

How we applied On-Account Credit Memo's to Invoices using AR_RECEIPT_API_PUB

Author: Anil Patil

Created: May 11, 2007

Product: Oracle Receivables

Overview

Automation of manual activities is the norm and that's where Oracle Standard API's helps me a lot. The Business scenario for my client is as follows

1. The Client imports feeder system transactions data into Oracle Receivables using RA INTERFACE
2. Credit Memo's are imported as On-Account Credit Memo's because there is not enough matching data available in the feeder system that would allow creating Credit transactions against the Invoices through autoinvoice.
3. The Credit Memo number and the related transaction number are maintained in a custom staging table.
4. The requirement was to be able to apply these On-Account Credit Memo's to the Invoices.

Our Solution

If the users had to perform this activity manually, they would query the Credit Memo in the Transactions form. Navigate to Menu > Actions > Applications and it would open up the Applications form. The users would then select the Invoice number in the 'Apply-To' field and save the application. This would apply the Credit Memo to the Invoice.

To automate the same steps manually, there is no standard transaction API.

(Enhancement Request [6036657](#) has been raised).

We used the following workaround to achieve the same result

1. Create a Zero amount Receipt using AR_RECEIPT_API_PUB.CREATE_CASH API
2. Apply the Receipt to the Credit Memo using AR_RECEIPT_API_PUB.APPLY API
3. Apply the Receipt to the Invoice using AR_RECEIPT_API_PUB.APPLY API

We considered the following validations in our program. These validations were specific to our Business requirements

1. The Credit Memo and the Invoice are both in the same currency.
2. There is no overapplication allowed
3. When the receipt is applied to the Invoice and the Credit Memo, the amount applied is the least of the Invoice balance amount or the Credit Memo balance amount.

Query to check if Invoice currency and CM currency is same or not

```

SELECT 'Y'
INTO l_flag
FROM ar_payment_schedules a , ar_payment_schedules b
WHERE a.customer_trx_id = l_inv_trx_id
AND b.customer_trx_id = l_cm_trx_id
AND a.invoice_currency_code = b.invoice_currency_code ;

```

l_inv_trx_id is the customer_trx_id of the invoice

l_cm_trx_id is the customer_trx_id of the credit memo

If the value of l_flag is NULL, then the invoice currency and the cm currency is not same.

Query to retrieve the balance amount of the transaction

```

-- INV BALANCE
SELECT acctd_amount_due_remaining
INTO l_inv_bal_amt
FROM ar_payment_schedules
Where customer_trx_id = l_inv_trx_id ;

-- CREDIT MEMO BALANCE
SELECT acctd_amount_due_remaining
INTO l_cm_bal_amt
FROM ar_payment_schedules
Where customer_trx_id = l_cm_trx_id ;

-- Amount applied should be the least of inv balance and cm balance
SELECT LEAST(l_inv_bal_amt, abs(l_cm_bal_amt))
INTO l_amount_applied
FROM DUAL;

```

Query to retrieve the customer_id of the transaction

```

SELECT customer_id
INTO l_customer_id
FROM ar_payment_schedules
WHERE customer_trx_id = l_inv_trx_id ;

```

At this point of time, we know the customer_trx_id , customer_id of the Invoice and the Credit Memo and we also know the amount to be applied. The next important thing I need is the receipt_method_id that I would use for creating the receipt.

The validations for the receipt_method_id is as follows

1. It must be a valid receipt method ID in the AR_RECEIPT_METHOD table.
2. Receipt date must lie between the receipt method start date and end date (if not null).
3. 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.

I am now ready to give calls to the AR_RECEIPT_API_PUB API

STEP 1 > Create a ZERO amount Receipt

To create a ZERO amount receipt, I use the AR_RECEIPT_API_PUB.CREATE_CASH API

```
AR_RECEIPT_API_PUB.CREATE_CASH(
  p_api_version => 1.0,
  p_init_msg_list => FND_API.G_TRUE,
  p_commit => FND_API.G_TRUE,
  p_receipt_number => 'TEST98',
  p_amount => 0,
  p_receipt_method_id => l_receipt_method_id,
  p_customer_id => l_customer_id ,
  p_cr_id => l_cr_id,
  x_return_status => l_return_status,
  x_msg_count => l_msg_count,
  x_msg_data => l_msg_data);
```

In the above code,

Parameter	Description
p_api_version	Always pass 1.0 for this parameter
p_init_msg_list	If set to FND_API.G_TRUE, the API does initialization of the message list.
p_commit	Used to specify if the API should commit.
p_receipt_number	The receipt number of the ZERO amount receipt
p_amount	The amount of the receipt. In our case, it is 0
p_receipt_method_id	Identifies the payment method of the receipt.
p_customer_id	The customer_id of the Customer.
p_cr_id	This is an OUT parameter. It is the cash_receipt_id of the receipt created by the API
x_return_status	OUT parameter. Return Status of the API Call
x_msg_count	OUT parameter. Number of messages in the API message list
x_msg_data	OUT parameter. Message in encoded format if x_msg_count = 1

STEP 2 > Apply the Receipt to the Credit Memo

To apply the Receipt to the Credit Memo, I use the AR_RECEIPT_API_PUB.APPLY API

```

AR_RECEIPT_API_PUB.APPLY(
    p_api_version => 1.0,
    p_init_msg_list => FND_API.G_TRUE,
    p_commit => FND_API.G_TRUE,
    p_receipt_number => 'TEST98',
    p_customer_trx_id => l_cm_trx_id ,
    p_amount_applied => -1 * (l_amount_applied) ,
    x_return_status => l_return_status,
    x_msg_count => l_msg_count,
    x_msg_data => l_msg_data
);

```

In the above code,

Parameter	Description
p_customer_trx_id	Customer_trx_id of the Credit Memo
p_amount_applied	The amount to which the receipt is to be applied. In our case we are applying the receipt to the Credit Memo and hence the amount_applied must be negative. We have calculated the l_amount_applied in one of the queries above

STEP 3 > Apply the Receipt to the Invoice

To apply the Receipt to the Invoice, I again use the AR_RECEIPT_API_PUB.APPLY API

```

AR_RECEIPT_API_PUB.APPLY(
    p_api_version => 1.0,
    p_init_msg_list => FND_API.G_TRUE,
    p_commit => FND_API.G_TRUE,
    p_receipt_number => 'TEST98',
    p_customer_trx_id => l_inv_trx_id ,
    p_amount_applied => l_amount_applied ,
    x_return_status => l_return_status,
    x_msg_count => l_msg_count,
    x_msg_data => l_msg_data
);

```

In the above code,

Parameter	Description
p_customer_trx_id	Customer_trx_id of the Invoice
p_amount_applied	The amount to which the receipt is to be applied. In our case we are applying the receipt to the Invoice and hence the amount_applied must be positive. We have calculated the l_amount_applied in one of the queries above

Summary

This document details our approach to apply On-Account Credit Memo's to Invoices using the AR_RECEIPT_API_PUB API. I have really found this API very useful. This process has helped us to save a lots of Users time, since this activity was earlier been performed manually. After all – Time Saved is Money Earned !!!

References

Oracle Receivables API User Notes

Oracle Receivables User Guide