATE

Parameter	Type	Data-type	Required	Description
p_reversal_reason_code	IN	VARCHAR2(30)		Indicates the reason for reversing receipt Default: None Validate: Validated against the values in ar_lookups for lookup_type = 'CKAJST_REASON' Error: AR_RAPI_REV_REAS_CD_INVALID AR_RAPI_REV_REAS_CD_NULL
p_reversal_reason_name	IN	VARCHAR2(80)		This is the translated lookup meaning for reversal reason code. Used for defaulting the reversal reason code if not specified. Default: None Validate: None Error: AR_RAPI_REV_REAS_NAME_INVALID
p_reversal_comments	IN	VARCHAR2 (240)		Comments regarding reversal
p_atttribute_rec	IN	p_attribute_rec		This is a record type which contains all the descriptive flexfields: One descriptive flexfield structure defining column and 15 segments. Default: None Validate: None Error: None
p_global_attribute_rec	IN	global_attribute_rec_type		This is a record type which contains all the global descriptive flexfields: One global descriptive flexfield structure defining column and 20 segments. Default: None Validate: None Error: None
p_cancel_claims_flag	IN	VARCHAR2(1)		Not used. Leave null.
p_called_from	IN	VARCHAR2(20)		This parameter is used to identify the calling routine. Default: None Validate: None Error: None

Validation

This section explains the validation mechanisms for the various parameters of this API which are relatively more complex in nature and could not be explained in the Description column of the preceding table.

Cash Receipt ID

We have to validate whether this is a valid cash receipt ID, and whether we can reverse this receipt.

The validation steps are:

- This is a valid value in the database. For an invalid value, the error message AR_RAPI_CASH_RCPT_ID_INVALID is raised.
- Status should not be 'Reversed' for this receipt because you cannot reverse an already reversed receipt. The error message raised for an invalid value is AR_RAPI_CASH_RCPT_ID_INVALID.

The receipt is not standard reversible if any two of the following conditions are true:

- If a chargeback was created against an invoice that is applied to the payment to be reversed.
- If there are any payments, adjustments, credit memos, or chargebacks against the above chargeback records in the AR_PAYMENT_SCHEDULES table.
- If the above chargeback has already been posted to the general ledger.

The AR_RAPI_NON_REVERSIBLE error message is raised for invalid values. In these cases, you can create a debit memo reversal to reverse the receipt. Since the Receipt API does not currently support debit memo reversals, you can manually create them using the Receipts workbench.

Example

Objective:

To reverse a cash receipt using a call to the API *Ar_receipt_api_pub.Reverse* and passing a minimum number of input parameters.

This table lists the entered parameters:

Parameter	Entered Value	Default Value
p_api_version	1.0	

Parameter	Entered Value	Default Value
p_receipt_ number	'aj_test_cr_7'	
p_reversal_ category_code	'NSF'	
p_reversal_ reason_code	'PAYMENT REVERSAL'	

This table lists the defaulted input parameters, which were not entered:

Parameter	Entered Value	Default Value
p_cr_id		20340
p_reversal_date		01-JUN-2000
p_reversal_gl_ date		01-JUN-2000

The retrieval and handling of the warnings and error messages, put on the message stack by the API during execution, are the same as described in *Defaulting* on page 7-24.

Ar_receipt_api_pub.activity_application

Description

Call this routine to do an activity application on a cash receipt. Such applications include Short Term Debit (STD) and Receipt Write-off applications.

This API routine has 4 output and 25 input parameters in total. Based on the type, the following is the breakdown of the parameters:

Input

Standard API parameters: 4

Application parameters: 22 + 1 (descriptive flexfield record type)

1 (global descriptive flexfield record type)

Output

Standard API parameters: 3

Application parameters: 1

The description of the seven standard API parameters is the same as given in *Description* on page 7-11.

The following table lists the descriptions of the activity application-related parameters of the API.

> **Note:** If required parameters are not passed in a call to this API, then the call will fail. However, depending on the business scenario, you will have to pass in values for other parameters to successfully create the business object. Otherwise, error messages will be reported.

Parameter	Type	Data-type	Required*	Description
p_cr_id	IN	NUMBER(15)		The cash_receipt_id of the receipt which is to be used for the activity application. Default: None Validate: 1. Type must be 'CASH' 2. Status must not be Reversed or Approved 3. The receipt must not be Unidentified Error: AR_RAPI_CASH_RCPT_ID_INVALID AR_RAPI_CASH_RCPT_ID_NULL
p_receipt_number	IN	VARCHAR2(30)		The receipt number of the receipt to be applied. Used to default the cash_receipt_id. Default: None Validate: None Error: AR_RAPI_RCPT_NUM_INVALID
p_amount_applied	IN	NUMBER		The amount on the cash receipt that is to be applied against the specified activity. Default: Amount due remaining on the receipt. Validate: 1. Greater than or equal to 0. 2. Less than or equal to the amount due remaining on the receipt. 3. If a receipt write-off, then must fall within user and system limits (limits must be set). Error: 1.AR_RAPI_APPLIED_AMT_NULL 2.AR_RW_APP_NEG_UNAPP 3.AR_RW_AMOUNT_LESS_THAN_APP 4.AR_WR_NO_LIMIT 5.AR_WR_USER_LIMIT 6.AR_SYSTEM_WR_NO_LIMIT_SET 7.AR_WR_TOTAL_EXCEED_MAX_AMOUNT

Parameter	Type	Data-type	Required*	Description
p_applied_payment_ schedule_id	IN	NUMBER(15)	Yes	The payment schedule identifier here corresponds to special seeded values, such as -2. Default: Validate: The value should correspond to the special seeded values, such as: -2 (Short Term Debt). Error: AR_RAPI_APP_PS_ID_INVALID
p_link_to_customer_trx_ id	IN	NUMBER(15)		The customer_trx_id of the Bill for which the activity (e.g. Short Term Debt) application is being done. Default: Validate: The customer_trx_id should correspond to that of a Bill which has a current status of FACTORED or MATURED_PEND_RISK_ ELIMINATION. Error: AR_RAPI_LK_CUS_TRX_ID_INVALID
p_receivables_trx_id	IN	NUMBER(15)		Identifier of the receivables activity. Default: None Validate: 1.Valid database value. 2. The activity_type for the receivables_ trx_id should be in sync with the applied payment schedule identifier passed in. Error: 1.AR_RAPI_REC_TRX_ID_INVALID 2.AR_RAPI_ACTIVITY_X_INVALID
p_apply_date	IN	DATE		Date the application was applied. Default: 1. Receipt date, if receipt date >= system date. 2.System date, if receipt date < system date. Validate: apply date >= receipt date Error: AR_APPLY_BEFORE_RECEIPT

Parameter	Type	Data-type	Required*	Description
p_apply_gl_date	IN	DATE		Date that this application will be posted to the General Ledger. Default: Defaulted to greater of the receipt date and the system date. Validate: 1. Validated as per standard GL date validation described for the GL date in Create_cash routine. 2. >= receipt GL date Error: 1.AR_INVALID_APP_GL_DATE 2.AR_RW_GL_DATE_BEFORE_REC_GL
p_ussgl_transaction_code	IN	VARCHAR2(30)		Code defined by public sector accounting. Default: None Validate: None Error: None
p_attribute_rec	IN	attribute_rec_type		This is a record type which contains all 15 descriptive flexfield segments and one descriptive flexfield structure defining column. It represents the Receipt Application Information flexfield. Default: DFF APIs used to do the defaulting and validation Validate: DFF APIs used to do the defaulting and validation Error: AR_RAPI_DESC_FLEX_INVALID
p_global_attribute_rec	IN	global_attribute_rec_type		This is a record type which contains all the global descriptive flexfields: one global descriptive flexfield structure defining column and 20 segments. Default: None Validate: None Error: None
p_comments	IN	VARCHAR2 (240)		User's comments for the activity application.
p_application_ref_type	IN	VARCHAR2(30)		Not used. Leave null.
p_application_ref_id	IN	NUMBER(15)		Not used. Leave null.

Parameter	Type	Data-type	Required*	Description
p_application_ref_num	IN	VARCHAR2(30)		If resulting from a settlement of a claim, then this will contain the deduction number.
p_secondary_ application_ref_id	IN	NUMBER(15)		If resulting from a settlement of a claim, then this will contain the claim ID.
p_payment_set_id	IN	NUMBER(15)		Payment set ID is populated only when doing a prepayment activity application on a prepayment receipt. Default: None Validate: None
p_receivable_ application_id	OUT	NUMBER(15)		The ID of the resulting activity receivable application.
p_customer_reference	IN	VARCHAR2 (100)		Customer supplied reference.
p_val_writeoff_limits_ flag	IN	VARCHAR2(1)		Flag to indicate whether user-level write-off limits should apply. Default: Y Validate: None Error: None
p_called_from	IN	VARCHAR2(20)		This parameter is used to identify the calling routine. Default: None Validate: None Error: None
p_netted_receipt_flag	IN	VARCHAR2(1)		Used for payment netting. Leave null.
p_netted_cash_receipt_id	IN	NUMBER(15)		Used for payment netting. Leave null.
p_secondary_app_ref_ type	IN	VARCHAR2(30)		Used for automated receipt handling. Leave null.
p_secondary_app_ref_ num	IN	VARCHAR2(30)		Used for automated receipt handling. Leave null.

Example

Objective:

To apply a cash receipt in then functional currency to a receipt write-off activity in the functional currency, using a call to the API *Ar_receipt_api_pub.activity_application* and passing a minimum number of input parameters.

This table lists the entered parameters:

Parameter	Entered Value	Default Value
p_api_version	1.0	
p_receipt_number	'aj_test_cr_2'	
p_receivables_trx_id	1300	
p_applied_payment_schedule_id	-3	

This table lists the defaulted input parameters, which were not entered:

Parameter	Entered Value	Default Value
p_cr_id		23927
p_gl_date		01-JUN-2000
p_apply_date		01-JUN-2000
p_amount_applied		100

The retrieval and handling of the warnings and error messages, put on the message stack by the API during execution, are the same as described in *Defaulting* on page 7-24.

Ar_receipt_api_pub.activity_unapplication

Description

Call this routine to do a reversal of an activity application on a cash receipt. Such applications include Short Term Debt and Receipt write-off.

This API routine has 3 output and 9 input parameters in total. Based on the type, the following is the breakdown of the parameters:

Input

Standard API parameters: 4

Application parameters: 5

Output

Standard API parameters: 3

Application parameters: 0

The description of the seven standard API parameters is the same as given in *Description* on page 7-11.

The following table lists the descriptions of the activity unapplication-related parameters of the API.

Note: If required parameters are not passed in a call to this API, then the call will fail. However, depending on the business scenario, you will have to pass in values for other parameters to successfully create the business object. Otherwise, error messages will be reported.

Parameter	Type	Data-type	Required	Description
p_cr_id	IN	NUMBER(15)		The cash_receipt_id of the receipt on which the activity application needs to be reversed. Default: None Validate: 1. Type must be 'CASH' 2. Status must not be Reversed or Approved 3. The receipt must not be Unidentified Error: AR_RAPI_CASH_RCPT_ID_INVALID AR_RAPI_CASH_RCPT_ID_NULL
p_receipt_number	IN	VARCHAR2(30)		The receipt number of the receipt to be reversed. Used to default the cash_receipt_id. Default: None Validate: None Error: AR_RAPI_RCPT_NUM_INVALID
p_receivable_application_id	IN	NUMBER(15)		Identifies the receivable application. Used to derive the customer trx id, cash_receipt_id and the applied_ps_id if not specified. Default: Refer to *Validation* on page 7-73. Validate: 1. Application type = 'CASH'. 2. Display flag = 'Y' (latest application) and status = 'ACTIVITY'. 3. The applied payment schedule id of the receivable application record must correspond to the p_applied_payment_schedule_id, if specified. 4. Must correspond to the cash receipt id specified. Error: AR_RAPI_REC_APP_ID_INVALID

Parameter	Type	Data-type	Required	Description
p_reversal_gl_date	IN	DATE		The reversal GL date. Default: Gets defaulted to the application GL date if it is a valid gl_date. Validate: 1. It is valid if the following conditions are true: --The date is in an Open or Future period --The period cannot be an Adjustment period 2. Reversal GL date >= application GL date 3. Reversal GL date >= receipt GL date If the date is invalid, then: ■ If the most recent open period is prior to the receipt date: last date of that period ■ If there is a period open after the receipt date: first date of the last open period Error: AR_INVALID_APP_GL_DATE AR_RW_BEFORE_APP_GL_DATE AR_RW_BEFORE_RECEIPT_GL_DATE
p_called_from	IN	VARCHAR2(20)	Yes	Indicates which program is calling this API. For example, the BR_REMIT program would be calling this routine for short term debt applications. Default: None Validate: None Error: None

Example

Objective:

To unapply an activity application, using a call to the API *Ar_receipt_api_pub.activity_unapplication* and passing minimum number of input parameters.

This table lists the entered parameters:

Parameter	Entered Value	Default Value
p_api_version	1.0	

Parameter	Entered Value	Default Value
p_receivable_ application_id	10051	
p_called_from	NULL	

This table lists the defaulted input parameters, which were not entered:

Parameter	Entered Value	Default Value
p_cr_id		20338
p_reversal_gl_date		01-JUN-2000

The retrieval and handling of the warnings and error messages, put on the message stack by the API during execution, are the same as described in *Defaulting* on page 7-24.

Ar_receipt_api_pub.Create_misc

Description

Call this routine to create a miscellaneous receipt.

> **Note:** This routine does ***not*** call Oracle iPayment directly. See *Integration with Oracle iPayment* on page 7-10.

This API routine has 4 output and 36 input parameters in total. Based on the type, the following is the breakdown of the parameters:

Input

Standard API parameters: 4

Application parameters: 32

Output

Standard API parameters: 3

Application parameters: 1

The following table lists the standard API parameters, which are common to all the routines in the Receipt API:

Parameter	Type	Data-type	Required	Default Value	Description
p_api_version	IN	NUMBER	Yes		Used to compare version numbers of incoming calls to its current version number. Unexpected error is raised if version incompatibility exists. In the current version of the API, you should pass in a value of 1.0 for this parameter.
p_init_msg_list	IN	VARCHAR2		FND_API.G_FALSE	Allows API callers to request that the API does initialization of the message list on their behalf.

Parameter	Type	Data-type	Required	Default Value	Description
p_commit	IN	VARCHAR2		FND_API.G_FALSE	Used by API callers to ask the API to commit on their behalf.
p_validation_level	IN	NUMBER		FND_API.G_VALID_ LEVEL_FULL	Not to be used currently as this is a public API.
x_return_status	OUT	VARCHAR2			Represents the API overall return status. Detailed in *Return Status* on page 7-8.
x_msg_count	OUT	NUMBER			Number of messages in the API message list.
x_msg_data	OUT	VARCHAR2			This is the message in encoded format if x_msg_ count=1.

The following table lists the parameters that are relevant to the miscellaneous receipt:

Parameter	Type	Data-type	Required*	Description
p_usr_currency_code	IN	VARCHAR2		The translated currency code. Used to derive the p_currency_code if it is not entered. Default: None Validate: Should be a valid currency, so that the corresponding currency code can be derived. Error: AR_RAPI_USR_CURR_CODE_ INVALID

Parameter	Type	Data-type	Required*	Description
p_currency_code	IN	VARCHAR2		The actual currency code that gets stored in AR tables. Default: 1. Derived from p_usr_currency_code if entered, else 2. Defaults to the functional currency code Validate: 1. Validated against the currencies in fnd_currencies table. Error: AR_RAPI_CURR_CODE_INVALID Warning: AR_RAPI_FUNC_CURR_DEFAULTED
p_usr_exchange_rate_type	IN	VARCHAR2		The translated exchange rate type. Used to derive the p_exchange_rate_type if it has not been entered. Default: None Validate: Should be a valid rate type. Error: AR_RAPI_USR_X_RATE_TYP_INVALID
p_exchange_rate_type	IN	VARCHAR2		Exchange rate type stored in AR tables. Default: 1. In case of foreign currency receipt, derived from p_usr_exchange_rate_type. 2. In case of foreign currency receipt, defaults from profile option AR: Default Exchange Rate Type Validate: 1. Validated against values in gl_daily_conversion_types table. Error: AR_RAPI_X_RATE_TYPE_INVALID

Parameter	Type	Data-type	Required*	Description
p_exchange_rate	IN	NUMBER		The exchange rate between the receipt currency and the functional currency. Default: 1. Derived from the Daily Rates table for rate_type <> 'User' in case of non-functional currency 2. If profile option Journals: Display Inverse Rate = 'Y', set user-entered value to 1/p_exchange_rate 3. The entered value is rounded to a precision of 38 Validate: 1. In case of non-functional currency, the rate should have a positive value for rate type = 'User' 2. For non-functional currency and type is <> 'User', do not specify any value Error: AR_RAPI_X_RATE_INVALID AR_RAPI_X_RATE_NULL
p_exchange_rate_date	IN	DATE		The date on which the exchange rate is valid. Default: Receipt date Validate: 1. For a non-functional currency and type is <> 'User', there should be a valid rate existing in the database for this date. This is a cross validation of type, currency, and date. Error: AR_NO_RATE_DATA_FOUND
p_amount	IN	NUMBER	Yes	The cash receipt amount. Default: Null Validate: > 0 Error: AR_RAPI_REC_AMT_NEGATIVE AR_RAPI_RCPT_AMOUNT_NULL

Parameter	Type	Data-type	Required*	Description
p_receipt_number	IN	VARCHAR2(30)		The receipt number of the receipt to be created. Default: If not specified, the receipt number is defaulted from the document sequence value. Validate: Receipt number should not be null Error: AR_RAPI_RCPT_NUM_NULL
p_receipt_date	IN	DATE		The receipt date of the entered cash receipt. Default: System date Validate: None Error: None
p_gl_date	IN	DATE		Date when this receipt will be posted to the general ledger. Default: Gets defaulted to the receipt date if it is a valid** gl_date, otherwise: --If the most recent open period is prior to the receipt date: last date of that period --If there is a period open after the receipt date: first date of the last open period Validate**: It is valid if the following conditions are true: --The date is in an Open or Future period --The period cannot be an Adjustment period Error: AR_INVALID_APP_GL_DATE

Parameter	Type	Data-type	Required*	Description
p_receivables_trx_id	IN	NUMBER(15)		Identifies the receivables activity. Default: If not specified, it is derived from p_activity. Validate: Validates it against the values in the ar_receivables_trx table --Type column having values: 'MISCCASH', 'BANK_ERROR', 'CCREFUND'. --Checks the receipt_date to be within start_date_active and end_date_active column values. --Status is Active or null. --Not null. Error: AR_RAPI_ACTIVITY_INVALID AR_RAPI_REC_TRX_ID_INVALID AR_RAPI_REC_TRX_ID_NULL
p_activity	IN	VARCHAR2(50)		Name of the receivables activity. This is used to derive the p_receivables_trx_id. Default: None Validate: None Error: None
p_misc_payment_source	IN	VARCHAR2(30)		Identifies the source of the miscellaneous receipt. Default: None Validate: None Error: None
p_tax_code	IN	VARCHAR2(50)		Depending on the sign of the amount entered, it is the asset tax code (for positive sign or zero) or the liability tax code (negative sign). This is used to derive the p_vat_tax_id. Default: None Validate: None Error: None

Parameter	Type	Data-type	Required*	Description
p_vat_tax_id	IN	NUMBER(15)		The VAT tax identifier for the current miscellaneous receipt. Default: --defaulted from p_tax_code --defaulted from receivables_trx_ id/activity Validate: 1. For 'Accrual' accounting method, the vat_tax_id is validated against the values in ar_vat_tax having --receipt_date between start_date_active and end_date_active column values --enabled_flag = 'Y' --tax_type should not be 'TAX_GROUP', 'LOCATION', 'SALES_TAX' --displayed_flag = 'Y' --The tax_class is 'O' (output) for positive or zero amount and 'I' (input) for negative amount --set of books should match the current set of books 2. For 'Cash basis' accounting method, the vat_tax_id should not be specified. Error: AR_RAPI_VAT_TAX_ID_INVALID AR_RAPI_TAX_CODE_INVALID
p_tax_rate	IN	NUMBER		The new tax rate specified when you override the rate for an ad-hoc tax code. Default: 1) Defaulted from the tax rate on the tax code (p_tax_code/p_vat_tax_id). 2) Defaulted from the p_tax_amount when the tax amount is specified for the ad-hoc tax code case. Validate: For 'Accrual' accounting method, tax rate can be specified only in case of an ad-hoc tax code (p_tax_code/p_vat_tax_id) and the profile option 'Tax: Allow Ad Hoc Tax Changes' set to Yes. For 'Cash basis' accounting method, the tax_rate should never be specified. Error: AR_RAPI_TAX_RATE_INVALID AR_RAPI_TAX_RATE_AMT_X_INVALID

Parameter	Type	Data-type	Required*	Description
p_tax_amount	IN	NUMBER		The tax amount specified in case where you override the rate for an ad-hoc tax code. It is used to derive the tax_rate. Default: None Validate: This needs to be specified only in case of an ad-hoc tax code (p_tax_code/p_vat_tax_id) and the profile option 'Tax: Allow Ad Hoc Tax Changes' set to Yes. For 'Cash basis' accounting method, the tax_amount should never be specified Error: AR_RAPI_TAX_RATE_AMT_X_INVALID
p_deposit_date	IN	DATE		The deposit date. Default: Receipt date Validate: None Error: None
p_reference_type	IN	VARCHAR2(30)		Indicates whether this miscellaneous receipt is a 'PAYMENT', 'RECEIPT', 'PAYMENT_BATCH' or 'REMITTANCE'. Default: None Validate: --Check it for the specified valid values. --Should not have a null value if either p_reference_id or p_reference_num is specified. Error: AR_RAPI_REF_TYPE_INVALID AR_RAPI_REF_TYPE_NULL
p_reference_id	IN	NUMBER(15)		A foreign key to AR_BATCHES, AR_CASH_RECEIPTS, AP_INVOICE_SELECTION_CRITERIA or AP_CHECKS, depending on the specified value of p_reference_type. Default: None Validate: Detailed in *Validation* on page 7-23. Error: AR_RAPI_REF_NUM_INVALID AR_RAPI_REF_ID_INVALID

Parameter	Type	Data-type	Required*	Description
p_reference_num	IN	VARCHAR2(30)		The reference number. It is used for deriving the p_reference_id. Default: None Validate: None Error: None
p_remittance_bank_account_id	IN	NUMBER(15)		Identifies the user's bank account for depositing the receipt. Default: 1. From remittance bank account number 2. From the receipt method based on logic mentioned in *Defaulting* on page 7-24. Validate: In addition to the validation logic detailed in *Validation* on page 7-23, those receipt methods which have notes_receivable = 'Y' or bill_of_exchange_flag = 'Y' on the receipt class are excluded for miscellaneous receipts. Error: AR_RAPI_REM_BK_AC_ID_INVALID AR_RAPI_REM_BK_AC_ID_NULL
p_remittance_bank_account_num	IN	VARCHAR2(30)		The remittance bank account number. Used to default the remittance bank account id if not specified. Default: None Validate: None Error: AR_RAPI_REM_BK_AC_NUM_INVALID
p_remittance_bank_account_name	IN	VARCHAR2(50)		The remittance bank account name. Used to default the remittance bank account id if not specified. Default: None Validate: None Error: AR_RAPI_REM_BK_AC_NAME_INVALID

Parameter	Type	Data-type	Required*	Description
p_ussgl_transaction_code	IN	VARCHAR2(30)		Code defined by public sector accounting. Default: None Validate: None Error: None
p_receipt_method_id	IN	NUMBER(15)		Identifies the payment method of the receipt. Default: From receipt method name Validate: In addition to the validation logic detailed in *Validation* on page 7-23, those receipt methods which have notes_receivable = 'Y' or bill_of_exchange_flag = 'Y' on the receipt class are excluded for the miscellaneous receipts. Error: AR_RAPI_INVALID_RCT_MD_ID
p_receipt_method_name	IN	VARCHAR2(30)		The payment method name of the receipt. Used to default the receipt method id if not specified Default: None Validate: None Error: AR_RAPI_RCPT_MD_NAME_INVALID
p_doc_sequence_value	IN	NUMBER		Value assigned to document receipt. Default: Detailed in *Defaulting* on page 7-24. Validate: --User should not pass in the value if the current document sequence is automatic --Document sequence value should not be entered if profile option Sequential Numbering is set to Not Used Error: AR_RAPI_DOC_SEQ_AUTOMATIC AR_RAPI_DOC_SEQ_VAL_INVALID

Parameter	Type	Data-type	Required*	Description
p_anticipated_clearing_date	IN	DATE		Date the receipt is expected to be cleared. Default: None Validate: greater than or equal to gl_date Error: AR_RW_EFFECTIVE_BEFORE_GL_DATE
p_attribute_rec	IN	attribute_rec_type		This is a record type which contains all 15 descriptive flexfield segments and one descriptive flexfield structure defining column. It represents the Receipt Information flexfield. Default: DFF APIs used to do the defaulting and validation Validate: DFF APIs used to do the defaulting and validation Error: AR_RAPI_DESC_FLEX_INVALID
p_global_attribute_rec	IN	global_attribute_rec_type		This is a record type which contains all 20 global descriptive flexfield segments and one global descriptive flexfield structure defining column. Default: None Validate: None Error:
p_comments	IN	VARCHAR2(240)		User's comments.
p_misc_receipt_id	OUT	NUMBER(15)	Yes	The cash_receipt_id of the receipt created by the API call.
p_called_from	IN	VARCHAR2 (20)		This parameter is used to identify the calling routine. Default: None Validate: None Error: None

Ar_receipt_api_pub.apply_other_account

Description

Call this routine to do an "other" account application on a cash receipt. Typically this would be to create a claim investigation application with a noninvoice-related deduction or overpayment in Trade Management (if installed).

This API routine has 4 output and 26 input parameters in total. Based on the type, the following is the breakdown of the parameters:

Input

Standard API parameters: 4

Application parameters: 18 + 1 (descriptive flexfield record type)
1 (global descriptive flexfield record type)

Output

Standard API parameters: 3

Application parameters: 1

The description of the seven standard API parameters is the same as given in *Description* on page 7-11.

The following table lists the descriptions of the other account application-related parameters of the API:

Parameter	Type	Data-type	Required	Description
p_cr_id	IN	NUMBER (15)		The cash_receipt_id of the receipt which is to be applied to the "other" account. Default: None Validate: 1. Type must be 'CASH'. 2. Status must not be Reversed or Approved. 3. The receipt must not be Unidentified. Error: AR_RAPI_CASH_RCPT_ID_INVALID AR_RAPI_CASH_RCPT_ID_NULL

Parameter	Type	Data-type	Required	Description
p_receipt_number	IN	VARCHAR2 (30)		The receipt number of the receipt to be applied to the "other" account. Used to default the cash_receipt_id. Default: None Validate: None Error : AR_RAPI_RCPT_NUM_INVALID
p_amount_applied	IN	NUMBER		The amount on the cash receipt that is to be applied to the "other" account. Default: Amount due remaining on the receipt. Validate: Less than or equal to the amount due remaining on the receipt. Error: 1. AR_RAPI_APPLIED_AMT_NULL 2. AR_RW_AMOUNT_LESS_THAN_APP
p_applied_payment_schedule_id	IN	NUMBER (15)	Yes	This payment schedule identifier corresponds to special seeded values, such as –4 (for Claim Investigation). Default: Validate: The value should correspond to the special seeded values, such as -4 (Claim Investigation). Error: AR_RAPI_APP_PS_ID_INVALID
p_receivables_trx_id	IN	NUMBER (15)		Identifier of receivables activity. Default: None Validate: 1. Valid database value. 2. The activity_type for the receivables_trx_id should be in sync with the provided applied payment schedule identifier. Error : 1. AR_RAPI_REC_TRX_ID_INVALID 2. AR_RAPI_ACTIVITY_X_INVALID

Parameter	Type	Data-type	Required	Description
p_apply_date	IN	DATE		Date the application was applied. Default: 1. Receipt date, if receipt date >= system date. 2. System date, if receipt date < system date. Validate: apply date >= receipt date Error: AR_APPLY_BEFORE_RECEIPT
p_apply_gl_date	IN	DATE		Date when this application will be posted to the General Ledger. Default: Defaulted to greater of the receipt date and the system date. Validate: 1. Validated as per standard gl date validation described for the gl date in the Create_cash routine. 2. >= receipt gl date Error: 1. AR_INVALID_APP_GL_DATE 2. AR_RW_GL_DATE_BEFORE_REC_GL
p_ussgl_transaction_code	IN	VARCHAR2 (30)		Code defined by public sector accounting. Default: None Validate: None Error: None
p_attribute_rec	IN	attribute_ rec_type		This is a record type which contains all the 15 descriptive flexfield segments and one descriptive flexfield structure defining column. It represents the Receipt Application Information flexfield. Default: DFF APIs used to do the defaulting and validation. Validate: DFF APIs used to do the defaulting and validation. Error : AR_RAPI_DESC_FLEX_INVALID

Parameter	Type	Data-type	Required	Description
p_global_attribute_rec	IN	global_ attribute_ rec_type		This is a record type which contains all the global descriptive flexfields: one global descriptive flexfield structure defining column and 20 segments. Default: None Validate: None Error: None
p_comments	IN	VARCHAR2 (240)		User's comments for the other account application.
p_application_ref_type	IN	VARCHAR2 (30)	Yes	Defines the context of the application reference columns. For Trade Management, the value should be 'CLAIM'. Default: None Validate: Must be 'CLAIM' if a Trade Management deduction is being created (Trade Management must be installed). Error: AR_RAPI_INVALID_APP_REF
p_application_ref_id	IN	NUMBER (15)		Not used. Leave null.
p_application_ref_num	IN	VARCHAR2 (30)		The reference number relating to the application reference type. If application reference type is 'CLAIM', then this would be a deduction number. Default: None Validate: If populated, then must be an existing deduction number in Trade Management. Error: AR_RAPI_INVALID_CLAIM_ NUM

Parameter	Type	Data-type	Required	Description
p_secondary_application_ref_id	IN	NUMBER (15)		The secondary application reference ID related to the application reference type. Default: None Validate: If populated, and if application reference type is 'CLAIM', then this must contain a valid claim ID in Trade Management. Error: AR_RW_INVALID_CLAIM_ID
p_payment_set_id	IN	NUMBER (15)		Payment set ID is populated only for a prepayment receipt which is to be applied to the "other" account. Default: None Validate: None
p_receivable_application_id	OUT	NUMBER (15)		The ID of the resulting activity receivable application.
p_application_ref_reason	IN	VARCHAR2 (30)		The reason code related to the application reference type. Default: None Validate: If populated, and if application reference type is 'CLAIM', then this must contain a valid reason code ID from Trade Management. Error: AR_RAPI_INVALID_REF_REASON
p_customer_reference	IN	VARCHAR2 (100)		Customer supplied reference.
p_customer_reason	IN	VARCHAR2 (30)		Reason code supplied by customer, in the context of an application reference type of 'CLAIM'. Default: None Validate: None in Oracle Receivables (the attempt to match to an Oracle reason code is made in Trade Management).

Parameter	Type	Data-type	Required	Description
p_called_from	IN	VARCHAR2 (20)		This parameter is used to identify the calling routine. Default: None Validate: None Error: None

Example

Objective:

To apply a cash receipt in functional currency to Claim Investigation, and to create a non-invoice overpayment in the functional currency using a call to the API *Ar_receipt_api_pub.apply_other_account* and passing a minimum number of input parameters.

This table lists the entered parameters:

Parameter	Entered Value	Default Value
p_api_version	1.0	
p_receipt_number	'aj_test_cr_2'	
p_receivables_trx_id	1400	
p_application_ref_type	'CLAIM'	
p_applied_payment_schedule_id	-4	

This table lists the defaulted input parameters, which were not entered:

Parameter	Entered Value	Default Value
p_cr_id		23927
p_gl_date		01-JUN-2000
p_apply_date		01-JUN-2000
p_amount_applied		100

The retrieval and handling of the warnings and error messages, put on the message stack by the API during execution, are the same as described in *Defaulting* on page 7-24.

Ar_receipt_api_pub.unapply_other_account

Description

Call this routine to do a reversal of an "other" account application on a cash receipt.

This API routine has 3 output and 9 input parameters in total. Based on the type, the following is the breakdown of the parameters:

Input

Standard API parameters: 4

Application parameters: 6

Output

Standard API parameters: 3

Application parameters: 0

The description of the seven standard API parameters is the same as given in *Description* on page 7-11.

The following table lists the descriptions of the other account unapplication-related parameters of the API:

Parameter	Type	Data-type	Required	Description
p_cr_id	IN	NUMBER (15)		The cash_receipt_id of the receipt which is to be applied to the "other" account. Default: None Validate: 1. Type must be 'CASH'. 2. Status must not be Reversed or Approved. 3. The receipt must not be Unidentified. Error: 1. AR_RAPI_CASH_RCPT_ID_INVALID 2. AR_RAPI_CASH_RCPT_ID_NULL

Parameter	Type	Data-type	Required	Description
p_receipt_number	IN	VARCHAR2 (30)		The receipt number of the receipt from which the "other" account application is to be unapplied. Used to default the cash_ receipt_id. Default: None Validate: None Error: AR_RAPI_RCPT_NUM_INVALID
p_receivable_application_id	IN	NUMBER (15)		Identifies the receivable application. Used to derive the customer trx id, cash_ receipt_id, and the applied_ps_id, if not specified. Default: Refer to *Validation* on page 7-73. Validate: 1. Application type = 'CASH'. 2. Display flag = 'Y' (latest application) and status = 'OTHER ACC'. 3. The applied payment schedule id of the receivable application record must correspond to the p_applied_payment_ schedules_id, if specified. 4. The cash receipt id must correspond to the cash receipt id specified. Error: AR_RAPI_REC_APP_ID_INVALID

Parameter	Type	Data-type	Required	Description
p_reversal_gl_date	IN	DATE		The reversal gl date. Default: Gets defaulted to the application gl date if it is a valid gl_date. Validate: 1. It is valid if the following conditions are true: --The date is in an Open or Future period. --The period cannot be an Adjustment period. 2. The reversal GL date >= application GL date. 3. The reversal GL date >= receipt GL date. If the date is invalid, then: ■ If the most recent open period is prior to the receipt date: last date of that period ■ If there is a period open after the receipt date: first date of the last open period Error: AR_INVALID_APP_GL_DATE AR_RW_BEFORE_APP_GL_DATE AR_RW_BEFORE_RECEIPT_GL_DATE
p_called_from	IN	VARCHAR2 (20)		Indicates which program is calling this API. Default: None Validate: None Error: None
p_cancel_claim_flag	IN	VARCHAR2 (1)		Not used. Leave null.

Example

Objective:

To unapply an "other" account application using the call to API *Ar_receipt_api_pub.unapply_other_account* and passing a minimum number of input parameters.

This table lists the entered parameters:

Parameter	Entered Value	Default Value
p_api_version	1.0	
p_receivable_ application_id	10053	

This table lists the defaulted input parameters, which were not entered:

Parameter	Entered Value	Default Value
p_cr_id		20338
p_reversal_gl_date		01-JUN-2000

The retrieval and handling of the warnings and error messages, put on the message stack by the API during execution, are the same as described in *Defaulting* on page 7-24.

Ar_receipt_api_pub.apply_open_receipt

Description

Call this routine to apply a cash receipt to another open receipt. Open receipts include unapplied cash, on-account cash, and claim investigation applications. Claim investigation applications can be applied only if Trade Management is installed.

This API routine has 8 output and 18 input parameters in total. Based on the type, the following is the breakdown of the parameters:

Input

Standard API parameters: 4

Application parameters: 12 +
2 (descriptive and global descriptive flexfield record type)

Output

Standard API parameters: 3

Application parameters: 5

The description of the seven standard API parameters is the same as given in *Description* on page 7-11.

The following table lists the descriptions of the apply open receipt-related parameters of the API:

Parameter	Type	Data-type	Required	Description
p_cr_id	IN	NUMBER (15)		The cash_receipt_id of the receipt which is to be applied to an open receipt. Default: None Validate: 1. Type must be 'CASH'. 2. Status must not be Reversed or Approved. 3. The receipt must not be Unidentified. 4. The receipt being applied and the open receipt must have the same currency. Error: AR_RAPI_CASH_RCPT_ID_INVALID AR_RAPI_CASH_RCPT_ID_NULL AR_RW_NET_DIFF_RCT_CURR
p_receipt_number	IN	VARCHAR2 (30)		The receipt number of the receipt to be applied to an open receipt. Used to default the cash_receipt_id. The receipt being applied and the open receipt must have the same currency. Default: None Validate: None Error: AR_RAPI_RCPT_NUM_INVALID AR_RW_NET_DIFF_RCT_CURR
p_applied_payment_schedule_id	IN	NUMBER (15)		Not used. Leave null.
p_open_cash_receipt_id	IN	NUMBER (15)		The cash_receipt_id of the open receipt which is to be applied to. Default: None Validate: 1. Type must be 'CASH'. 2. Status must not be Reversed or Approved. 3. The receipt must not be Unidentified. 4. The receipt being applied and the open receipt must have the same currency. Error: AR_RAPI_CASH_RCPT_ID_INVALID AR_RAPI_CASH_RCPT_ID_NULL AR_RW_NET_DIFF_RCT_CURR

Parameter	Type	Data-type	Required	Description
p_open_receipt_number	IN	VARCHAR2 (30)		The receipt number of the open receipt. Used to default the open cash_receipt_id. The receipt being applied and the open receipt must have the same currency. Default: None Validate: None Error: AR_RAPI_RCPT_NUM_INVALID AR_RW_NET_DIFF_RCT_CURR
p_open_rec_app_id	IN	NUMBER (15)		The ID of the receivable application of the open receipt, if on-account or claim investigation. Default: None Validate: Must have status of ACC or OTHER ACC, and display must be 'Y'. Errors: AR_RAPI_REC_APP_ID_INVALID AR_RW_NET_OPEN_RCT_ONLY
p_amount_applied	IN	NUMBER (15)		The amount on the cash receipt that is to be applied to an open receipt. Default: None Validate: 1. Must be a natural application, i.e. it must move the balance on the open receipt closer to zero. Error: 1. AR_RAPI_APPLIED_AMT_NULL 2. AR_RW_AMOUNT_LESS_THAN_APP 3. AR_RW_NET_OPEN_AMT_INC
p_apply_date	IN	DATE		Date the application was applied. Default: 1. Receipt date, if receipt date >= system date. 2. System date, if receipt date < system date. Validate: apply date >= receipt date. Error: AR_APPLY_BEFORE_RECEIPT

Parameter	Type	Data-type	Required	Description
p_apply_gl_date	IN	DATE		Date when this application will be posted to the General Ledger. Default: Defaulted to greater of the receipt GL date, the open receipt GL date, and the system date. Validate: 1. Validated as per standard gl date validation described for the gl date in the Create_cash routine. 2. >= receipt gl date. Error: 1. AR_INVALID_APP_GL_DATE 2. AR_RW_GL_DATE_BEFORE_REC_GL 3. AR_RW_GL_DATE_BEFORE_OPEN_REC
p_ussgl_transaction_code	IN	VARCHAR2 (30)		Code defined by public sector accounting. Default: None Validate: None Error: None
p_attribute_rec	IN	attribute_ rec_type		This is a record type which contains all the 15 descriptive flexfield segments and one descriptive flexfield structure defining column. It represents the Receipt Application Information flexfield. Default: DFF APIs used to do the defaulting and validation. Validate: DFF APIs used to do the defaulting and validation. Error: AR_RAPI_DESC_FLEX_INVALID
p_global_attribute_rec	IN	global_ attribute_ rec_type		This is a record type which contains all the global descriptive flexfields: One global descriptive flexfield structure defining column and 20 segments. Default: None Validate: None Error: None
p_comments	IN	VARCHAR2 (240)		User's comments for the other account application.

Parameter	Type	Data-type	Required	Description
x_application_ref_num	OUT	VARCHAR2 (30)		The reference number from the open receipt application, if applicable. If the application reference type is 'CLAIM', then this would be a deduction number.
x_receivable_application_id	OUT	NUMBER (15)		The ID of the resulting payment netting receivable application.
x_applied_rec_app_id	OUT	NUMBER (15)		The ID of the corresponding payment netting receivable application created on the applied-to receipt.
x_acctd_amount_applied_from	OUT	NUMBER (15)		Amount applied from the receipt, in functional currency and converted using the main receipt's exchange rate.
x_acctd_amount_applied_to	OUT	VARCHAR2 (30)		Amount applied to the open receipt, in functional currency and converted using the open receipt's exchange rate. Used in conjunction with x_applied_amount_ applied_from to determine exchange gain/loss.
p_called_from	IN	VARCHAR2 (20)		This parameter is used to identify the calling routine. Default: None Validate: None Error: None

Example

Objective:

To apply a cash receipt in your functional currency to unapplied cash on another receipt, using a call to the API *Ar_receipt_api_pub.apply_open_receipt* and passing a minimum number of input parameters.

This table lists the entered parameters:

Parameter	Entered Value	Default Value
p_api_version	1.0	
p_receipt_number	'aj_test_cr_10'	

Parameter	Entered Value	Default Value
p_open_receipt_number	'aj_test_cr_30'	
p_amount_applied	-200	

This table lists the defaulted input parameters, which were not entered:

Parameter	Entered Value	Default Value
p_cr_id		23935
p_open_cash_receipt_id		23973
p_gl_date		01-JUN-2000
p_apply_date		01-JUN-2000

The retrieval and handling of the warnings and error messages, put on the message stack by the API during execution, are the same as described in *Defaulting* on page 7-24.

Ar_receipt_api_pub.unapply_open_receipt

Description

Call this routine to reverse a payment netting application on a cash receipt.

This API routine has 3 output and 7 input parameters in total. Based on the type, the following is the breakdown of the parameters:

Input

Standard API parameters: 4

Application parameters: 3

Output

Standard API parameters: 3

Application parameters: 0

The description of the seven standard API parameters is the same as given in *Description* on page 7-11.

The following table lists the descriptions of the unapply open receipt-related parameters of the API:

Parameter	Type	Data-type	Required	Description
p_receivable_application_id	IN	NUMBER(15)		Identifies the receivable application to be unapplied. Default: Refer to *Validation* on page 7-73. Validate: 1. Application type = 'CASH'. 2. Display flag = 'Y' (latest application) and status = 'ACTIVITY', receivables_trx_id = -163. 3. Unapplying this application must not result in either receipt becoming negative. Error: AR_RAPI_REC_APP_ID_INVALID AR_RW_NET_UNAPP_OVERAPP

Parameter	Type	Data-type	Required	Description
p_reversal_gl_date	IN	DATE		The reversal gl date. Default: Gets defaulted to the application gl date if it is a valid gl_date. Validate: 1. It is valid if the following conditions are true: --The date is in an Open or Future period. --The period cannot be an Adjustment period. 2. The reversal GL date >= application GL date. 3. The reversal GL date >= receipt GL date. If the date is invalid, then: ■ If the most recent open period is prior to the receipt date: last date of that period ■ If there is a period open after the receipt date: first date of the last open period Error: AR_INVALID_APP_GL_DATE AR_RW_BEFORE_APP_GL_DATE AR_RW_BEFORE_RECEIPT_GL_DATE
p_called_from	IN	VARCHAR2(20)	Yes	Indicates which program is calling this API. Default: None Validate: None Error: None

Example

Objective:

To unapply an open receipt/payment netting application using the call to API *Ar_receipt_api_pub.unapply_open_receipt* and passing a minimum number of input parameters.

This table lists the entered parameters:

Parameter	Entered Value	Default Value
p_api_version	1.0	
p_receivable_ application_id	10055	

This table lists the defaulted input parameters, which were not entered:

Parameter	Entered Value	Default Value
p_cr_id		20340
p_reversal_gl_date		01-JUN-2000

The retrieval and handling of the warnings and error messages, put on the message stack by the API during execution, are the same as described in *Defaulting* on page 7-24.

Ar_receipt_api_pub.Create_apply_on_acc

Description

This routine is called to create a cash receipt and place it on account. Use this routine when no specific debit item is referenced for receipt application, but you do not want to leave the cash as an unapplied liability.

This is essentially a superset of *Ar_receipt_api_pub.Create_cash* and *Ar_receipt_api_pub.Apply_on_account* APIs, and contains the same parameters as contained in those two APIs. During the call to this API, if the receipt is successfully created but its on-account application fails, then the receipt creation is also rolled back.

This routine calls Oracle *i*Payment, where required. See *Integration with Oracle iPayment* on page 7-10.

> **Note:** To create credit card receipts that need to be processed by *i*Payment APIs, you must pass the p_call_payment_processor parameter as fnd_api.g_true. Additionally, you must specify the p_customer_bank_account_id parameter.

This API routine has 4 output and 57 input parameters:

Input

Standard API parameters: 4

Application parameters: 49 + 2 (descriptive flexfield parameter)

+ 2 (global descriptive flexfield parameter)

Output

Standard API parameters: 3

Application parameters: 1

For a description of this routine's standard parameters, see the standard parameters described on page 7-12.

The following table lists the parameters that pertain specifically to the receipt creation and on-account application routine:

Parameter	Type	Data-type	Required	Description
p_usr_currency_code	IN	VARCHAR2		The translated currency code. Used to derive the p_currency_code if it is not entered. Default: None Validate: Should be a valid currency, so that the corresponding currency code can be derived. Error: AR_RAPI_USR_CURR_CODE_ INVALID
p_currency_code	IN	VARCHAR2 (15)		The actual currency code that gets stored in AR tables. Default: 1. Derived from p_usr_currency_code if entered, else 2. Defaults to the functional currency code Validate: 1. Validated against the currencies in the fnd_currencies table. Error: AR_RAPI_CURR_CODE_INVALID Warning: AR_RAPI_FUNC_CURR_DEFAULTED
p_usr_exchange_rate_type	IN	VARCHAR2		The translated exchange rate type. Used to derive the p_exchange_rate_type if it has not been entered. Default: None Validate: Should be a valid rate type. Error: AR_RAPI_USR_X_RATE_TYP_INVALID

Parameter	Type	Data-type	Required	Description
p_exchange_rate_type	IN	VARCHAR2 (30)		Exchange rate type stored in AR tables. Default: 1. In case of foreign currency receipt, derived from p_usr_exchange_rate_type. 2. In case of foreign currency receipt, defaults from AR: Default Exchange Rate Type profile option. 3. Should be left null, if receipt is in the same denomination as functional currency. Validate: 1. Validated against values in gl_daily_ conversion_types table Error: AR_RAPI_X_RATE_TYPE_ INVALID
p_exchange_rate	IN	NUMBER		The exchange rate between the receipt currency and the functional currency. Default: 1. Derived from the Daily Rates table for rate_type <> 'User' in case of non-functional currency. 2. If profile option Journals: Display Inverse Rate = 'Y', set user-entered value to 1/ p_exchange_rate. 3. The entered value is rounded to a precision of 38. Validate: 1.In case of non-functional currency, the rate should have a positive value for rate type='User'. 2.For non-functional currency and type <> 'User', do not specify any value. Error: AR_RAPI_X_RATE_INVALID AR_RAPI_X_RATE_NULL

Parameter	Type	Data-type	Required	Description
p_exchange_rate_date	IN	DATE		The date on which the exchange rate is valid. Default: Receipt date Validate: 1. For a non-functional currency and type <> 'User', a valid rate should exist in the database for this date. This is a cross validation of type, currency, and date. Error: AR_NO_RATE_DATA_FOUND
p_amount	IN	NUMBER	Yes	The cash receipt amount. Default: Null Validate: > 0 Error: AR_RAPI_REC_AMT_NEGATIVE AR_RAPI_RCPT_AMOUNT_NULL
p_factor_discount_amount	IN	NUMBER		The bank charges on the cash receipt. Default: None Validate: 1. Bank charges are not allowed if profile option AR: Create Bank Charges = 'No'. 2. Bank charges not allowed if the receipt state, derived from the receipt class of the receipt method, <> 'CLEARED'. 3.If allowed, then >= 0. Error: AR_BK_CH_NOT_ALLWD_IF_NOT_CLR AR_JG_BC_AMOUNT_NEGATIVE
p_receipt_number	IN	VARCHAR2 (30)		The receipt number of the receipt to be created. Default: If not specified, the receipt number is defaulted from the document sequence value. Validate: Receipt number should not be null. Error: AR_RAPI_RCPT_NUM_NULL

Parameter	Type	Data-type	Required	Description
p_receipt_date	IN	DATE		The receipt date of the entered cash receipt. Default: System date Validate: None Error: None
p_gl_date	IN	DATE		Date that this receipt will be posted to the general ledger. Default: Gets defaulted to the receipt date if it is a valid gl_date. Validate: The date is valid if the following conditions are true: ■ The date is in an Open or Future period ■ The period cannot be an Adjustment period If the date is invalid, then: ■ If the most recent open period is prior to the receipt date: last date of that period ■ If there is a period open after the receipt date: first date of the last open period Error: AR_INVALID_APP_GL_DATE
p_maturity_date	IN	DATE		Receipt maturity date. Default: Deposit date Validate: >= p_receipt_date Error: AR_RW_MAT_BEFORE_RCT_DATE

Parameter	Type	Data-type	Required	Description
p_customer_id	IN	NUMBER (15)		The customer_id for the paying customer. Default: Refer to *Defaulting* on page 7-53. Validate: 1. Customer exists and has prospect code = 'CUSTOMER' 2. Customer has a profile defined at the customer level Error: AR_RAPI_CUST_ID_INVALID
p_customer_name	IN	VARCHAR2 (50)		The name for the entered customer. Used to default the customer id if not specified. Default: None Validate: None Error: AR_RAPI_CUS_NAME_INVALID
p_customer_number	IN	VARCHAR2 (30)		The customer number. Used to default the customer_id if not specified. Default: None Validate: None Error: AR_RAPI_CUS_NUM_INVALID
p_customer_bank_account_id	IN	NUMBER (15)		The customer bank account id. Default: From bank account id/number Validate: 1. It must be a valid bank account of the paying customer . 2. The inactive date (if defined) of the bank account should be greater than the receipt_date. 3. The receipt date must be within the Start date and the End date of the bank account uses. Error: AR_RAPI_CUS_BK_AC_2_INVALID AR_RAPI_CUS_BK_AC_ID_INVALID

Parameter	Type	Data-type	Required	Description
p_customer_bank_account_num	IN	VARCHAR2 (30)		The customer bank account number. Used to default the customer bank account id, if not specified. Default: None Validate: None Error: None
p_customer_bank_account_name	IN	VARCHAR2 (80)		The customer bank account name. Used to default the customer bank account id, if not specified. Default: None Validate: None Error: None
p_location	IN	VARCHAR2 (40)		The Bill_To location for the customer. Used to derive the p_customer_site_use_id. Default: None Validate: None Error: AR_RAPI_CUS_LOC_INVALID
p_customer_site_use_id	IN	NUMBER (15)		The Bill_To site_use_id for the customer. Default: 1. Defaulted from customer location, else 2. Primary Bill_To customer site_use_id of the customer. Validate: It should be a valid Bill_To site of the paying customer. Error: AR_RAPI_CUS_SITE_USE_ID_INVALID
p_customer_receipt_reference	IN	VARCHAR2 (30)		This column is used to store a customer receipt reference value supplied by the customer at the confirmation time. Default: None Validate: None Error: None

Parameter	Type	Data-type	Required	Description
p_override_remit_bank_account_flag	IN	VARCHAR2 (1)		The flag value decides when the remittance bank account is can be overridden by the remittance selection process. Default: 'Y' Validate: valid values 'Y' and 'N' Error: AR_RAPI_INVALID_OR_REMIT_BK_AC
p_remittance_bank_account_id	IN	NUMBER (15)		Identifies the user's bank account for depositing the receipt. Default: 1.From remittance bank account number 2.From the receipt method based on logic mentioned in *Defaulting* on page 7-24. Validate: Validation logic detailed in *Validation* on page 7-23. Error: AR_RAPI_REM_BK_AC_ID_INVALID AR_RAPI_REM_BK_AC_ID_NULL
p_remittance_bank_account_num	IN	VARCHAR2 (30)		The remittance bank account number. Used to default the remittance bank account id, if not specified. Default: None Validate: None Error: AR_RAPI_REM_BK_AC_NUM_INVALID
p_remittance_bank_account_name	IN	VARCHAR2 (50)		The remittance bank account name. Used to default the remittance bank account id, if not specified. Default: None Validate: None Error: AR_RAPI_REM_BK_AC_NAME_INVALID

Parameter	Type	Data-type	Required	Description
p_deposit_date	IN	DATE		The deposit date. Default: receipt date Validate: None Error: None
p_receipt_method_id	IN	NUMBER (15)		Identifies the payment method of the receipt. Default: From receipt method name. Validate: Validation detailed in *Validation* on page 7-23. Error: AR_RAPI_INVALID_RCT_MD_ID
p_receipt_method_name	IN	VARCHAR2 (30)		The payment method name of the receipt. Used to default the receipt method id if not specified. Default: None Validate: None Error: None **Note:** To use credit card refund functionality, ensure that remittance of the original receipt is performed within Oracle Receivables. Do this by setting the remittance method on the payment method's associated receipt class to *Standard*. **Caution:** If you use this API to both authorize and capture credit card payments, then set the remittance method to *None*. Note, however, that with this setting, you cannot use standard credit card refund functionality. Instead, you must refund such payments *outside* Receivables.

Parameter	Type	Data-type	Required	Description
p_doc_sequence_value	IN	NUMBER		Value assigned to document receipt. Default: Detailed in *Defaulting* on page 7-24. Validate: ■ User should not pass in the value if the current document sequence is automatic. ■ Document sequence value should not be entered if profile option Sequential Numbering is set to Not Used. Error: AR_RAPI_DOC_SEQ_AUTOMATIC AR_RAPI_DOC_SEQ_VAL_INVALID
p_ussgl_transaction_code	IN	VARCHAR2 (30)		Code defined by public sector accounting. Default: None Validate: None Error: None
p_anticipated_clearing_date	IN	DATE		Date the receipt is expected to be cleared. Default: None Validate: >= gl_date Error: AR_RW_EFFECTIVE_BEFORE_GL_DATE
p_event	IN	VARCHAR2		The event that resulted in the creation of the receipt. Currently used only by Bills Receivable. Default: None Validate: None Error: None
p_called_from	IN	VARCHAR2 (20)		This parameter is used to identify the calling routine. Currently used to identify only the 'BR_REMIT' program. Default: None Validate: None Error: None

Parameter	Type	Data-type	Required	Description
p_attribute_record	IN	attribute_ rec_type		This is a record type which contains all the 15 descriptive flexfield segments and one descriptive flexfield structure defining column. It represents the Receipt Information flexfield. Default: DFF APIs complete the defaulting and validation. Validate: DFF APIs complete the defaulting and validation. Error: AR_RAPI_DESC_FLEX_INVALID
p_global_attribute_record	IN	global_ attribute_ rec_type		This is a record type which contains all the 20 global descriptive flexfield segments and one global descriptive flexfield structure defining column. Default: None Validate: None
p_receipt_comments	IN	VARCHAR2 (240)		User's comments for the application.
p_issuer_name	IN	VARCHAR2 (50)		Issuer name of notes receivable (Asia Pacific requirement). Default: None Validate: None Error: None
p_issue_date	IN	DATE		Date when notes receivable was issued (Asia Pacific requirement). Default: None Validate: None Error: None
p_issuer_bank_branch_id	IN	NUMBER (15)		Bank/ Branch issuing the notes receivable (Asia Pacific Requirement). Default: None Validate: None Error: None
p_cr_id	OUT	NUMBER (15)		The cash_receipt_id of the receipt created by the API call.

Parameter	Type	Data-type	Required	Description
p_amount_applied	IN	NUMBER		The amount on the cash receipt that is to be applied to an account. Default: Depending on the profile option AR: Cash-Default Amount Applied, it is defaulted either to: ■ the open amount of the transaction, or ■ the unapplied amount of the receipt. Validate: Less than or equal to the amount due remaining on the receipt. Error: AR_RAPI_APPLIED_AMT_NULL AR_RW_AMOUNT_LESS_THAN_APP
p_apply_date	IN	DATE		Date the application was applied. Default: 1.Receipt date, if receipt date >= system date. 2.System date, if receipt date < system date. Validate: apply date >= receipt date Error: AR_APPLY_BEFORE_RECEIPT
p_apply_gl_date	IN	DATE		Date that this application will be posted to the general ledger. Default: Defaulted to greater of the receipt date and the system date. Validate: 1.Validated as per standard gl date validation described for the gl date in create_cash routine 2. >= receipt gl date Error: AR_INVALID_APP_GL_DATE AR_RW_GL_DATE_BEFORE_REC_GL
p_app_ussgl_transaction_code	IN	VARCHAR2 (30)		Code defined by public sector accounting. Default: None Validate: None

Parameter	Type	Data-type	Required	Description
p_app_attribute_record	IN	attribute_ rec_type		This is a record type which contains all the 15 descriptive flexfield segments and one descriptive flexfield structure defining column. It represents the Receipt Application Information flexfield. Default: DFF APIs complete the defaulting and validation. Validate: DFF APIs complete the defaulting and validation. Error: AR_RAPI_DESC_FLEX_INVALID
p_app_global_attribute_record	IN	global_ attribute_ rec_type		This is a record type which contains all the 20 global descriptive flexfield segments and one global descriptive flexfield structure defining column. Default: None Validate: None
app_comments	IN	VARCHAR2 (240)		User's comments for the application.
p_application_ref_num	IN	VARCHAR2 (30)		Deduction number, if resulting from Trade Management claim settlement.
p_secondary_application_ref_ id	IN	NUMBER (15)		Claim ID, if resulting from Trade Management claim settlement.
p_customer_reference	IN	VARCHAR2 (100)		Reference supplied by customer.
p_customer_reason	IN	VARCHAR2 (20)		Reason code supplied by customer.
p_secondary_app_ref_type	IN	VARCHAR2 (30)		Used for automated receipt handling, leave null.
p_secondary_app_ref_num	IN	VARCHAR2 (30)		Used for automated receipt handling, leave null.
p_call_payment_processor	IN	VARCHAR2 (1)		This is the payment processing indicator flag. Pass as FND_API.G_TRUE, if you want to call *i*Payment payment APIs for credit card processing.

Example

Objective:

To create a cash receipt and apply to On Account in the functional currency using a call to the API Ar_receipt_api_pub.Create_Apply_on_acc and passing a minimum number of input parameters.

This table lists the entered parameters:

Parameter	Entered Value	Default Value
p_api_version	1.0	
p_receipt_number	'aj_test_api_3'	
p_amount	1000	
p_receipt_method_id	1001	
p_customer_name	'Computer Service and Rentals'	

This table lists the defaulted input parameters, which were not entered:

Parameter	Entered Value	Default Value
p_customer_id		1006
p_currency_code		USD
p_receipt_date		19-APR-2004
p_gl_date		19-APR-2004
p_deposit_date		19-APR-2004
p_customer_site_use_id		1025
p_override_remit_bank_account_flag		'Y'
p_remittance_bank_account_id		10001
p_maturity_date		19-APR-2004
p_apply_gl_date		19-APR-2004
p_apply_date		19-APR-2004

Parameter	Entered Value	Default Value
p_amount_applied		1000
p_amount_applied_from		1000
p_call_payment_processor*		fnd_api.g_false

Result:

We were able to create the cash receipt 'aj_test_api_3' and then apply it to 'On account' by specifying only 5 input parameters in our call to this API. The receipt is in the functional currency. The retrieval and handling of the warnings and the error messages, put on the message stack by the API during execution, are the same as described in *Defaulting* on page 7-24.

Messages

Messages play an important role in the effectiveness of your API calls. The right message is raised at the right point to convey to you the exact error that has occurred or any warnings that have been raised.

The Receipt API puts on the message stack all error messages and warnings raised during execution. You can retrieve messages and warnings as described in *Exception Handling and Result Messages* on page 7-8.

WARNINGS AND ERRORS

The following table lists all the error messages raised by the Receipt API:

TYPE

E: Error message

W: Warning message

MESSAGE_CODE	MESSAGE_TEXT	EXPLANATION	TYPE
AR_APPLY_BEFORE_RECEIPT	Apply Date must be greater than or equal to the Receipt Date.		E
AR_APPLY_BEFORE_ TRANSACTION	Apply Date must be greater than or equal to the Transaction Date.		E
AR_BK_CH_NOT_ALLWD_IF_ NOT_CLR	For a receipt status other than cleared, bank charges are not allowed.		E
AR_EXCHANGE_RATE_NEGATIVE	Please enter a positive exchange rate.		E
AR_EXCHANGE_RATE_ZERO	The exchange rate cannot be zero.		E
AR_INVALID_APP_GL_DATE	GL date, &GL_DATE, is not in an open or future-enterable period.		E
AR_JG_BC_AMOUNT_NEGATIVE	The Bank Charges amount cannot be negative.		E
AR_NO_PARTIAL_DISC	No discounts allowed on this installment unless it is fully paid.		E
AR_NO_RATE_DATA_FOUND	There is no rate for this currency, rate date and rate type in the database.		E

MESSAGE_CODE	MESSAGE_TEXT	EXPLANATION	TYPE
AR_OVERR_REM_BK_FLAG_ INVALID	Override remittance bank flag has invalid value.		E
AR_RAPI_CUS_BK_NAME_NUM_ IGN	Customer bank account identifier has taken a precedence over the customer bank account name and number.		W
AR_RAPI_ACTIVITY_INVALID	The receivables activity name is invalid.		E
AR_RAPI_ACTIVITY_IGN	Both a receivables transaction identifier and a receivables activity exist for this record. The receivables transaction identifier takes precedence over the receivables activity.		W
AR_RAPI_TAX_RATE_AMT_X_ INVALID	Please enter a different combination of receipt amount, tax amount, and tax rate.		E
AR_RAPI_TAX_CODE_INVALID	The tax code is invalid.		E
AR_RAPI_TAX_RATE_INVALID	The tax rate is invalid.		E
AR_RAPI_TAX_CODE_IGN	Both a VAT identifier and a tax code exist for this record. The VAT identifier takes precedence over the tax code.		W
AR_RAPI_REC_TRX_ID_NULL	Please enter a receivables transaction identifier.		E
AR_RAPI_VAT_TAX_ID_INVALID	The VAT identifier is invalid.		E
AR_RAPI_REF_TYPE_INVALID	The reference type is invalid.		E
AR_RAPI_REF_NUM_INVALID	The reference number is invalid.		E
AR_RAPI_REF_NUM_IGN	Both a reference identifier and a reference number exist for this record. The reference identifier takes precedence over the reference number.		W
AR_RAPI_REF_ID_INVALID	The reference identifier is invalid.		E
AR_RAPI_REF_ID_NULL	Please enter a reference identifier.		E
AR_RAPI_REF_TYPE_NULL	Please enter a reference type.		E

MESSAGE_CODE	MESSAGE_TEXT	EXPLANATION	TYPE
AR_RAPI_ACTIVITY_X_INVALID	The specified combination of payment schedule identifier and receivables transaction identifier is invalid.	The activity type derived from the receivables_trx_ id does not match with the activity type of the specified payment_ schedule_id.	E
AR_RAPI_AMT_APP_FROM_ INVALID	The allocated receipt amount and the applied amount should be same for the functional currency receipt.		E
AR_RAPI_APP_PS_ID_INVALID	Applied payment schedule identifier has an invalid value.		E
AR_RAPI_APP_PS_RA_ID_X_ INVALID	Invalid receivable application identifier for the specified applied payment schedule identifier.		E
AR_RAPI_APPLIED_AMT_NULL	Applied amount could not be defaulted.	The p_applied_amount was not specified by the user and it could not be defaulted from the specified receipt or the specified transaction. For explanation on defaulting mechanism refer *Defaulting* on page 7-34	E
AR_RAPI_CASH_RCPT_ID_ INVALID	Invalid cash receipt identifier.		E
AR_RAPI_CASH_RCPT_ID_NULL	Cash receipt identifier is null.		E
AR_RAPI_CC_RATE_AMTS_ INVALID	The entered combination of the applied amount, allocated amount and the cross currency rate is invalid.	This error is raised if the following condition is violated in the cross currency applications: p_trans_to_receipt_rate* p_amount_applied = p_amount_applied_from.	E
AR_RAPI_CC_RATE_INVALID	Do not enter the cross currency rate if the receipt and the transaction are in same currency.	For the same currency receipt application, p_ trans_to_receipt_rate should not be specified.	E

MESSAGE_CODE	MESSAGE_TEXT	EXPLANATION	TYPE
AR_RAPI_CC_RATE_NULL	Cross currency rate is null.	In case of a cross currency receipt application, the p_trans_to_receipt_rate could neither be defaulted nor derived.	E
AR_RAPI_CURR_CODE_INVALID	Currency code is invalid.	The specified currency code has an invalid value.	E
AR_RAPI_CUS_BK_AC_2_INVALID	Invalid combination of customer bank account name and number.	The specified combination of the p_customer_bank_account_number and p_customer_bank_account_name is invalid and cannot be used to derive the p_customer_bank_account_id.	E
AR_RAPI_CUS_BK_AC_ID_INVALID	Customer bank account identifier is invalid.	The specified value of p_customer_bank_account_id is invalid.	E
AR_RAPI_CUS_BK_AC_NAME_INVALID	Customer bank account name is invalid.	The specified value of p_customer_bank_account_name is invalid.	E
AR_RAPI_CUS_BK_AC_NUM_INVALID	Customer bank account number is invalid.	The specified value of p_customer_bank_account_number is invalid.	E
AR_RAPI_CUS_LOC_INVALID	Customer location is invalid for the specified customer.	The specified value of p_location has an invalid value.	E
AR_RAPI_CUS_NAME_INVALID	Invalid customer name.		E
AR_RAPI_CUS_NAME_NUM_INVALID	Invalid combination of customer name and number.		E
AR_RAPI_CUS_NUM_INVALID	Invalid customer number.		E
AR_RAPI_CUS_SITE_USE_ID_INVALID	Customer site use identifier is invalid for the specified customer.	The specified value of p_customer_site_use_id is invalid for the given customer. It should be a valid BILL_TO site_use_id for the customer.	E

MESSAGE_CODE	MESSAGE_TEXT	EXPLANATION	TYPE
AR_RAPI_CUS_STE_USE_ID_NOT_DEF	Location could not be defaulted for the specified customer.	Neither the user had passed in any value for the p_location / p_customer_site_use_id, nor could it be defaulted to the primary Bill_To location for the given customer.	W
AR_RAPI_CUST_ID_INVALID	Customer identifier is invalid.		E
AR_RAPI_CUST_ID_NULL	Customer identifier is null.	The p_customer_id is null. For details, refer to *API Usage* on page 7-11.	E
AR_RAPI_CUS_NAME_NUM_IGN	Customer identifier has taken a precedence over name and number.	The specified values of p_customer_number and/or p_customer_name are ignored if the value for p_customer_id has been passed in.	W
AR_RAPI_CUST_TRX_ID_INVALID	Invalid customer transaction identifier.		E
AR_RAPI_CUST_TRX_ID_NULL	Customer transaction identifier is null.		E
AR_RAPI_DEF_TAX_FLAG_INVALID	Invalid deferred tax flag.	The valid values are 'Y'/'N'	E
AR_RAPI_DESC_FLEX_INVALID	The entered values for the descriptive flexfield &DFF_NAME is invalid.		E
AR_RAPI_DOC_SEQ_AUTOMATIC	You have passed in the document sequence value, even though the current document sequence is automatic.		E
AR_RAPI_DOC_SEQ_NOT_EXIST_A	Document sequence does not exist for the current document even though profile option Sequential Numbering is set to Always Used.		E
AR_RAPI_DOC_SEQ_NOT_EXIST_P	Document sequence does not exist for the current document even though profile option Sequential Numbering is set to Partially Used.		W

MESSAGE_CODE	MESSAGE_TEXT	EXPLANATION	TYPE
AR_RAPI_DOC_SEQ_VAL_ INVALID	Document sequence value should not be entered if profile option Sequential Numbering is set to Not Used.		E
AR_RAPI_DOC_SEQ_VALUE_ NULL_A	The profile option Sequential Numbering is set to Always Used and the document sequence is manual. The document sequence value is null.		E
AR_RAPI_DOC_SEQ_VALUE_ NULL_P	The profile option Sequential Numbering is set to Partially Used and the document sequence is manual. The document sequence value is null.		W
AR_RAPI_FUNC_CURR_ DEFAULTED	Functional currency defaulted as the receipt currency.		W
AR_RAPI_INS_PS_NOT_DEF_CUS	The customer could not be defaulted from the applied payment schedule identifier and the installment.	This error is raised if the customer_id cannot be derived from the p_ applied_payment_ schedule_id and the p_ installment specified in the create_and_apply routine.	E
AR_RAPI_INSTALL_NULL	The installment number is null.		E
AR_RAPI_INVALID_APP_REF	Please supply a valid application reference type.		E
AR_RAPI_INVALID_CLAIM_ID	A valid claim ID &CLAIM_ID does not exist for the specified receipt and amount.		E
AR_RAPI_INVALID_CLAIM_NUM	The claim is invalid. Please enter a different claim number.		E
AR_RAPI_INVALID_REF_REASON	Please supply a valid reference reason.		E

MESSAGE_CODE	MESSAGE_TEXT	EXPLANATION	TYPE
AR_RAPI_MULTIPLE_ON_AC_APP	More than one On Account application exists for the current receipt. Please specify the receivable application identifier.	This error is raised in the unapply_on_account routine if for the specified cash receipt, more than one On Account application exists and the p_receivable_application_id is not specified.	E
AR_RAPI_NON_REVERSIBLE	Standard reversal not possible for this receipt.	Explanation: refer to *Defaulting* on page 7-69.	E
AR_RAPI_PSID_NOT_DEF_CUS	The customer could not be defaulted from the applied payment schedule identifier.	This error is raised in the create_and_apply routine if the customer is not entered and cannot be derived from the specified p_applied_payment_schedule_id.	E
AR_RAPI_RCPT_AMOUNT_NULL	Receipt amount is null.	This is a required field in the create_cash and the create_and_apply routines.	E
AR_RAPI_RCPT_MD_ID_NULL	Receipt method identifier is null.		E
AR_RAPI_RCPT_MD_NAME_IGN	Receipt method identifier has taken precedence over receipt method name.		W
AR_RAPI_RCPT_MD_NAME_INVALID	Invalid receipt method name.	This error is raised if the p_receipt_method_id is not passed in and the specified p_receipt_method_name is invalid.	E
AR_RAPI_RCPT_NOT_APP_TO_INV	There is no application of the entered receipt against the entered transaction.	This error is raised in the Unapply routine, if the specified receipt has no application against the specified transaction.	E
AR_RAPI_RCPT_NUM_IGN	Cash receipt identifier has taken a precedence over the receipt number.		W
AR_RAPI_RCPT_NUM_INVALID	Invalid receipt number.		E

MESSAGE_CODE	MESSAGE_TEXT	EXPLANATION	TYPE
AR_RAPI_RCPT_RA_ID_X_ INVALID	Invalid combination of receivable application identifier and the cash receipt identifier.	The p_cr_id derived from the p_receivable_ application_id specified by the user does not match with the p_cr_id which is either specified by the user or defaulted from the p_ receipt_number.	E
AR_RAPI_RCT_MD_ID_INVALID	Invalid receipt method identifier.		E
AR_RAPI_RCPT_MD_NAME_ INVALID	Invalid receipt method name.		E
AR_RAPI_REC_APP_ID_INVALID	Invalid receivable application identifier.		E
AR_RAPI_REC_APP_ID_NULL	Receivable application identifier is null.	BR	E
AR_RAPI_REC_TRX_ID_INVALID	Invalid receivable transaction identifier.		E
AR_RAPI_REM_BK_AC_2_INVALID	Invalid combination of remittance bank account name and number.	The specified combination of the p_remittance_bank_ account_number and p_ remittance_bank_account_ name is invalid, and cannot be used to derive the p_remittance_bank_ account_id.	E
AR_RAPI_REM_BK_AC_ID_ INVALID	Invalid remittance bank account identifier.	This error is raised if the specified p_remittance_ bank_account_id is not associated with the specified p_receipt_ method_id.	E
AR_RAPI_REM_BK_AC_ID_NULL	Remittance bank account identifier is null.		E
AR_RAPI_REM_BK_AC_NAME_ INVALID	Invalid remittance bank account name.		E
AR_RAPI_REM_BK_AC_NAME_ NUM_IGN	Remittance bank account identifier has taken a precedence over the remittance bank account name and number.		W

MESSAGE_CODE	MESSAGE_TEXT	EXPLANATION	TYPE
AR_RAPI_REM_BK_AC_NUM_INVALID	Invalid remittance bank account number.		E
AR_RAPI_REV_CAT_CD_INVALID	Invalid reversal category code.		E
AR_RAPI_REV_CAT_CD_NULL	Reversal category code is null.		E
AR_RAPI_REV_CAT_NAME_IGN	Reversal category code has taken precedence over the reversal category name.		W
AR_RAPI_REV_CAT_NAME_INVALID	Invalid reversal category name.		E
AR_RAPI_REV_GL_DATE_NULL	Reversal GL date is null.		E
AR_RAPI_REV_REAS_CD_INVALID	Invalid reversal reason code.		E
AR_RAPI_REV_REAS_CD_NULL	Reversal reason code is invalid.		E
AR_RAPI_REV_REAS_NAME_IGN	Reversal reason code has taken a precedence over the reversal reason name.		W
AR_RAPI_REV_REAS_NAME_INVALID	Invalid reversal reason name.		E
AR_RAPI_TRX_ID_INST_INVALID	Invalid combination of the customer transaction identifier and installment.		E
AR_RAPI_TRX_INS_NOT_DEF_CUS	The customer could not be defaulted from the entered transaction and the installment.	This error is raised in the create_and_apply routine if the customer is not entered and cannot be derived from the specified transaction and installment.	E
AR_RAPI_TRX_INS_PS_NOT_DEF_CUS	The customer could not be defaulted from the entered transaction, installment and applied payment schedule identifier.	This error is raised in the create_and_apply routine if the customer is not entered and cannot be derived from the specified p_customer_trx_id/trx_number, p_installment and p_applied_payment_schedule_id.	E

MESSAGE_CODE	MESSAGE_TEXT	EXPLANATION	TYPE
AR_RAPI_TRX_LINE_AMT_DEFLT	Amount applied has been defaulted to the line amount of the specified transaction line.		W
AR_RAPI_TRX_LINE_ID_INVALID	Invalid customer transaction line identifier.		E
AR_RAPI_TRX_LINE_NO_INVALID	Invalid transaction line number.		E
AR_RAPI_TRX_NOT_DEF_CUST	The customer could not be defaulted from the entered transaction.	This error is raised in the create_and_apply routine if the customer is not entered and cannot be derived from the specified p_customer_trx_id/trx_number.	E
AR_RAPI_TRX_NUM_IGN	Customer transaction identifier has taken a precedence over the transaction number.		W
AR_RAPI_TRX_NUM_INST_INVALID	Invalid combination of transaction number and installment.		E
AR_RAPI_TRX_NUM_INVALID	Invalid transaction number.		E
AR_RAPI_TRX_PS_ID_X_INVALID	Invalid applied payment schedule identifier for the specified transaction.	The p_applied_payment_schedule_id specified by the user does not match with the payment_schedule_id derived from the p_customer_trx_id and the p_installment.	E
AR_RAPI_TRX_PS_NOT_DEF_CUS	The customer could not be defaulted from the entered transaction and the applied payment schedule identifier.	This error is raised in the create_and_apply routine if the customer is not entered and cannot be derived from the specified p_customer_trx_id/trx_number and the p_applied_payment_schedule_id.	E
AR_RAPI_TRX_RA_ID_X_INVALID	The activity type for the entered receivable transaction identifier does not match with the activity of the entered payment schedule identifier.	This message is to be used by the API, activity_application, added as part of the Bills Receivables changes.	E

MESSAGE_CODE	MESSAGE_TEXT	EXPLANATION	TYPE
AR_RAPI_USR_CURR_CODE_IGN	Currency code took a precedence over the user currency code.		W
AR_RAPI_USR_CURR_CODE_ INVALID	User currency code is invalid.		E
AR_RAPI_USR_X_RATE_TYP_ INVALID	User exchange rate type is invalid.		E
AR_RAPI_USR_X_RATE_TYPE_IGN	Exchange rate type took a precedence over the User exchange rate type.		W
AR_RAPI_X_RATE_DATE_INVALID	Invalid exchange rate date.		E
AR_RAPI_X_RATE_INVALID	Exchange rate should not be entered.	This would be raised if the exchange rate type is not 'User' and the exchange rate has been specified.	E
AR_RAPI_X_RATE_NULL	Exchange rate is null.		E
AR_RAPI_X_RATE_TYPE_INVALID	Invalid exchange rate type.		E
AR_RAPI_X_RATE_TYPE_NULL	Exchange rate type is null.		E
AR_RW_AMOUNT_LESS_THAN_ APP	The receipt amount cannot be less than the sum of the applied and on-account amounts.		E
AR_RW_APP_NEG_ON_ACCT	Amount applied cannot be negative for an On Account application.		E
AR_RW_APP_NEG_UNAPP	You may not apply more than the receipt amount.	This error is raised if you try to apply more than the unapplied amount on the receipt against a transaction.	E
AR_RW_APPLIED_GREATER_LINE	Amount applied cannot be greater than the original line amount of &AMOUNT.	This error is raised in the apply and create_and_ apply routines if the line number of transaction has been specified and the amount applied is greater than the original line amount of the transaction line.	E

MESSAGE_CODE	MESSAGE_TEXT	EXPLANATION	TYPE
AR_RW_BEFORE_APP_GL_DATE	Reversal GL Date must be on or after original GL Date of &GL_ DATE.		E
AR_RW_BEFORE_RECEIPT_GL_ DATE	The Reversal GL Date cannot be before the Receipt GL Date.		E
AR_RW_CASH_DUPLICATE_ RECEIPT	A cash receipt with this number, date, amount and customer already exists.		E
AR_RW_CC_RATE_POSITIVE	Cross currency rate must be greater than zero.	This error is raised in the apply and create_and_ apply routines if the p_ trans_to_receipt_rate has a negative value.	E
AR_RW_GL_DATE_BEFORE_REC_ GL	The GL date cannot be before the receipt GL date.	This error is raised in the apply and the create_and_ apply routines if the apply gl_date is before the receipt gl_date.	E
AR_RW_GL_DATE_BEFORE_ OPEN_REC_GL	The application GL date must be later than the open receipt GL date for a receipt-to-receipt application.		E
AR_RW_MAT_BEFORE_RCT_DATE	The Maturity Date cannot be before the Receipt Date.		E
AR_RW_NET_DIFF_RCT_CURR	Both receipts in a receipt to receipt application must have the same currency.		E
AR_RW_NET_OPEN_AMT_INC	A receipt-to-receipt application must decrease the open receipt balance or bring the receipt balance closer to zero.		E
AR_RW_NET_OPEN_RCT_ONLY	Netting is allowed on open receipts only (unapplied cash, on-account cash and claim investigation applications).		E
AR_RW_NET_UNAPP_OVERAPP	Unapplying this payment netting application is not allowed because it would cause the applied receipt balance to become negative.		

MESSAGE_CODE	MESSAGE_TEXT	EXPLANATION	TYPE
AR_RW_NO_DISCNT	Discounts are not permitted for transactions with a negative original balance.		E
AR_RW_PAID_INVOICE_TWICE	You have paid the same invoice twice. Please correct.		E
AR_RW_RCT_AMOUNT_NEGATIVE	You cannot enter a negative receipt amount for cash receipts.		E
AR_RW_VAL_DISCOUNT	Discount taken is greater than the discount available (&DISC_AVAILABLE).		E
AR_RW_VAL_NEG_DISCNT	Discount cannot be negative.		E
AR_RW_VAL_ONACC_DISC	Discount not allowed for On Account application. Clear discount amount field or enter zero.		E
AR_RW_VAL_UNEARNED_DISCOUNT	Cannot take unearned discount because the Allow Unearned Discount system option is set to No.		E
AR_SYSTEM_WR_NO_LIMIT_SET	Please set the receipt write-off limits range system option.		E
AR_VAL_GL_INV_GL	The GL date should not be prior to the invoice's GL date.		E
AR_WR_NO_LIMIT	User Write-off limit does not exist.		E
AR_WR_TOTAL_EXCEED_MAX_AMOUNT	The total write-off amount must fall within the receipt write-off limits range system option.		E
AR_WR_USER_LIMIT	Total write-off amount must be in the range of &FROM_AMOUNT to &TO_AMOUNT.		E

8

Revenue Adjustment API User Notes

Overview

This document outlines the specifications and the methodology for using the various Revenue Adjustment APIs. These APIs provide an extension to existing functionality of adjusting revenue and sales credits through the standard AR Revenue Management form.

You can access these APIs:

- As standard PL/SQL servers-side routine calls
- Through forms, utilizing the capability of Forms6 to have a procedure as its underlying base table

Basic Business Needs

The Revenue Adjustment API addresses the following basic functionality via different API calls:

- Unearning revenue
- Earning revenue
- Transferring sales credits between salespersons
- Adding new non-revenue sales credits

Presently, the main business need for the API is the requirement to have event-based revenue recognition. In Receivables, it is now possible to defer revenue recognition, and to earn the revenue at a later date using the API. Throughout the process, the API uses AutoAccounting to determine the accounts to be debited/credited with each operation.

Before you begin....

Initialization of ARP_STANDARD and ARP_GLOBAL

Custom code that uses AR or HZ APIs will set the ORG_ID via dbms_application_info.set_client_info() and then call the APIs. The APIs in turn might access either ARP_STANDARD and ARP_GLOBAL, which initialize the global variables that are used across Oracle Receivables when the package is first called. Most of these global variable values are organization dependent, and the first such call sets the global variables based on the current ORG_ID.

If additional custom code then changes the ORG_ID via another call to dbms_application_info.set_client_info(), then the ORG context changes, ***but the ARP_STANDARD and ARP_GLOBAL context does not.***

In such cases, you should explicitly re-initialize the global variables by a call to these two public procedures:

1. ARP_GLOBAL.INIT_GLOBAL: For setting public variables in ARP_GLOBAL.
2. ARP_STANDARD.INIT_STANDARD: For setting public variables in ARP_STANDARD.

Major Features

Flexibility

Per Oracle API coding standards, the various APIs in the Revenue Adjustment API package provide the option of specifying an ID or its associated value for any attribute which is an input parameter of the API. If both of the ID and value are specified, then the ID takes precedence over the value. This provides a wide degree of flexibility for using the API, both as a base table of the form and as a server-side routine call from the PL/SQL code.

The extensive defaulting mechanism for the input parameters ensures that you can achieve the basic business needs of adjusting revenue by calling the relevant APIs with a minimum number of parameters.

Modular Approach

The Revenue Adjustment API has been designed in a modular fashion, giving you code that is:

- Easy to understand
- Easy to maintain
- Easy to extend

Error Handling

The Revenue Adjustment API provides an extensive error-handling and error-reporting mechanism whereby all errors encountered in the Defaulting and Validation phases are reported and put on the message stack. The calling program has the option of looking at all the error messages or only the first error message on the stack. If only one error is in the message stack, then the message comes out as one of the output parameters of the API routine; you do not need to fetch the message from the stack.

Robust Validation

The validation performed by the Revenue Adjustment API collects all errors encountered and puts them on the message stack. The relevant entity handler is called only if no errors are reported during the Defaulting and Validation phases.

Debug Messages

Extensive debug messages have been incorporated. In the case of unexpected problems, these can be used to troubleshoot any problems encountered with the API.

Debug messages can be written to the log file by calling the appropriate routines described in *Exception Handling and Result Messages* on page 8-7.

Solution Outline

PL/SQL APIs

To achieve the basic functionality of earning revenue, unearning revenue, transferring sales credits, and adding non-revenue sales credits at the transaction, item, category, or transaction line level, the following four APIs can be called:

- *AR_RevenueAdjust_PUB.Unearn_Revenue* on page 8-10: Transfers the specified amount of revenue from the revenue account to the unearned revenue account on the specified transaction lines.
- *AR_RevenueAdjust_PUB.Earn_Revenue* on page 8-23: Transfers the specified amount of revenue from the unearned revenue account to the revenue account on the specified transaction lines.
- *AR_RevenueAdjust_PUB.Transfer_Sales_Credits* on page 8-26: Transfers revenue and/or non-revenue sales credits between salespersons on the specified transaction lines. In the case of revenue sales credits, the associated revenue is also transferred between cost centers, assuming that AutoAccounting derives the cost center segment of the accounting flexfield from the salesperson.
- *AR_RevenueAdjust_PUB.Add_Non_Revenue_Sales_Credits* on page 8-31: Adds non-revenue sales credits for any salesperson to the specified transaction lines.

For all options, a specific amount or percentage of the total value can be specified. All available revenue can also be specified, except for Add_Non_Revenue_Sales_Credits, where this is not applicable.

> **Note:** You cannot specify *both* revenue and nonrevenue sales credits when passing sales group information to the above APIs.

Modular Approach

The Revenue Adjustment API is divided into four parts. The API:

1. Defaults the ID's from the values and cross validating if both the values and the IDs are entered by the user.
2. Defaults all the entity level information which the user has not entered or which the API needs internally.
3. Validates the entity level information entered by the user.

4. Calls the entity handlers to perform the relevant task, such as Unearn, Earn, Transfer, or Add sales credits.

This results in easy to understand and easy to maintain code. Any new functionality can be added by a simple code plug-in at each of the four parts.

Defaulting

In general, the various parameters in the API call, if not entered, get defaulted based on the values of the other parameters in the API call.

Depending on the exact business requirements, the minimum number of parameters that may be required to perform certain business tasks might vary.

Null values are defaulted for the parameters which could not be defaulted by the API defaulting routines.

For various attributes of the business objects, you can pass either the ID or the value of the attribute. If you specify only the value, then the value is used to derive the ID; otherwise, the ID (if specified) is taken directly. If you specify both the ID and the value, then the ID takes precedence over the value.

Exception Handling and Result Messages

The Revenue Adjustment APIs give back three types of information to their calling programs:

- Overall status
- Messages describing the operations performed or errors encountered by the APIs
- Some output values that the API caller might want to use (this is different for different API routines and is described in *API Usage* on page 8-10)

Return Status

The return status (x_return_status) of the API informs the caller about the result of the operation (or operations) performed by the API. The different possible values for an API return status are:

- Success (FND_API.G_RET_STS_SUCCESS)
- Error (FND_API.G_RET_STS_ERROR)
- Unexpected error (FND_API G_RET_STS_UNEXP_ERROR)

The following section describes the different values of return status and their meanings.

Success

A success return status means that the API was able to perform all the operations requested by its caller. A success return status may be accompanied by informative messages in the API message list.

Error

An error return status means that the API failed to perform some or all of the operations requested by its caller. An error return status is usually accompanied by messages describing the error (or errors) and how to fix it.

In most cases, you should be able to take corrective action to fix regular, expected errors such as missing attributes or invalid date ranges.

Unexpected error

An unexpected error status means that the API has encountered an error condition it did not expect or could not handle. In this case, the API is unable to continue with its regular processing. Examples of such errors are irrecoverable data inconsistency errors, memory errors, and programming errors (such as attempting a division by zero).

In most cases, only system administrators or application developers can fix these unexpected errors.

Messages

The APIs put result messages into a message list. Programs calling these APIs can then get the messages from the list and process them by issuing them, loading them into a database table, or writing them to a log file.

Messages are stored in an encoded format to let the API callers find message names using the standard functions provided by the message dictionary. It also allows the storing these messages in database tables and reporting off these tables in different languages. See *Messages* on page 8-35 for more information.

The API message list must be initialized every time a program calls an API. API callers can either call the message list utility function FND_MSG_PUB.Initialize or request that the API do the initialization on their behalf by setting the p_init_msg_ list parameter to TRUE.

The program calling the API can retrieve messages from the message stack using the existing FND API functions FND_MSG_PUB.Count_Msg and FND_MSG_PUB.Get.

Message Level Threshold:

There is currently no message level threshold. All error messages are output to the message stack.

Debug Messages

The calling program enables debugging by calling the routine *arp_standard.enable_file_debug (<pathname>, <filename>)*.

This routine takes in two parameters, path_name and file_name. To find the path name, use the following select statement:

```
select value from v$parameter where name = 'utl_file_dir',
```

The file name can be any name that you choose.

Example:

```
arp_standard.enable_file_debug ('/sqlcom/log', 'txt.log');
```

This call would write the output debug file 'txt.log' in the path '/sqlcom/log'.

API Usage

AR_RevenueAdjust_PUB.Unearn_Revenue

Description

Call this routine to move revenue from the earned revenue account to the unearned revenue account using AutoAccounting. This API routine has 4 input and 5 output parameters in total. One of the input parameters is a record type that holds all the revenue adjustment information and has 120 elements. The output parameters include the revenue_adjustment_number and revenue_adjustment_id of the revenue adjustment.

The following is the breakdown of the parameters:

Input

Standard API parameters: 3

Revenue Adjustment parameters: 1 (revenue adjustment record type)

Output

Standard API parameters: 3

Revenue Adjustment parameters: 2

The input revenue adjustment parameter is a record of type *AR_Revenue_Adjustment_PVT.Rev_Adj_Rec_Type.*

```
TYPE Rev_Adj_Rec_Type IS RECORD
     (CUSTOMER_TRX_ID                    NUMBER(15)
    ,TRX_NUMBER                          RA_CUSTOMER_TRX.trx_number%TYPE
    ,BATCH_SOURCE_NAME                   RA_BATCH_SOURCES.name%TYPE
    ,ADJUSTMENT_TYPE                     VARCHAR2(15) DEFAULT 'UN'
    ,FROM_SALESREP_ID                    NUMBER(15)
    ,FROM_SALESREP_NUMBER                RA_SALESREPS.salesrep_number%TYPE
    ,TO_SALESREP_ID                      NUMBER(15)
    ,TO_SALESREP_NUMBER                  RA_SALESREPS.salesrep_number%TYPE
    ,FROM_SALESGROUP_ID                  jtf_rs_groups_b.group_id%TYPE
    ,TO_SALESGROUP_ID                    jtf_rs_groups_b.group_id%TYPE
    ,SALES_CREDIT_TYPE                   VARCHAR2(15) DEFAULT 'R'
    ,AMOUNT_MODE                         VARCHAR2(15) DEFAULT 'T'
    ,AMOUNT                              NUMBER
    ,PERCENT                             NUMBER
    ,LINE_SELECTION_MODE                 VARCHAR2(15) DEFAULT 'A'
```

,FROM_CATEGORY_ID	NUMBER(15)
,FROM_CATEGORY_SEGMENT1	VARCHAR2(40)
,FROM_CATEGORY_SEGMENT2	VARCHAR2(40)
,FROM_CATEGORY_SEGMENT3	VARCHAR2(40)
,FROM_CATEGORY_SEGMENT4	VARCHAR2(40)
,FROM_CATEGORY_SEGMENT5	VARCHAR2(40)
,FROM_CATEGORY_SEGMENT6	VARCHAR2(40)
,FROM_CATEGORY_SEGMENT7	VARCHAR2(40)
,FROM_CATEGORY_SEGMENT8	VARCHAR2(40)
,FROM_CATEGORY_SEGMENT9	VARCHAR2(40)
,FROM_CATEGORY_SEGMENT10	VARCHAR2(40)
,FROM_CATEGORY_SEGMENT11	VARCHAR2(40)
,FROM_CATEGORY_SEGMENT12	VARCHAR2(40)
,FROM_CATEGORY_SEGMENT13	VARCHAR2(40)
,FROM_CATEGORY_SEGMENT14	VARCHAR2(40)
,FROM_CATEGORY_SEGMENT15	VARCHAR2(40)
,FROM_CATEGORY_SEGMENT16	VARCHAR2(40)
,FROM_CATEGORY_SEGMENT17	VARCHAR2(40)
,FROM_CATEGORY_SEGMENT18	VARCHAR2(40)
,FROM_CATEGORY_SEGMENT19	VARCHAR2(40)
,FROM_CATEGORY_SEGMENT20	VARCHAR2(40)
,TO_CATEGORY_ID	NUMBER(15)
,TO_CATEGORY_SEGMENT1	VARCHAR2(40)
,TO_CATEGORY_SEGMENT2	VARCHAR2(40)
,TO_CATEGORY_SEGMENT3	VARCHAR2(40)
,TO_CATEGORY_SEGMENT4	VARCHAR2(40)
,TO_CATEGORY_SEGMENT5	VARCHAR2(40)
,TO_CATEGORY_SEGMENT6	VARCHAR2(40)
,TO_CATEGORY_SEGMENT7	VARCHAR2(40)
,TO_CATEGORY_SEGMENT8	VARCHAR2(40)
,TO_CATEGORY_SEGMENT9	VARCHAR2(40)
,TO_CATEGORY_SEGMENT10	VARCHAR2(40)
,TO_CATEGORY_SEGMENT11	VARCHAR2(40)
,TO_CATEGORY_SEGMENT12	VARCHAR2(40)
,TO_CATEGORY_SEGMENT13	VARCHAR2(40)
,TO_CATEGORY_SEGMENT14	VARCHAR2(40)
,TO_CATEGORY_SEGMENT15	VARCHAR2(40)
,TO_CATEGORY_SEGMENT16	VARCHAR2(40)
,TO_CATEGORY_SEGMENT17	VARCHAR2(40)
,TO_CATEGORY_SEGMENT18	VARCHAR2(40)
,TO_CATEGORY_SEGMENT19	VARCHAR2(40)
,TO_CATEGORY_SEGMENT20	VARCHAR2(40)
,FROM_INVENTORY_ITEM_ID	NUMBER(15)
,FROM_ITEM_SEGMENT1	VARCHAR2(40)
,FROM_ITEM_SEGMENT2	VARCHAR2(40)

```
,FROM_ITEM_SEGMENT3                VARCHAR2(40)
,FROM_ITEM_SEGMENT4                VARCHAR2(40)
,FROM_ITEM_SEGMENT5                VARCHAR2(40)
,FROM_ITEM_SEGMENT6                VARCHAR2(40)
,FROM_ITEM_SEGMENT7                VARCHAR2(40)
,FROM_ITEM_SEGMENT8                VARCHAR2(40)
,FROM_ITEM_SEGMENT9                VARCHAR2(40)
,FROM_ITEM_SEGMENT10               VARCHAR2(40)
,FROM_ITEM_SEGMENT11               VARCHAR2(40)
,FROM_ITEM_SEGMENT12               VARCHAR2(40)
,FROM_ITEM_SEGMENT13               VARCHAR2(40)
,FROM_ITEM_SEGMENT14               VARCHAR2(40)
,FROM_ITEM_SEGMENT15               VARCHAR2(40)
,FROM_ITEM_SEGMENT16               VARCHAR2(40)
,FROM_ITEM_SEGMENT17               VARCHAR2(40)
,FROM_ITEM_SEGMENT18               VARCHAR2(40)
,FROM_ITEM_SEGMENT19               VARCHAR2(40)
,FROM_ITEM_SEGMENT20               VARCHAR2(40)
,TO_INVENTORY_ITEM_ID              NUMBER(15)
,TO_ITEM_SEGMENT1                  VARCHAR2(40)
,TO_ITEM_SEGMENT2                  VARCHAR2(40)
,TO_ITEM_SEGMENT3                  VARCHAR2(40)
,TO_ITEM_SEGMENT4                  VARCHAR2(40)
,TO_ITEM_SEGMENT5                  VARCHAR2(40)
,TO_ITEM_SEGMENT6                  VARCHAR2(40)
,TO_ITEM_SEGMENT7                  VARCHAR2(40)
,TO_ITEM_SEGMENT8                  VARCHAR2(40)
,TO_ITEM_SEGMENT9                  VARCHAR2(40)
,TO_ITEM_SEGMENT10                 VARCHAR2(40)
,TO_ITEM_SEGMENT11                 VARCHAR2(40)
,TO_ITEM_SEGMENT12                 VARCHAR2(40)
,TO_ITEM_SEGMENT13                 VARCHAR2(40)
,TO_ITEM_SEGMENT14                 VARCHAR2(40)
,TO_ITEM_SEGMENT15                 VARCHAR2(40)
,TO_ITEM_SEGMENT16                 VARCHAR2(40)
,TO_ITEM_SEGMENT17                 VARCHAR2(40)
,TO_ITEM_SEGMENT18                 VARCHAR2(40)
,TO_ITEM_SEGMENT19                 VARCHAR2(40)
,TO_ITEM_SEGMENT20                 VARCHAR2(40)
,FROM_CUST_TRX_LINE_ID             NUMBER(15)
,FROM_LINE_NUMBER                  NUMBER(15)
,TO_CUST_TRX_LINE_ID               NUMBER(15)
,TO_LINE_NUMBER                    NUMBER(15)
,GL_DATE                           DATE
,REASON_CODE                       VARCHAR2(15)
```

```
,COMMENTS                      VARCHAR2(2000)
,ATTRIBUTE_CATEGORY            VARCHAR2(30)
,ATTRIBUTE1                    VARCHAR2(150)
,ATTRIBUTE2                    VARCHAR2(150)
,ATTRIBUTE3                    VARCHAR2(150)
,ATTRIBUTE4                    VARCHAR2(150)
,ATTRIBUTE5                    VARCHAR2(150)
,ATTRIBUTE6                    VARCHAR2(150)
,ATTRIBUTE7                    VARCHAR2(150)
,ATTRIBUTE8                    VARCHAR2(150)
,ATTRIBUTE9                    VARCHAR2(150)
,ATTRIBUTE10                   VARCHAR2(150)
,ATTRIBUTE11                   VARCHAR2(150)
,ATTRIBUTE12                   VARCHAR2(150)
,ATTRIBUTE13                   VARCHAR2(150)
,ATTRIBUTE14                   VARCHAR2(150)
,ATTRIBUTE15                   VARCHAR2(150));
```

The following table lists standard API parameters that are common to all the routines in the Revenue Adjustment API.

Parameter	Type	Data-type	Required	Default Value	Description
p_api_version	IN	NUMBER	Yes		Used to compare version numbers of incoming calls to its current version number. Unexpected error is raised if version incompatibility exists. In the current version of the API, you should pass in a value of 1.0 for this parameter.
p_init_msg_list	IN	VARCHAR2		FND_API.G_ FALSE	Allows API callers to request that the API does initialization of the message list on their behalf.
p_commit	IN	VARCHAR2		FND_API.G_ FALSE	Used by API callers to ask the API to commit on their behalf.
p_rev_adj_rec	IN	AR_Revenue_ Adjustment_ PVT.Rev_Adj_ Rec_Type	Yes	See break-down below for individual elements	Revenue Adjustment record type
x_return_status	OUT	VARCHAR2			Represents the API overall return status. Detailed in *Return Status* on page 8-7.

Parameter	Type	Data-type	Required	Default Value	Description
x_msg_count	OUT	NUMBER			Number of messages in the API message list.
x_msg_data	OUT	VARCHAR2			This is the message in encoded format if x_msg_count=1.
x_adjustment_id	OUT	NUMBER			The ID of the resulting revenue adjustment.
x_adjustment_number	OUT	VARCHAR2			The user visible number of the resulting revenue adjustment.

The following table lists Rev_Adj_Rec_Type elements that are relevant to Unearn_Revenue:

Note: If required parameters are not passed in a call to this API, then the call will fail. However, depending on the business scenario, you will have to pass in values for other parameters to successfully create the business object. Otherwise, error messages will be reported.

At least one of the numbered sets of parameters is required.

Parameter	Data-type	Required	Description
p_customer_trx_id	NUMBER(15)	1	The ID of the transaction on which revenue is to be adjusted. Default: None Validation: Must exist if specified. Must not have a class of 'CB','DM','BR','DEP','GUAR' (i.e. chargeback, debit memo, bills receivable, deposit, guarantee). Must not have had credit memo(s) raised against the full transaction value. Warning if partial credit memo has been raised. Every line must have revenue sales credits adding to 100%. Errors: AR_TAPI_TRANS_NOT_EXIST AR_TW_INCORRECT_SALESCREDIT AR_RA_CB_DISALLOWED AR_RA_DM_DISALLOWED AR_RA_BR_DISALLOWED AR_RA_DEP_DISALLOWED AR_RA_GUAR_DISALLOWED AR_TW_INCORRECT_SALESCREDIT AR_RA_FULL_CREDIT Warnings: AR_RA_PARTIAL_CREDIT
trx_number	ra_customer_trx.trx_number%TYPE	1	The user visible transaction number Default: None Validation: Ignored if customer_trx_id has a value. Must be unique. Batch source can be optionally passed as extra assurance of uniqueness - then must be unique for that batch source. Otherwise, validation is the same as for customer_trx_id. Errors: AR_RA_TRX_NOTFOUND AR_RA_TRX_TOO_MANY_ROWS
batch_source_name	ra_batch_sources.name%TYPE		Name of the batch source associated with the trx_number, if specified. Only used in association with trx_number to help ensure uniqueness. Default: None Validation: Ignored if trx_number is not passed. If an invalid string is passed, the trx not found message will result.

Parameter	Data-type	Required	Description
adjustment_type	VARCHAR2(15)		Type of revenue adjustment. This element should be left null. Default: 'UN'
from_salesrep_id	NUMBER(15)		The ID of the salesperson whose revenue is being adjusted. Validation: If specified, must exist, must be currently active, and must have been active on transaction date. Must have revenue sales credits on at least one line on the transaction. Error: AR_TAPI_INVALID_SALESREP_ID AR_RA_SALESREP_NOT_ON_TRX
from_salesrep_number	ra_ salesreps.salesrep _number%TYPE		The user visible number of the salesperson whose revenue is being adjusted. Validation: Ignored if from_salesrep_id is specified. Otherwise, validation is as for from_ salesrep_id. Error: AR_RA_INAVLID_SALESREP_NUMBER
to_salesrep_id	NUMBER		Not used for unearning revenue and should be left null.
to_salesrep_number	VARCHAR2(30)		Not used for unearning revenue and should be left null.
from_salesgroup_id	jtf_rs_groups_b. group_id%TYPE		The ID of the sales group of the salesperson whose revenue is being adjusted. Validation: Must have revenue sales credits on at least one line on the transaction. Error: AR_RA_SALESREP_NOT_ON_TRX
to_salesgroup_id	jtf_rs_groups_b. group_id%TYPE		Not used for unearning revenue and should be left null.
sales_credit_type	VARCHAR2(15)		Not used for unearning revenue and should be left null.

Parameter	Data-type	Required	Description
amount_mode	VARCHAR2(15)		The amount mode specifies whether an amount, a percentage (of total value of selected lines), or all adjustable revenue is to be adjusted. Possible values are: T - total adjustable revenue A - amount P - percent Default: 'T' Validation: Must be one of the above values Error: AR_RA_INVALID_AMOUNT_MODE
amount	NUMBER		The amount of revenue to be adjusted Default: None Validation: Ignored unless amount_mode = 'A', in which case it must have a value. Must be =< total recognized revenue for selected lines, and salesperson (if specified). Errors: AR_RA_AMT_EXCEEDS_AVAIL_REV AR_RA_ZERO_AMOUNT
percent	NUMBER		The percentage of total selected transaction line value to be adjusted. Default: None Validation: Ignored unless amount_mode = 'P' in which case it must have a value. Must be =< percentage of total value of selected lines represented by recognized revenue for selected lines, and salesperson (if specified). Errors: AR_RA_PCT_EXCEEDS_AVAIL_PCT AR_RA_ZERO_AMOUNT

Parameter	Data-type	Required	Description
line_selection_mode	VARCHAR2(15)		The line selection mode determines how lines were selected for adjustment. Possible values are: A - All transaction lines C - Specific category I - Specific item S - Specific line Default: 'A' Validation: Must be one of the above values Error: AR_RA_INVALID_LINE_MODE
from_category_id	NUMBER(15)		The ID of the item category used to identify the lines to be adjusted. Default: None Validation: Must be a valid category ID, and there must be lines on the transaction that have items belonging to this category. Must be specified if line selection mode = 'C'. Errors: AR_RA_NO_FROM_CATEGORY AR_RA_INVALID_CATEGORY_ID AR_RA_CATEGORY_NOT_ON_TRX
from_category_ segment1 -from_ category_segment20	VARCHAR2(40)		Segments 1 to 20 of the category flexfield Default: None Validation: Ignored if from_category_id has a value. Enough segment values to uniquely identify a category must be passed - ideally all defined segments. Otherwise, validation is the same as for from_category_id. Error: AR_RA_INVALID_CAT_SEGMENTS
to_category_id	NUMBER(15)		Not currently used and should be left null.
to_category_segment1 -to_category_ segment20	VARCHAR2(40)		Not currently used and should be left null.

Parameter	Data-type	Required	Description
from_inventory_item_id	NUMBER(15)		The ID of the inventory item used to identify the lines to be adjusted. Default: None Validation: Must be a valid inventory item ID and there must be lines on the transaction that have items with this ID. Must be specified if line selection mode = 'I'. Errors: AR_RA_NO_FROM_ITEM AR_RA_INVALID_ITEM_ID AR_RA_ITEM_NOT_ON_TRX
from_item_segment1 -from_item_segment20	VARCHAR2(40)		Segments 1 to 20 of the item flexfield Default: None Validate: Ignored if from_inventory_item_id has a value. Enough segment values to uniquely identify an item must be passed - ideally all defined segments. Otherwise, validation is the same as for from_inventory_item_id. Error: AR_RA_INVALID_ITEM_SEGMENTS
to_inventory_item_id	NUMBER(15)		Not currently used and should be left null.
to_item_segment1 -to_item_segment20	VARCHAR2(40)		Not currently used and should be left null.
from_cust_trx_line_id	NUMBER(15)		The ID of the transaction line to be adjusted. Default: None Validation: Must be a valid line ID on the transaction. Must be specified if line selection mode = 'S' and from_line_number is null. Errors: AR_RA_NO_FROM_LINE AR_RA_INVALID_LINE_ID

Parameter	Data-type	Required	Description
from_line_number	NUMBER(15)		The user visible transaction line number. Default: None Validation: Ignored if from_cust_trx_line_id has a value. Must be a valid line number on the transaction. Errors: AR_RA_NO_FROM_LINE AR_RA_LINE_NOT_ON_TRX
gl_date	DATE		Date that adjusted revenue will be posted to the general ledger if revenue is recognized immediately. Start date of revenue recognition if revenue is deferred. Default: Gets defaulted to the current date if it is a valid gl_date. Validation: Ignored for lines that have non-deferred accounting rules AND a duration > 1. It is valid if the following conditions are true: ■ The date is in an Open or Future period, or it is in a Never Opened period and the Allow Not Open Flag is set to Yes. ■ The date is greater than or equal to the trx_ date ■ The period cannot be an Adjustment period. If the date passed is not valid, then a warning message is written to the stack and the date is automatically overridden with a valid date using the default: ■ If the most recent open period is prior to the current date: last date of that period ■ If there is a period open after the current date: first date of the last open period Warning: AR_RA_GL_DATE_CHANGED
reason_code	VARCHAR2(15)	Yes	Lookup code for revenue adjustment reason Default: None Validation: Must be defined under AR lookup type 'REV_ADJ_REASON' Error: AR_RA_INVALID_REASON_CODE

Parameter	Data-type	Required	Description
comments	VARCHAR2 (2000)		Free text Default: None Validation: None
attribute_category	VARCHAR2(30)		Context of the revenue adjustment descriptive flexfield. Default: None Validation: None
attribute1 - attribute15	VARCHAR2(150)		Attributes of the revenue adjustment descriptive flexfield Default: None Validation: Standard descriptive flexfield validation

Example

Objective:

To unearn all revenue on a transaction using a call to *AR_RevenueAdjust_PUB.Unearn_Revenue* and passing a minimum number of input parameters.

This table lists the entered parameters:

Parameter	Entered Value	Default Value
p_api_version	2.0	
p_init_msg_list	FND_API.G_TRUE	
p_rev_adj_rec.trx_number	'test_api_1'	
p_rev_adj_rec.reason_code	'RA'	

This table lists the defaulted input parameters, which were not entered:

Parameter	Entered Value	Default Value
p_rev_adj_ rec.amount_mode		'T'
p_rev_adj_rec.line_ selection_mode		'A'
p_rev_adj_rec.gl_ date		SYSDATE

The API call in this case would be:

```
AR_RevenueAdjust_PUB.Unearn_Revenue(
    p_api_version => 2.0,
    p_init_msg_list    => FND_API.G_TRUE,
    p_rev_adj_rec.trx_number  => 'test_api_1',
    p_rev_adj_rec.reason_code => 'RA',
    x_return_status    => l_return_status,
    x_msg_count    => l_msg_count,
    x_msg_data    => l_msg_data,
    x_adjustment_id    => l_adjustment_id,
    x_adjustment_number       => l_adjustment_number);
```

After execution of this API, the calling program retrieves the warnings and the error messages, put on the message stack by the API, in the following manner:

The warnings and the error messages put on the message stack by the API are retrieved after the execution of this API by the calling program, in the following manner:

```
  IF l_msg_count  = 1 Then
    --there is one message raised by the API, so it has been sent out
    --in the parameter x_msg_data, get it.
   l_msg_data_out := l_msg_data;
  ELSIF l_msg_count  > 1 Then
  --the messages on the stack are more than one so call them in a loop
  -- and put the messages in a PL/SQL table.
     loop
      count := count +1 ;
      l_mesg := FND_MSG_PUB.Get;
       If l_mesg  IS NULL Then
        EXIT;
      else
```

```
      Mesg_tbl(count).message := l_mesg;
    End if;
  end loop;
END IF;
```

Depending on the message level threshold set by the profile option FND_API_MSG_LEVEL_THRESHOLD, the messages put on the message stack may contain both the error messages and the warnings.

Result:

All revenue on this transaction was unearned by specifying only four input parameters in the call to this API.

AR_RevenueAdjust_PUB.Earn_Revenue

Description

Call this routine to move revenue from the unearned revenue account to the earned revenue account using AutoAccounting. This API routine has 4 input and 5 output parameters in total and is almost exactly the same as the Unearn_Revenue routine described above in *Description* on page 8-10.

The following is the breakdown of the parameters:

Input

Standard API parameters: 3

Revenue Adjustment parameters: 1 (revenue adjustment record type)

Output

Standard API parameters: 3

Revenue Adjustment parameters: 2

The description of the standard API parameters is the same as that already given in *Description* on page 8-10.

The Rev_Adj_Rec_Type elements that are relevant to Earn_Revenue are exactly the same as already listed in *AR_RevenueAdjust_PUB.Unearn_Revenue* on page 8-10, with the following exceptions listed in this table:

Parameter	Data-type	Required	Description
to_salesrep_id	NUMBER		Not used for earning revenue and should be left null.
to_salesrep_ number	VARCHAR2		Not used for earning revenue and should be left null.
to_salesgroup_id	jtf_rs_groups_b. group_ id%TYPE		Not used for earning revenue and should be left null.
sales_credit_type	VARCHAR2(15)		Not used for earning revenue and should be left null.

Example

Objective:

To earn all revenue on a transaction using a call to *AR_RevenueAdjust_PUB.Earn_ Revenue* and passing a minimum number of input parameters.

This table lists the entered parameters:

Parameter	Entered Value	Default Value
p_api_version	2.0	
p_init_msg_list	FND_API.G_TRUE	
p_rev_adj_rec.trx_ number	'test_api_1'	
p_rev_adj_rec.reason_ code	'RA'	

This table lists the defaulted input parameters, which were not entered:

Parameter	Entered Value	Default Value
p_rev_adj_ rec.amount_mode		'T'
p_rev_adj_rec.line_ selection_mode		'A'
p_rev_adj_rec.gl_date		SYSDATE

The API call in this case would be:

```
AR_RevenueAdjust_PUB.Earn_Revenue(
    p_api_version    => 2.0,
    p_init_msg_list    => FND_API.G_TRUE,
    p_rev_adj_rec.trx_number  => 'test_api_1',
    p_rev_adj_rec.reason_code => 'RA',
    x_return_status    => l_return_status,
    x_msg_count    => l_msg_count,
    x_msg_data    => l_msg_data,
    x_adjustment_id    => l_adjustment_id,
    x_adjustment_number       => l_adjustment_number);
```

The warnings and the error messages put on the message stack by the API are retrieved after the execution of this API by the calling program, as described in *Example* on page 8-21.

Result:

All revenue on this transaction was earned by specifying only four input parameters in the call to this API.

AR_RevenueAdjust_PUB.Transfer_Sales_Credits

Description

Call this routine to transfer sales credits from any salesperson with sales credits on the transaction to any other salesperson. In addition, if revenue sales credits are transferred, then the associated revenue is transferred between cost centers if the AutoAccounting rules call the salesperson table and the cost center segment is derived from the salesperson.

This API routine has 4 input and 5 output parameters in total and is similar to the Unearn_Revenue routine as described in *Description* on page 8-10. The following is the breakdown of the parameters:

Input

Standard API parameters: 3

Revenue Adjustment parameters: 1 (revenue adjustment record type)

Output

Standard API parameters: 3

Revenue Adjustment parameters: 2

The description of the standard API parameters is the same as in *Description* on page 8-10.

The Rev_Adj_Rec_Type elements that are relevant to Transfer_Sales_Credits are the same as already listed in *AR_RevenueAdjust_PUB.Unearn_Revenue* on page 8-10, with the following exceptions/additions listed in this table.

> **Note:** If required parameters are not passed in a call to this API, then the call will fail. However, depending on the business scenario, you will have to pass in values for other parameters to successfully create the business object. Otherwise, error messages will be reported.

Parameter	Data-type	Required	Description
from_salesrep_id	NUMBER(15)		The ID of the salesperson from whom sales credits are being transferred. Default: Null Validation: If specified, must exist, must be currently active, and must have been active on transaction date. Must have revenue sales credits on at least one line on the transaction. If neither from_salesrep_id nor from_salesrep_number are specified, sales credits of the specified type are transferred belonging to all salesreps on the transaction (i.e. null = all). Error: AR_TAPI_INVALID_SALESREP_ID AR_RA_SALESREP_NOT_ON_TRX
from_salesrep_number	ra_ salesreps.salesrep_ number%TYPE		The user visible number of the salesperson from whom sales credits are being transferred. Validation: Ignored if from_salesrep_id is specified. Otherwise, validation is as for from_ salesrep_id. Error: AR_RA_INVALID_SALESREP_NUMBER

Parameter	Data-type	Required	Description
to_salesrep_id	NUMBER(15)	2	The ID of the salesperson to whom sales credits are being transferred. Validation: If specified, must exist, and must be currently active and must have been active on transaction date. Errors: AR_TAPI_INVALID_SALESREP_ID AR_RA_NO_TO_SALESREP
to_salesrep_number	ra_ salesreps.salesrep_ number%TYPE	2	The user visible number of the salesperson to whom sales credits are being transferred. Validation: Ignored if to_salesrep_id is specified. Otherwise, validation is as for to_salesrep_id. Error: AR_RA_INVALID_SALESREP_NUMBER
from_salesgroup_id	jtf_rs_groups_b. group_id%TYPE		The ID of the sales group of the salesperson from whom sales credits are being transferred. Default: Null Validation: Must have sales credits (of the type being transferred) on at least one line on the transaction. If FROM_SALESGROUP_ID is not specified, then all sales credits of the specified type for the chosen salesperson are transferred (ie. null = all). Error: AR_RA_SALESREP_NOT_ON_TRX
to_salesgroup_id	jtf_rs_groups_ b.group_id%TYPE		The ID of the sales group of the salesperson to whom sales credits are being transferred. Validation: If specified, then must exist and must be currently active. Salesperson must have been an active member of this group at some time between: --the earliest of the transaction date and any parent commitment/invoice dates, and --the latest of the current date, transaction date, and any parent commitment/invoice dates. Error: AR_INVALID_SALESGROUP_ID

Parameter	Data-type	Required	Description
sales_credit_type	VARCHAR2(15)	Yes	The type of sales credit being transferred. Possible values: R = revenue sales credits N = non-revenue sales credits B = both Default: 'R' Validation: Must be one of the above values. **Note:** The value B cannot be used if either FROM_SALESGROUP_ID or TO_SALESGROUP_ID is specified. Error: AR_INCOMPATIBLE_CREDIT_TYPE AR_RA_INVALID_SALESCRED_TYPE

Example

Objective:

To transfer all revenue sales credits on a transaction from a salesperson to a new salesperson using a call to *AR_RevenueAdjust_PUB.Transfer_Sales_Credits* and passing a minimum number of input parameters.

This table lists the entered parameters:

Parameter	Entered Value	Default Value
p_api_version	2.0	
p_init_msg_list	FND_API.G_TRUE	
p_rev_adj_rec.trx_number	'test_api_1'	
p_rev_adj_rec.from_salesrep_number	'101'	
p_rev_adj_rec.to_salesrep_number	'299'	
p_rev_adj_rec.reason_code	'RA'	

This table lists the defaulted input parameters, which were not entered:

Parameter	Entered Value	Default Value
p_rev_adj_rec.amount_ mode		'T'
p_rev_adj_rec.sales_credit_ type		'R'
p_rev_adj_rec.line_ selection_mode		'A'
p_rev_adj_rec.gl_date		SYSDATE

The API call in this case would be:

```
AR_RevenueAdjust_PUB.Transfer_Sales_Credits(
    p_api_version    => 2.0,
    p_init_msg_list    => FND_API.G_TRUE,
    p_rev_adj_rec.trx_number  => 'test_api_1',
    p_rev_adj_rec.from_salesrep_number => '101',
    p_rev_adj_rec.to_salesrep_number => '299'
    p_rev_adj_rec.reason_code => 'RA',
    x_return_status    => l_return_status,
    x_msg_count    => l_msg_count,
    x_msg_data    => l_msg_data,
    x_adjustment_id    => l_adjustment_id,
    x_adjustment_number       => l_adjustment_number);
```

The warnings and the error messages put on the message stack by the API are retrieved after execution of this API by the calling program, as described in *Example* on page 8-21.

Result:

All revenue sales credits on this transaction belonging to salesperson 101 were transferred to salesperson 299 by specifying only six input parameters in the call to this API. Additionally, all associated revenue was transferred between corresponding cost centers. Note that if salesrep number 101 was the only salesperson with revenue sales credits on this transaction, then from_salesrep_ number could have been omitted. This is because no specified salesperson means *all* salespersons, thereby cutting the required number of parameters to five.

AR_RevenueAdjust_PUB.Add_Non_Revenue_Sales_Credits

Description

Call this routine to add non-revenue sales credits to any existing or new salesperson on a transaction. This does not involve a transfer of revenue. This API routine has 4 input and 5 output parameters in total and is similar to the Unearn_Revenue routine described in *AR_RevenueAdjust_PUB.Unearn_Revenue* on page 8-10.

The following is the breakdown of the parameters:

Input

Standard API parameters: 3

Revenue Adjustment parameters: 1 (revenue adjustment record type)

Output

Standard API parameters: 3

Revenue Adjustment parameters: 2

The description of the standard API parameters is the same as already given in *AR_RevenueAdjust_PUB.Unearn_Revenue* on page 8-10.

The Rev_Adj_Rec_Type elements that are relevant to Add_Non_Revenue_Sales_Credits are the same as already listed in *AR_RevenueAdjust_PUB.Unearn_Revenue* on page 8-10, with the following exceptions/additions listed in this table:

> **Note:** If required parameters are not passed in a call to this API, then the call will fail. However, depending on the business scenario, you will have to pass in values for other parameters to successfully create the business object. Otherwise, error messages will be reported.
>
> At least one of the numbered sets of parameters is required.

Parameter	Data-type	Required	Description
from_salesrep_id	NUMBER(15)		Not applicable in this context and should be left null.
from_salesrep_number	ra_salesreps.salesrep_number%TYPE		Not applicable in this context and should be left null.

Parameter	Data-type	Required	Description
to_salesrep_id	NUMBER(15)	2	The ID of the salesperson to whom non-revenue sales credits are being added. Validation: If specified, must exist, and must be currently active and must have been active on transaction date. Errors: AR_TAPI_INVALID_SALESREP_ID AR_RA_NO_TO_SALESREP
to_salesrep_number	ra_salesreps.salesrep_number%TYPE	2	The user visible number of the salesperson to whom sales credits are being transferred. Validation: Ignored if to_salesrep_id is specified. Otherwise, validation is as for to_salesrep_id. Error: AR_RA_INVALID_SALESREP_NUMBER
from_salesgroup_id	jtf_rs_groups_b.group_id%TYPE		Not applicable in this context and should be left null.
to_salesgroup_id	jtf_rs_groups_b.group_id%TYPE		The ID of the sales group of the salesperson to whom nonrevenue sales credits are being added. Validation: If specified, then must exist and must be currently active. Salesperson must have been an active member of this group at some time between: --the earliest of the transaction date and any parent commitment/invoice dates, and --the latest of the current date, transaction date, and any parent commitment/invoice dates. Error: AR_INVALID_SALESGROUP_ID
sales_credit_type	VARCHAR2(15)		Not applicable in this context and should be left null.
amount_mode	VARCHAR2(15)		The amount mode specifies whether an amount, a percentage (of total value of selected lines) is to be adjusted. Possible values are: A - amount P - percent Default: 'T', or all adjustable revenue is not applicable in this context. Validation: Must be one of the above values (A or P). Error: AR_RA_INVALID_AMOUNT_MODE

Example

Objective:

To add 50% of the total transaction value in non-revenue sales credits to a new salesperson on a transaction, using a call to AR_RevenueAdjust_PUB.Add_Non_Revenue_Sales_Credits and passing a minimum number of input parameters.

This table lists the entered parameters:

Parameter	Entered Value	Default Value
p_api_version	2.0	
p_init_msg_list	FND_API.G_TRUE	
p_rev_adj_rec.trx_number	'test_api_1'	
p_rev_adj_rec.to_salesrep_number	'299'	
p_rev_adj_rec.amount_mode	'P'	
p_rev_adj_rec.percent	50	
p_rev_adj_rec.reason_code	'RA'	

This table lists the defaulted input parameters, which were not entered:

Parameter	Entered Value	Default Value
p_rev_adj_rec.line_selection_mode		'A'
p_rev_adj_rec.gl_date		SYSDATE

The API call in this case would be:

```
AR_RevenueAdjust_PUB.Add_Non_Revenue_Sales_Credits(
    p_api_version    => 2.0,
    p_init_msg_list    => FND_API.G_TRUE,
    p_rev_adj_rec.trx_number  => 'test_api_1',
    p_rev_adj_rec.to_salesrep_number => '299'
    p_rev_adj_rec.amount_mode => 'P',
    p_rev_adj_rec.percent     => 50,
```

```
p_rev_adj_rec.reason_code => 'RA',
x_return_status    => l_return_status,
x_msg_count    => l_msg_count,
x_msg_data    => l_msg_data,
x_adjustment_id    => l_adjustment_id,
x_adjustment_number       => l_adjustment_number);
```

The warnings and the error messages put on the message stack by the API are retrieved after execution of this API by the calling program, as described in *Example* on page 8-21.

Result:

Non-revenue sales credits were added to salesperson 299 on this transaction by specifying only seven input parameters in the call to this API.

Messages

Messages play an important role in the effectiveness of API calls. The right message is raised at the right point to convey the exact error that has occurred or any warnings that have been raised. In the Revenue Adjustment API, all error messages and warnings raised during execution are put on the message stack and can be retrieved by the user as described in *Exception Handling and Result Messages* on page 8-7.

WARNINGS AND ERRORS

The following table lists all the error messages raised by the Revenue Adjustment API:

TYPE

E: Error message

W: Warning message

MESSAGE CODE	MESSAGE TEXT	DESCRIPTION	TYPE
AR_INCOMPATIBLE_ CREDIT_TYPE	The option of transferring "both" sales credit types is not available in conjunction with sales group transfers.		E
AR_INVALID_ SALESGROUP_ID	Please provide a valid sales group ID for sales credit transfers or additions.		E
AR_RA_AMT_EXCEEDS_ AVAIL_REV	The amount entered is greater than &TOT_AVAIL_REV, the total available revenue on the lines selected	This message is generated by the revenue adjustment API when there is insufficient adjustable revenue on the selected transaction lines to meet the specified amount.	E
AR_RA_BR_DISALLOWED	Revenue cannot be adjusted on bills receivable		E
AR_RA_CATEGORY_NOT_ ON_TRX	There are no lines with items for category ID &CATEGORY_ID on this transaction.		E
AR_RA_CB_DISALLOWED	Revenue cannot be adjusted on chargebacks		E

MESSAGE CODE	MESSAGE TEXT	DESCRIPTION	TYPE
AR_RA_DEP_ DISALLOWED	Revenue cannot be adjusted on deposits.		E
AR_RA_DM_ DISALLOWED	Revenue cannot be adjusted on debit memos or debit memo reversals		E
AR_RA_FULL_CREDIT	One or more credit memos have been applied for the full amount of this invoice		E
AR_RA_GL_DATE_ CHANGED	GL date, &GL_DATE, is not in an open or future-enterable period. GL date has been changed to &NEW_GL_DATE		W
AR_RA_GUAR_ DISALLOWED	Revenue cannot be adjusted on guarantees.		E
AR_RA_INVALID_ AMOUNT_MODE	Amount mode &AMOUNT_ MODE is invalid.		E
AR_RA_INVALID_CAT_ SEGMENTS	This combination of category segments is invalid: &CONCAT_ SEGS.		E
AR_RA_INVALID_ CATEGORY	A valid category to which items belong that are currently on one or more lines on this transaction must be entered		E
AR_RA_INVALID_ CATEGORY_ID	Category ID &CATEGORY_ID is invalid.		E
AR_RA_INVALID_CODE_ COMB	An error occurred while generating the following accounting flexfield code combination: &CODE_ COMBINATION	This message is generated by the revenue adjustment API because of an error with the specified accounting flexfield code combination. Possible causes: segment values could not be found by AutoAccounting or have been disabled.	E
AR_RA_INVALID_ITEM	A valid item that is currently on one or more lines on this transaction must be entered		E
AR_RA_INVALID_ITEM_ ID	Inventory item ID &ITEM_ID is invalid.		E

MESSAGE CODE	MESSAGE TEXT	DESCRIPTION	TYPE
AR_RA_INVALID_ITEM_SEGMENTS	This combination of item segments is invalid: &CONCAT_SEGS.		E
AR_RA_INVALID_LINE_ID	Transaction line ID &CUST_TRX_LINE_ID is invalid.		E
AR_RA_INVALID_LINE_MODE	Line selection mode &LINE_MODE is invalid.		E
AR_RA_INVALID_REASON	Reason code &REASON_CODE is not a valid lookup code.		E
AR_RA_INVALID_SALESCRED_TYPE	Sales credit type &SALESCRED_TYPE is invalid.		E
AR_RA_INVALID_SALESREP_NUMBER	Salesperson number &SALESREP_NUMBER is invalid.		E
AR_RA_ITEM_NOT_ON_TRX	There are no lines with item &ITEM_ID on this transaction.		E
AR_RA_LINE_NOT_ON_TRX	There are no lines with line number &LINE_NUMBER on this transaction.		E
AR_RA_NO_EARNED_REVENUE	There is no earned revenue on this transaction	This message is generated by the revenue adjustment API when there is no earned revenue on the selected transaction lines.	E
AR_RA_NO_FROM_CATEGORY	Please provide a from-category.		E
AR_RA_NO_FROM_ITEM	Please provide a from-item.		E
AR_RA_NO_FROM_LINE	Please provide a from-line.		E
AR_RA_NO_OPEN_PERIODS	The transaction date must fall during an open period or prior to a future period	This message is generated by the revenue adjustment API because there are no open or future periods relating to the transaction date or following the transaction date. Revenue cannot be posted to periods prior to the transaction date.	E
AR_RA_NO_REV_SALES_CREDIT	Line &LINE_NUMBER has no revenue sales credits	This message is generated by the revenue adjustment API when a transaction line with no sales credits is encountered.	E

MESSAGE CODE	MESSAGE TEXT	DESCRIPTION	TYPE
AR_RA_NO_REV_TO_ ADJUST	There is no adjustable revenue on the selected lines	This message is generated by the revenue adjustment API when there is no adjustable revenue on the selected transaction lines.	E
AR_RA_NO_SELECTED_ SALESCRED	There are no sales credits for this line selection available to transfer		E
AR_RA_NO_TO_ SALESREP	Please provide a valid salesperson number or ID for sales credit transfers or additions.		E
AR_RA_NO_TRX_ NUMBER	Please provide a valid transaction number or ID.		E
AR_RA_NO_UNEARNED_ REVENUE	There is no unearned revenue on this transaction	This message is generated by the revenue adjustment API when there is no unearned revenue on the selected transaction lines.	E
AR_RA_PARTIAL_CREDIT	One or more partial credit memos have been applied against this invoice		W
AR_RA_PCT_EXCEEDS_ AVAIL_PCT	The percentage entered is greater than &TOT_AVAIL_PCT, the total available percentage of adjustable revenue on the lines selected	This message is generated by the revenue adjustment API when there is insufficient adjustable revenue on the selected transaction lines to meet the specified percentage.	E
AR_RA_SALES_CREDIT_ LIMIT	Revenue and non-revenue sales credits exceed &SALES_CREDIT_ LIMIT percent for salesperson &SALESREP_NAME on line &LINE_NUMBER	This message is generated by the revenue adjustment API when the total percentage of revenue and non-revenue sales credits per salesperson per line exceeds the limit specified in system options.	E
AR_RA_SALESREP_NOT_ ON_TRX	Salesperson &SALESREP_NAME does not have any sales credits on this transaction.		E
AR_RA_TRX_NOTFOUND	Transaction number &TRX_ NUMBER cannot be found.		E
AR_RA_TRX_TOO_ MANY_ROWS	There is more than one transaction with the transaction number &TRX_NUMBER. Please also provide a batch source to ensure uniqueness of the transaction.		E

MESSAGE CODE	MESSAGE TEXT	DESCRIPTION	TYPE
AR_RA_ZERO_AMOUNT	Amount entered cannot be zero	This message is generated by the revenue accounting API when attempting to adjust an amount of zero.	E
AR_RAPI_DESC_FLEX_ INVALID	The entered values for the descriptive flexfield &DFF_ NAME is invalid.		E
AR_TW_INCORRECT_ SALESCREDIT	Revenue sales credit not equal to line amount or 100% for line &LINE_NUMBER.		E
AR_TAPI_TRANS_NOT_ EXIST	Transaction does not exist. (CUSTOMER_TRX_ID: &CUSTOMER_TRX_ID).		E
AR_TAPI_INVALID_ SALESREP_ID	Invalid salesrep id. (SALESREP_ ID: &SALESREP_ID)		E