

# **iTransact Gateway Recurring Billing Guide**

---

# **iTransact Gateway Recurring Billing Guide**

---

---

---

---

## Table of Contents

1. Version and Legal Information .....	1
2. The Recurring Billing System .....	2
3. Setting Up Recurring Recipes .....	3
The Recurring Transaction Window .....	3
Considerations .....	3
The Recurring Recipe Builder .....	4
The Recurring Help Window .....	7
4. Setting Transactions To Recur .....	8
Setting Transactions To Recur .....	8
Automatic Recurring Activation Using HTML .....	8
Automatic Recurring Activation Using XML .....	8
Manual Recurring Activation .....	8
Recurring Activation In The Virtual Terminal .....	12
Modifying Recurring Transaction Information Through The Transaction Listing ...	19
Modifying Recurring Transaction Information Using XML .....	28
Placing Recurring Transactions On Hold and Off Hold .....	30
Canceling Recurring Transactions .....	33
Setting Scheduled Transactions .....	37
How Does The Customer Recurring Billing Update Tool Work? .....	40
Customer Recurring Billing Update Tool Details .....	42
Activating These Features .....	43
How Can I Run A Form-Based Recurring Transaction Update? .....	43
Introduction .....	43
Process Overview .....	43
Module Interface .....	45
Payload Signature Generation .....	45
Input Fields For Form-Based Recurring Billing Update Request .....	45
Recurring Billing Update Responses .....	48
5. The Recurring Transaction History .....	52
Viewing The Transaction History .....	52
6. The Recurring PostBack Feature .....	54
Recurring PostBack Feature .....	54

---

## List of Figures

3.1. Recurring Transaction Recipe Window Example .....	3
3.2. Recurring Recipe Builder Example .....	4
4.1. Transaction Listing Example .....	9
4.2. Transaction Detail Access Window Example .....	9
4.3. Transaction Detail Example .....	10
4.4. Recurring Detail Example: Non-Recurring Transaction .....	10
4.5. Recurring Setup Information Page Example .....	11
4.6. Recurring Virtual Terminal Welcome Section Example .....	12
4.7. Recurring Virtual Terminal Transaction Information Section Example .....	13
4.8. Standard Virtual Terminal Payment Section Example .....	14
4.9. Standard Virtual Terminal Recurring Section Example .....	16
4.10. Standard Virtual Terminal Information Section Example .....	17
4.11. Approval Page Example .....	18
4.12. Transaction Listing Example .....	19
4.13. Transaction Detail Access Window Example .....	19
4.14. Transaction Detail Example .....	20
4.15. Recurring Detail Example: Recurring Transaction .....	22
4.16. Recurring User Info Editing Interface Example .....	23
4.17. Recurring User Info Changed Example .....	25
4.18. Successful Update Page Example .....	25
4.19. Recurring Items Editing Interface Example .....	26
4.20. Recurring Items Changed Example .....	27
4.21. Transaction Listing Example .....	30
4.22. Transaction Detail Access Window Example .....	30
4.23. Transaction Detail Example .....	31
4.24. Recurring Detail Example: Recurring Transaction .....	32
4.25. Transaction Listing Example .....	33
4.26. Transaction Detail Access Window Example .....	34
4.27. Transaction Detail Example .....	34
4.28. Recurring Detail Example: Recurring Transaction .....	35
4.29. Recurring Setup Edit Page Example: Recurring Transaction .....	36
4.30. Recurring Transaction Recipe Window Example .....	37
4.31. Scheduling Tool Example .....	38
4.32. Scheduling Calendar Tool Example .....	39
4.33. Scheduling Tool Example .....	39
4.34. Successful Schedule Example .....	40
4.35. Customer Recurring Billing Update Verification Example .....	40
4.36. Customer Recurring Billing Update Interface Example .....	41
5.1. Recurring Transaction Recipe Window Example .....	52
5.2. Recurring Recipe History Example .....	52
5.3. Recurring Transaction History Example .....	53

---

## List of Tables

4.1. HTML Recurring Example .....	8
4.2. XML Recurring Example .....	8
4.3. XMLTrans2.cgi RecurUpdate Example .....	28
4.4. XMLTrans2.cgi RecurUpdateResponse Example .....	29
4.5. XMLTrans2.cgi RecurDetails Example .....	29
4.6. XMLTrans2.cgi RecurDetailsResponse Example .....	30
4.7. Recurring Billing Update Input Fields .....	45
4.8. Passback[] Examples .....	47
4.9. Form-Based Recurring Billing Update Example - Card .....	47
4.10. Form-Based Recurring Billing Update Example - Check .....	48
4.11. Standard Recurring Billing Update Response Fields .....	48
4.12. Potential AVS_Category Values - Recurring Billing Update Response Values .....	50
4.13. Field Validation Error Indicators - Recurring Billing Update Response Fields .....	50

---

# Chapter 1. Version and Legal Information

**iTransact Gateway Recurring Billing Guide**

*iTransact Gateway Recurring Billing Guide*

Version: *1.9*

Date: *7/26/11*

Copyright: *iTransact, Inc. 2011*

---

# Chapter 2. The Recurring Billing System

This interface allows a merchant access to one of the most robust recurring billing features available on the Internet. This is an ideal tool for merchants who bill according to a subscription or according to a set schedule. This Recurring Billing tool automates ongoing billings in a simple manner which gives a merchant as much control as if the merchant was manually entering each of the transactions - without the hassle of manually entering the transactions.



# Chapter 3. Setting Up Recurring Recipes

## The Recurring Transaction Window

The window can be accessed from the Recurring Transaction link in the Control Panel, the "List Recipes" link in the Recipe Builder, and from the "View/Select Recipe" button in the Recurring Detail interface of any transaction (See Figure 3.1).

Figure 3.1. Recurring Transaction Recipe Window Example

RECURRING TRANSACTION RECIPES					
<a href="#">Add Recipe</a> <a href="#">Recurring Help</a>					
EDIT RECIPE	RECIPE NAME	CREATED	DEFINITION	HISTORY	SCHEDULE
<input type="button" value="go"/>	1stday	1/13/2010 13:54:27	Repeat every month on the 1st day	<input type="button" value="go"/>	
<input type="button" value="go"/>	Bimonthly15	1/13/2010 13:55:22	Repeat every 2 months on the 15th day	<input type="button" value="go"/>	
<input type="button" value="go"/>	Daily	1/13/2010 13:54:51	Repeat every day	<input type="button" value="go"/>	
<input type="button" value="go"/>	friday	1/13/2010 13:55:54	Repeat every week on Friday	<input type="button" value="go"/>	
<input type="button" value="go"/>	GROUP1	1/13/2010 13:56:18	Repeat every	<input type="button" value="go"/>	<input type="button" value="go"/>
<input type="button" value="go"/>	GROUP2	1/13/2010 13:56:37	Repeat every	<input type="button" value="go"/>	<input type="button" value="go"/>
<input type="button" value="go"/>	quarterly	1/13/2010 13:57:08	Repeat every 90 days	<input type="button" value="go"/>	
<input type="button" value="go"/>	YEARLY	1/13/2010 13:57:25	Repeat every 365 days	<input type="button" value="go"/>	

## Considerations

- A Recurring Recipe is the schedule which contains the instructions as to when a recurring transaction is billed. The Recurring Repetitions/Remaining Repetitions is the number of times that a transaction follows the recipe. Once a transaction is set as a Recurring Transaction, it will continue to follow the recipe until the number of repetitions cycles down to or is manually set to zero.
- Separate recipes do not need to be built for transactions following the same schedule, even if the

transactions are initiated at different times, have different amounts, and different necessary repetitions. There is no limit to the number of transactions that can use the same recipe.

- There is no limit to the number of recipes that a merchant can build.
- The recurring cycle begins each night at 12 Midnight, Mountain time. Any necessary modifications to a recurring transaction, or recurring recipe must be completed prior to 11:59 PM for it to be reflected as a part of the next day's recurring cycle. For instance, if it is January 31st and a recurring transaction is scheduled to process on February 1st, that transaction can be modified up to 11:59 PM on January 31st. In further explanation, if a merchant needed to set the remaining repetitions to zero to prevent future transactions, but missed the 11:59 PM deadline, the merchant would have access to change the number of remaining repetitions to zero. However, since the cycle had already begun, the transaction would still be billed. Future transactions would be prevented, but a refund or void would now be necessary because the transaction which the merchant had intended to stop was billed. The remaining repetitions in such a case would display as "-1".
- When a transaction is initially submitted for processing, recurring details may be passed as part of the form that will automatically create future recurring charges, based on the details that you provide. In addition, you may also modify previously submitted transactions and mark them as recurring. This is done via the Transaction Listing.
- If you do not need to initiate a charge at the time of entry, use a zero amount AVSOnly transaction type.
- The calculations used to determine when a transaction will recur are based on the initial transaction date.
- The largest allowed value for the Recurring Repetitions is "99999".

## The Recurring Recipe Builder

To access the Recurring Recipe Builder, please click the "Add Recipe" link in the Recurring Transaction window. Complete this interface and click the "Create Recipe" button and the recipe will be added to the list of recipes.

### Figure 3.2. Recurring Recipe Builder Example

---

## RECURRING TRANSACTION RECIPE BUILDER

---

**Merchant Name:** ACC Live Test Co.

---

**Recipe Name:**   
One word only. Must be lower-case. A through Z and 0 through 9 allowed.

---

**Scheduled** Delay Period  days.

---

**Day** Repeat every  days.

---

**Week** Repeat every  weeks on  
 Mon  Tue  Wed  Thu  Fri  Sat  Sun

---

**Month** Repeat every  months on the following day(s) of the month:

<input type="checkbox"/> 1	<input type="checkbox"/> 2	<input type="checkbox"/> 3	<input type="checkbox"/> 4	<input type="checkbox"/> 5	<input type="checkbox"/> 6	<input type="checkbox"/> 7	<input type="checkbox"/> 8	<input type="checkbox"/> 9	<input type="checkbox"/> 10
<input type="checkbox"/> 11	<input type="checkbox"/> 12	<input type="checkbox"/> 13	<input type="checkbox"/> 14	<input type="checkbox"/> 15	<input type="checkbox"/> 16	<input type="checkbox"/> 17	<input type="checkbox"/> 18	<input type="checkbox"/> 19	<input type="checkbox"/> 20
<input type="checkbox"/> 21	<input type="checkbox"/> 22	<input type="checkbox"/> 23	<input type="checkbox"/> 24	<input type="checkbox"/> 25	<input type="checkbox"/> 26	<input type="checkbox"/> 27	<input type="checkbox"/> 28	<input type="checkbox"/> Last Day	

---

**Split Amounts**  **Yes**  
Allows two separate transaction totals for recurring billing: one for the initial transaction and one for future recurring transactions. (This is required if you plan to change transaction totals for future recurring instances.)

---

**Email Text** *Text entered here will appear in customer confirmation email.*

---

**CREATE RECIPE** NOTE! Changes will affect all transactions currently using this recipe!

Calculations used to determine when transactions are processed are always based on the initial (parent) transaction date.

[List Recipes](#)

## Recipe Name

When selecting a recipe name, please remember that it will be case-sensitive (must be lowercase) and can be only one word. Any alpha-numeric characters can be used. You should make it easy to remember. For instance, you may want to name a recipe "1stofmonth" if it's designed to bill on the first day of the month.

## Recipe Types

The next section of the Recipe Builder is the Recipe Types. Only one Recipe Type may be used per recipe.

- **Scheduled Recipes** - Using a Scheduled Recipe allows you to run ALL transactions linked to a recipe at a date that can be controlled and scheduled manually using the scheduling tool. The scheduling tool may be accessed from your Recipe List after a Scheduled Recipe is built. The scheduling tool is only available for scheduled recipes. The "Delay Period" can be used to prevent a transaction from recurring too soon after the initial transaction is processed. The "Delay Period" is the number of days after the original transaction before it is eligible for a scheduled recurrence. To build a Scheduled Recipe, choose a recipe name, click the radio button to the left of "Scheduled", enter a numeric value for the number of days in the "Delay Period", add any additional features, and click the "Create Recipe" button.
- **Day Recipes** - This type of recipe allows a merchant to bill transactions applied to a recipe to bill every X number of days after the initial billing (and from billing to billing). To build a Day Recipe, choose a recipe name, click the radio button to the left of "Day", enter the value for the number of days between recurrings, add any additional features, and click the "Create Recipe" button.
- **Week Recipes** - Building a Week Recipe allows a merchant to bill a transaction on specific days of the week - even multiple days during the same week. A merchant can select 1, 2, or 3 weeks between billings and can check any day or (days) for the billings to take place. To build a Week Recipe, choose a recipe name, click the radio button to the left of "Week", select the value for the number of weeks between recurrings, select the day (or days) of the week on which the billings will take place, add any additional features, and click the "Create Recipe" button.
- **Month Recipes** - The type of recipe allows a merchant to bill transactions every X number of months on the Nth day (or days) of that month. Since some months have only 28, 30, or 31 days in the month, days 29-31 are covered under the "Last Day" selection. This type of recipe assumes that the recurring will begin in the calendar month after the initial transaction is processed. This means if, for instance, a transaction is billed on January 5th, and the recipe instructions are built to bill every 1 month on the 15th day of the month, the transaction would experience its first recurring billing on February 15th (not on January 15th). To build a Month Recipe, choose a recipe name, click the radio button to the left of "Month", select the value for the number of months between recurrings, select the day (or days) of the month on which the billings will take place, add any additional features, and click the "Create Recipe" button.

## The Split Amounts Function

Don't be confused by the name of this feature. This feature can be used with any of the Recipe Types. This feature allows the recurring transactions to be billed a different amount than the initial transaction. The amount can be changed automatically is setting up a form based recurring transaction using a recipe built with the Split Amounts function, or the amount can be changed manually in the "Edit Recurring Items" interface. If there is ever the potential that the amount of a billing may increase, it is wise to set all recipes to allow Split Amounts.

## The Hold On Failure Function

This feature is available for credit card transactions only. If this is enabled on a recipe and a recurring attempt fails, that transaction will be put "on hold" automatically to prevent future billing attempts so that account information can be updated (either by the merchant or the customer). If this is enabled, the recurring postback will include the "on hold" parameter to notify if this has been triggered due to a failure.

### **The Allow Customer Update Function**

This feature is available for credit card transactions only. If enabled for a recipe, confirmation emails for successful and failed recurring transactions will include a link to a secure billing update page. A cardholder will be able to follow the simple instructions to change their credit card billing information through a secure billing interface. If a recurring transaction failure triggered the necessary update, a new transaction will be attempted at the time of the update to bring the account payments up to date for the failed billing. If enabled, the recurring postback will include the "billing\_update\_token".

### **The Email Text Entry**

This allows a merchant to pass a generic message in the text of each of the confirmation emails sent out when a transaction using the recipe recurs.

### **The Recurring Help Window**

This window offers a quick reference guide when creating new recipes or setting transactions as recurring.

---

# Chapter 4. Setting Transactions To Recur

## Setting Transactions To Recur

A transaction can be set to recur automatically at the time of the transaction or manually anytime there after.

### Automatic Recurring Activation Using HTML

Recurring transactions may be initiated at the time the original transaction is processed. To initially set a transaction as recurring, simply add the following input fields to your HTML order form. In this example we'll use the "monthly13" recipe and have the transaction recur six times, with a recurring total of \$100.00 and a recurring description of "test2":

**Table 4.1. HTML Recurring Example**

```
<input type="hidden" name="recur_recipe" value="monthly13">
<input type="hidden" name="recur_reps" value="6">
<!-- Optional (For Split Recurring) -->
<input type="hidden" name="recur_total" value="100.00">
<input type="hidden" name="recur_desc" value="test2">
```

### Automatic Recurring Activation Using XML

Recurring transactions may be initiated at the time the original transaction is processed. To initially set a transaction as recurring, simply include the following fields to your AuthTransaction XML query. In this example we'll use the "monthly13" recipe and have the transaction recur six times, with a recurring total of \$100.00 and a recurring description of "test2":

**Table 4.2. XML Recurring Example**

```
<RecurringData>
<RecurRecipe>monthly13</RecurRecipe>
<RecurReps>6</RecurReps>
<!-- Optional (For Split Recurring) -->
<RecurTotal>100.00</RecurTotal>
<!-- Optional (For Split Recurring) -->
<RecurDesc>test2</RecurDesc>
<!-- Optional (For Split Recurring) -->
</RecurringData>
```

### Manual Recurring Activation

This manual activation method can be used for transactions that were submitted via HTML or XML. Once a sale transaction has been processed successfully, it can be set as a recurring transaction by following these simple steps:

## Setting Transactions To Recur

1. Log into the Control Panel and open the Transaction Listing for the day when the original transaction was processed. Locate the transaction that needs to be set to recur and click on the XID number to open the Transaction Detail request screen.

**Figure 4.1. Transaction Listing Example**

(There may be a 10-15 minute delay before new transactions appear.)

**ACC Live Test Co. (66646) Transaction Report**

[Print Listing](#)   [Explanation of Codes](#)   [Close Window](#)

Page 1 of 1   First   Prev   Next   Last

Download data set in [CSV](#) format.   [Click the XID for transaction details.](#)   [Estimated Open Batch Total](#)

DATE & TIME	XID	PXID	CXID	ACTION	STATUS	AVS	CVV	TYPE	LAST FOUR	FIRST	LAST	AUTH #	BATCH	AMOUNT	RECUR	OPTIONS
4/17/2007 13:34:47	20840479			Credit	Ok				5454	test	Smith		21	\$5.00		<a href="#">GO</a>
4/18/2007 11:14:37	20850033			Order	Ok	NM			5454	Joel	Dare	TAS566	22	\$10.00		<a href="#">GO</a>
5/7/2007 13:19:16	21058688		23742992 23755897	Order	Ok	NM			5454	Demo	Buyer	TAS486	25	\$4.00		<a href="#">GO</a>
5/16/2007 10:38:16	21158932			Order	Fail				5454					\$1.00		<a href="#">GO</a>
5/16/2007 10:39:38	21158964			Order	Fail				5454					\$1.00		<a href="#">GO</a>
5/16/2007 11:08:32	21159428		21206180	Order	Fail				5454	John	Smith			\$10.00		<a href="#">GO</a>
5/16/2007 12:27:26	21160635		21160648 21322946	Order	Fail					John	Smith			\$10.00		<a href="#">GO</a>
5/16/2007 12:28:01	21160648	21160635		Order	Fail					John	Smith			\$10.00		<a href="#">GO</a>
5/21/2007 12:17:36	21206009		22257882 23802232 23802235	Order	Ok	NM			5454	Jack	Tester	TAS197	28	\$4.75		<a href="#">GO</a>
5/21/2007 12:18:25	21206018		21636413 21826587 21605106 21616373 21647665	Order	Ok	NM			1111	Rick	Tester	TAS218	28	\$10.75		<a href="#">GO</a>
5/21/2007 12:19:00	21206024		21949437	Order	Ok	U			0005	John	Demo	AXS565	28	\$10.75		<a href="#">GO</a>
5/21/2007 12:23:05	21206070			Order	Fail				8431	Bill	Test-Demo			\$0.20		<a href="#">GO</a>
5/21/2007 12:23:55	21206076			Order	Ok	NM			4444	Bill	Test-Demo	TAS259	28	\$5.20		<a href="#">GO</a>
5/21/2007 12:24:46	21206088			Order	Ok	NM			1117	Jim	Test	TAS274	28	\$5.20		<a href="#">GO</a>
5/21/2007 12:26:02	21206099		21206105	Order	Ok	NM			1117	Jim	Test	TAS291	28	\$5.20		<a href="#">GO</a>
5/21/2007 12:26:25	21206105	21206099		Void	Ok				1117	Jim	Test		28	\$5.20		<a href="#">GO</a>
5/21/2007 12:28:12	21206133		21206168	Order	Ok	NM			1111	Dave	Tester	TAS996	28	\$10.00		<a href="#">GO</a>
5/21/2007 12:29:16	21206151			Order	Ok	NM			5454	Steve	Demotester	TAS318	28	\$10.00		<a href="#">GO</a>
5/21/2007 12:29:51	21206157			Order	Ok	NM			5454	Steve	Demotester	TAS562	28	\$6.00		<a href="#">GO</a>
5/21/2007 12:30:40	21206168	21206133		Credit	Ok				1111	Dave	Tester		28	\$7.00		<a href="#">GO</a>
5/21/2007 12:31:28	21206180	21159428		Order	Ok	NM			5454	John	Smith	TAS590	28	\$7.85		<a href="#">GO</a>

Page 1 of 1   First   Prev   Next   Last

2. From the Transaction Detail request screen, click the "Get Detail" button.

**Figure 4.2. Transaction Detail Access Window Example**

**Transaction Detail**


Enter an XID:

[Get Detail](#)

[Go Back](#)   [Close Window](#)

3. In the Transaction Detail Screen, click on the "GO" button in the "RECUR" column.

**Figure 4.3. Transaction Detail Example**

Transaction Detail for XID 32428094												
DATE & TIME	XID	P-XID	C-XID	ACTION	STATUS	INSTR	FIRST NAME	LAST NAME	AUTH #	AMOUNT	OPTIONS	RECUR
1/13/2010 10:12:15	32428094			Order	Ok		Customer	Name	TAS995	\$5.50	<input type="button" value="go"/>	<input type="button" value="go"/>
<b>CUSTOMER INFORMATION:</b>						<b>SHIPPING ADDRESS:</b>						
Name:		Customer Name				Person Name						
Address:		123 Main St				321 Center St						
		BH, CA 90210				BH ,CA 90210						
		USA				USA						
Telephone:		8885551234										
Email:		email@domain.com										
Cust ID:		ABC123										
<b>TRANSACTION DATA:</b>												
Last Four:		5454				AVS Response:		N				
Batch:		458				CVV Response:						
Order Total:		\$5.50				IP Address:		65.103.239.179				
Transaction Source:		session				Error Message:						
<b>FORM ITEMS:</b>												
<b>Item 1</b>												
cost		5.50										
desc		Monthly Payment										
qty		1										

- In the Recurring Detail window, enter the number of repetitions (number of times a transaction needs to rebill) and choose the Recurring Recipe Name from the drop down menu. If you need to change the amount to be billed or the items, uncheck the "Use Original Order Items" box and editing and additional line tools will display. Once the necessary fields are completed, click the "Setup Recurring" button.

**Figure 4.4. Recurring Detail Example: Non-Recurring Transaction**



## Create New Recurring Transaction

XID 103019 is not currently setup for recurring. Use this form to create a new recurring transaction using the billing information from the parent transaction.

### Parent Info

**Type:** Card  
**Card Type:** MasterCard  
**Last Four:** 5454  
**Expiration:** 6/2010

### Contact Info

**Name:** Customer Name  
**Phone:** 8885551234  
**Email:** email@domain.com  
**User ID:** Test123ABC

### Select Recipe

Recipe  
testrec - day

### Recurring Repetitions

Repetitions  
0

### Recurring Items

Use Original Order Items

Description	Quantity	Price	Total
Item 1	1	10.0	10.0

Recurring Amount  
\$10.00

Setup Recurring

Go Back

Back To Detail

5. The Recurring Setup information page will display if all of the fields have been filled out correctly and the detail has been updated.

**Figure 4.5. Recurring Setup Information Page Example**

## Recurring Setup

### Parent Transaction

**XID:** 103019  
**Date:** 2010-03-24 11:27:04

### Schedule

**Recipe:** testrec  
**Interval:** day  
**Next Run:** 2010-03-24  
**Remaining Reps:** 1

## Billing Items

Description	Quantity	Price	Total
Item 1edited	1	\$5.00	\$5.00

[Edit Setup](#)

## Billing Information

### Contact Info

**Name:** Customer Name  
**Phone:** 8885551234  
**Email:** email@domain.com  
**User ID:** Test123ABC

### Payment Info

**Type:** Card  
**Card Type:** MasterCard  
**Last Four:** 5454  
**Expiration:** 6/2010

### Billing Address

123 Main St  
BH Ca, 90210  
USA

### Shipping Address

[Edit Billing Info](#)

[Go Back](#)

[Back To Detail](#)

## Recurring Activation In The Virtual Terminal

The Recurring Transaction Virtual Terminal Interface can be accessed by opening the Standard Virtual Terminal interface. Choosing this interface allows for the entry of multiple Order Items, as well as separate shipping and tax charges. It also allows for the entry of recurring billing information by clicking the Recurring toggle. Using this interface will charge a transaction in real-time, but will also schedule future transactions on that payment account. If you do not need to initiate a credit card charge at the time of entry, use a zero amount AVSOnly transaction type. This interface will not charge a card without valid recurring billing information.

### Figure 4.6. Recurring Virtual Terminal Welcome Section Example

[Classic Virtual Terminal](#) [Swipe Card](#) [Swipe Card Express](#)

#### Transaction Information

**Pay To:** Carl's Computers **ID:** 27

This Initial Transaction Information section is where a merchant enters the general customer information when generating the initial charge. This same general information will be default information for future recurring transactions (unless the user information is modified in the Recurring Transaction Detail interface).

**Figure 4.7. Recurring Virtual Terminal Transaction Information Section Example**

Item	Description	Qty	Price	Total
1	Order Item 1	1	5.55	5.55
2	Order Item 2	1	45.20	45.20
3		0	0.00	0.00
4		0	0.00	0.00
5		0	0.00	0.00
<b>Total</b>				<b>50.75</b>

Check the boxes below to add optional amounts for shipping and tax. Then enter separate amounts for each. These amounts will be added to the order total.

<input checked="" type="checkbox"/> Include Shipping   Amount:	7.99
<input type="checkbox"/> Include Tax   Amount:	0.00
<b>Order Total:</b>	<b>58.74</b>

**Email Text:**

Some of the entry fields in this area are required and others are optional (See Standard Virtual Terminal Order Section Example). A merchant can choose to enter up to ten separate items plus shipping and tax amounts, or can submit a single item which is a total of the amount to be billed to the customer. To access items 6-10, please use the scroll bar on the left side of the "Total" column.

- **Item Description** - A merchant should enter the name of the product that a customer is purchasing in this field. This information will be recorded in the merchant's Transaction Listing in the Control Panel and in the Merchant/Customer confirmation emails. Some merchants choose to enter all of the items in a single line item - either with each item detailed, or with a generic description like "Purchased Items". This can be done as long as the value for the Item Qty is "1" and the total price of the purchase is entered into the Item Price field.
- **Item Qty** - This value will be multiplied by the amount listed in the Item Price field to provide the value for the Item Total. This value can be "1", even if you are selling multiple quantities - as long as the Item Price amount is the cost of all of the products combined.
- **Item Price** - The amount listed here will be multiplied by the value listed in the Item Qty to provide the value for the Item Total.

- **Item Total** - This value is arrived at when the Virtual Terminal automatically multiplies the value of the Item Qty and the Item Price for a single item.
- **Total** - This amount is the sum of the Item Totals for all items purchased.
- **Include Shipping Checkbox** - This should be selected if a merchant would like shipping to be a separate line item. This must be used in conjunction with an entry in the Shipping Amount field.
- **Shipping Amount** - This value should be the amount of shipping for the entire purchase. The Virtual Terminal does not calculate shipping. A merchant will need to calculate that prior to entering the transaction in this interface. If the Include Shipping checkbox is selected, there must be a value in this field.
- **Include Tax Checkbox** - This should be selected if a merchant would like tax to be a separate line item. This must be used in conjunction with an entry in the Tax Amount field.
- **Tax Amount** - This value should be the amount of tax for the entire purchase. The Virtual Terminal does not calculate tax rates. A merchant will need to calculate that prior to entering the transaction in this interface. If the Include Tax checkbox is selected, there must be a value in this field.
- **Order Total** - This value is the sum of the Total, the Shipping amount, and the Tax amount. This is the amount that will be charged to the customer's card. If this value is zero for a credit card, the transaction will run as an AVSOnly transaction.
- **Email Text** - This field allows a merchant to enter a message up to 255 characters which will display on both the merchant confirmation email and on the customer confirmation email.
  
- **Optional Fields** - A merchant may place an entry into any of these non-required fields:
  - **Include Shipping Checkbox** - This should be selected if a merchant would like shipping to be a separate line item. This must be used in conjunction with an entry in the Shipping Amount field.
  - **Shipping Amount** - This value should be the amount of shipping for the entire purchase. The Virtual Terminal does not calculate shipping. A merchant will need to calculate that prior to entering the transaction in this interface. If the Include Shipping checkbox is selected, there must be a value in this field.
  - **Include Tax Checkbox** - This should be selected if a merchant would like tax to be a separate line item. This must be used in conjunction with an entry in the Tax Amount field.
  - **Tax Amount** - This value should be the amount of tax for the entire purchase. The Virtual Terminal does not calculate tax rates. A merchant will need to calculate that prior to entering the transaction in this interface. If the Include Tax checkbox is selected, there must be a value in this field.
  - **Email Text** - This field allows a merchant to enter a message up to 255 characters which will display on both the merchant confirmation email and on the customer confirmation email.

The payment area of the interface will look different for each merchant depending on what payment types they are authorized to accept.

### **Figure 4.8. Standard Virtual Terminal Payment Section Example**

**Payment Method:**  Credit Card  Check

**Card Information**

**Card Number:**  **Exp. Date:**

**CVV Number:**  (CVV information is not required.)

**Approval Code:**  (Enter ONLY for force transaction.)

**Auth ONLY:**  (Only use for pre-auth/auth-only.)

**Checking Account Information**

**ABA Number:** "  " **Account Number:** "  "

**Account Source**

To begin to enter payment information, a merchant must select the radio button for the customer's payment method (either Check or Credit Card). This radio button will enable the appropriate/required fields for the payment type and disable the others.

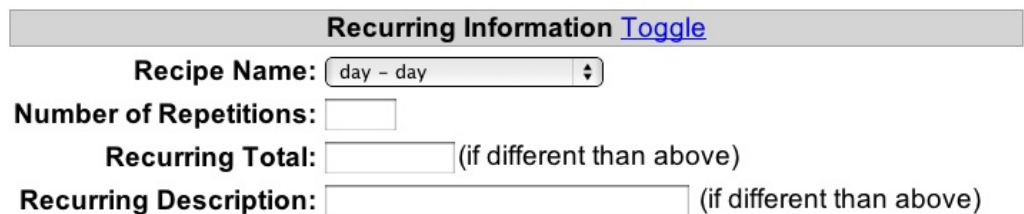
- **Credit Card Information** - These fields will be enabled if a merchant selects the Credit Card Payment Method radio button.
  - **Card Number** - The customer's credit card number should be entered into this field without any dashes or spaces.
  - **Exp. Date** - The expiration month and year should be selected in this area.
  - **Approval Code** - The value for this field can only be obtained directly from the Credit Card Merchant Account Processor's Voice Approval phone service. This feature should only be used if a "call authorization center" error response was received during a previous authorization attempt. The approval code will be a numeric or alpha-numeric code provided by the Voice Approval service. The gateway does not provide voice approval codes. Those codes must be obtained directly from the Merchant Account Processor.
  - **CVV Number** - The value for this field is the CVV or CVV2/CID code listed on the credit card. This three or four digit numeric code is used as a fraud deterrent.
  - **Auth Only Checkbox** - A merchant should never check this box, unless they do not desire to actually charge a customer's card. When this is selected the transaction will only run a "pre-authorization" which verifies the card account and a set amount in the account, but it does not actually charge the card. The pre-authorized amount is "frozen" on the account. A pre-authorized transaction can be converted to a full transaction by running a post-authorization from the Transaction Listing. If no post-authorization is run, the money is never paid to the merchant and the "frozen" funds will be released back to the customer's available credit limit after 10 business days.
- **Checking Account Information** - These fields will be enabled if a merchant selects the Check Payment Method radio button.
  - **ABA Number** - This is the nine digit ABA Routing number for a customer's bank. These

are generally the first nine numbers listed in the line of numbers across the bottom of a check.

- **Account Number** - This is the customer's checking account number as it appears on a check.
- **Account Type** - (For EFT Transactions Only) A merchant needs to use the selection tool to indicate whether the customer's checking account is a Personal or a Business checking account.
- **Account Source** - A NACHA authorized merchant needs to use the selection tool to indicate whether the customer's checking account is a checking or a savings account.
- **SEC Code** - (For EFT Transactions Only) Depending upon the nature of the EFT processing account, a merchant may be required to designate the three letter Standard Entry Category for the transaction. Potential values are:
  - PPD - Prearranged payment and deposit
  - CCD - Corporate credit or debit
  - ARC - Accounts receivable entry
  - BOC - Back office conversion
  - POP - Point of purchase
  - RCK - Returned check entry
  - WEB - Internet initiated entry
  - TEL - Telephone initiated entry

To make your transaction a recurring transaction, click the "Toggle" button to display the appropriate entry fields.

**Figure 4.9. Standard Virtual Terminal Recurring Section Example**



**Recurring Information [Toggle](#)**

**Recipe Name:**

**Number of Repetitions:**

**Recurring Total:**  (if different than above)

**Recurring Description:**  (if different than above)

Recurring Fields

- **Recipe Name** - This drop down menu displays all of a merchant's pre-built recipes, which will provide the rules and schedule by which a transaction will re-bill when set with at least one remaining repetition.
- **Number of Repetitions** - This is the numeric value for the amount of times the Recurring Recipe needs to cycle. Each successful repetition will cycle down the number of remaining

repetitions by one until it reaches zero.

- **Recurring Total** - If the amount that is to recur is the same as the total amount listed in the Initial Transaction Information, please leave this blank. This feature can be used in conjunction with Recurring Recipes designated by the merchant as a "Split Amount" recipe. When an amount is entered into this field, that Recurring Total will be the amount billed when the transaction recurs. For example, merchants who bill a one time setup fee and then a different amount for monthly service fees, would put the amount of the monthly service fee in the Recurring Total field.
- **Recurring Description** - If the Item Description used in the Initial Transaction Information is a sufficient explanation for both the initial payment and any subsequent recurring transactions, please leave this blank. If, however, the merchant would like this to display differently on subsequent recurring billings, please enter an adequate description in this field.

This area of the interface allows the merchant do enter customer data that will be saved to the gateway.

**Figure 4.10. Standard Virtual Terminal Information Section Example**

Additional Information	
Billing	Optional Shipping
<b>First Name:</b> <input type="text"/> <b>Last Name:</b> <input type="text"/> <b>Address:</b> <input type="text"/> <b>City:</b> <input type="text"/> <b>State:</b> <input type="text"/> <b>Zip:</b> <input type="text"/> <b>Country:</b> <input type="text" value="USA"/> <b>Phone:</b> <input type="text"/> <b>Cust ID (optional):</b> <input type="text"/> <b>E-Mail:</b> <input type="text"/> (You MUST enter a VALID email address. Use your own if your customer does not have one.)	<input type="button" value="Same As Billing"/> <b>First Name:</b> <input type="text"/> <b>Last Name:</b> <input type="text"/> <b>Address:</b> <input type="text"/> <b>City:</b> <input type="text"/> <b>State:</b> <input type="text"/> <b>Zip:</b> <input type="text"/> <b>Country:</b> <input type="text"/>

*Please Do Not Press This Button More Than Once*

- **Billing Information** - All fields here are required unless otherwise indicated.
  - **First Name** - This should be the customer's first name.
  - **Last Name** - This should be the customer's last name.

- **Address** - This should be the cardholder's street address as listed with the account issuer.
- **City** - This should be the cardholder's city as listed with the account issuer.
- **State** - This should be the state abbreviation of the cardholder as listed with the account issuer.
- **ZIP** - This should be the cardholder's postal code as listed with the account issuer.
- **Country** - This should be the cardholder's country as listed with the account issuer.
- **Phone** - This should be a contact phone number for the customer.
- **Cust ID** - This is an optional field that allows a merchant to enter a tracking number for their customers.
- **Email** - This should be the customer's email address. The transaction confirmation email will be sent to this address.
- **Optional Shipping Information** - Each of these fields are optional and can be populated with alternative shipping address data. The processing banks are unable to verify this information.

To submit a transaction through this interface, enter the correct data into the required fields and any of the desired optional fields. Be sure to double check the credit card number, the amount of the charge, and the recurring billing information. Click the "Process Payment" button. The transaction will be attempted in real-time. The gateway will either display an approval screen or a failure screen. The failure screen will list the reason for the failure. An approval page will display if the transaction is successful.

**Figure 4.11. Approval Page Example**

---

	<b>Merchant Information</b>
Merchant Name: ACC Live Test Co.	

---

	<b>Customer Information</b>
Customer Name: Test Customer	
Address: 123 Main St	
BHS, Ca 90210	
USA	
Phone: 8885551234	
Cust ID: 123ABC	
Email: email@domain.com <i>A confirmation email has been sent.</i>	

---

	<b>Order Information</b>
Transaction ID (XID): 9999999999	
Authorization Code: 000000	
Date: 1/11/2010 14:27:13 MT	
AVS Response: X	
Order Total: 58.74	

---

<a href="#">Enter Another Transaction</a>	<a href="#">Print This Page</a>	<a href="#">Close This Window</a>
---	---------------------------------	-----------------------------------

---



## Modifying Recurring Transaction Information Through The Transaction Listing

Recurring information can be modified so that future recurring attempts will be made using updated information. Updating the data for a recurring transaction does not change information applied to the originating transaction itself, only future transactions. Resubmits using the originating transaction will use the original transaction information.

1. Log into the Control Panel and open the Transaction Listing for the day when the original transaction was processed. Locate the transaction that needs to be modified and click on the XID number to open the Transaction Detail request screen.

**Figure 4.12. Transaction Listing Example**

(There may be a 10-15 minute delay before new transactions appear.)

**ACC Live Test Co. (66646) Transaction Report**

Page 1 of 1 First Prev Next Last

Download data set in   [Click the XID for transaction details.](#) Estimated Open Batch Total

DATE & TIME	XID	PXID	CXID	ACTION	STATUS	AVS	CVV	TYPE	LAST FOUR	FIRST	LAST	AUTH #	BATCH	AMOUNT	RECUR	OPTIONS
4/17/2007 13:34:47	20840478			Credit	Ok				5454	test	Smith		21	\$5.00		<input type="button" value="GO"/>
4/18/2007 11:14:37	20850033			Order	Ok	NM			5454	Joel	Dare	TAS566	22	\$10.00		<input type="button" value="GO"/>
5/7/2007 13:19:16	21058688		23742992 23755897	Order	Ok	NM			5454	Demo	Buyer	TAS486	25	\$4.00		<input type="button" value="GO"/>
5/16/2007 10:38:16	21158932			Order	Fail				5454	1				\$1.00		<input type="button" value="GO"/>
5/16/2007 10:39:38	21158964			Order	Fail				5454					\$1.00		<input type="button" value="GO"/>
5/16/2007 11:08:32	21159428		21206180	Order	Fail				5454	John	Smith			\$10.00		<input type="button" value="GO"/>
5/16/2007 12:27:26	21160635		21160648 21322946	Order						John	Smith			\$10.00	<input type="button" value="GO"/>	<input type="button" value="GO"/>
5/16/2007 12:28:01	21160648	21160635		Order	Fail				-	John	Smith			\$10.00		<input type="button" value="GO"/>
5/21/2007 12:17:36	21206009		22257882 23802232 23802235	Order	Ok	NM			5454	Jack	Tester	TAS197	28	\$4.75		<input type="button" value="GO"/>
5/21/2007 12:18:25	21206018		21636413 21626587 21605106 21616373 21647665	Order	Ok	NM			1111	Rick	Tester	TAS218	28	\$10.75		<input type="button" value="GO"/>
5/21/2007 12:19:00	21206024		21949437	Order	Ok	U			0005	John	Demo	AXS565	28	\$10.75		<input type="button" value="GO"/>
5/21/2007 12:23:05	21206070			Order	Fail				8431	Bill	Test-Demo			\$9.20		<input type="button" value="GO"/>
5/21/2007 12:23:55	21206076			Order	Ok	NM			4444	Bill	Test-Demo	TAS259	28	\$5.20		<input type="button" value="GO"/>
5/21/2007 12:24:46	21206088			Order	Ok	NM			1117	Jim	Test	TAS274	28	\$5.20		<input type="button" value="GO"/>
5/21/2007 12:26:02	21206099		21206105	Order	Ok	NM			1117	Jim	Test	TAS291	28	\$5.20		<input type="button" value="GO"/>
5/21/2007 12:26:25	21206105	21206099		Void	Ok				1117	Jim	Test		28	\$5.20		<input type="button" value="GO"/>
5/21/2007 12:28:12	21206133		21206168	Order	Ok	NM			1111	Dave	Tester	TAS996	28	\$10.00		<input type="button" value="GO"/>
5/21/2007 12:29:16	21206151			Order	Ok	NM			5454	Steve	Demotester	TAS318	28	\$10.00		<input type="button" value="GO"/>
5/21/2007 12:29:51	21206157			Order	Ok	NM			5454	Steve	Demotester	TAS562	28	\$6.00		<input type="button" value="GO"/>
5/21/2007 12:30:40	21206168	21206133		Credit	Ok				1111	Dave	Tester		28	\$7.00		<input type="button" value="GO"/>
5/21/2007 12:31:28	21206180	21159428		Order	Ok	NM			5454	John	Smith	TAS590	28	\$7.85		<input type="button" value="GO"/>

Page 1 of 1 First Prev Next Last

2. From the Transaction Detail request screen, click the "Get Detail" button (*See Figure 4.13*).

**Figure 4.13. Transaction Detail Access Window Example**



3. In the Transaction Detail screen, click on the "GO" button in the "RECUR" column (to change the recipe being used and/or the remaining repetitions) or click on the "GO" button in either the "Edit Recurring User Info" section (to update the customer's billing or address information) or the "Edit Recurring Item Info" section (to update the amount or description of a recurring transaction) to open the editing interface (See *Figure 4.14*).

**Figure 4.14. Transaction Detail Example**

Transaction Detail for XID 32428094

DATE & TIME	XID	P-XID	C-XID	ACTION	STATUS	INSTR	FIRST NAME	LAST NAME	AUTH #	AMOUNT	OPTIONS	RECUR
1/13/2010 10:12:15	32428094			Order	Ok		Customer	Name	TAS995	\$5.50	<input type="button" value="go"/>	<input type="button" value="go"/>

CUSTOMER INFORMATION:		SHIPPING ADDRESS:	
Name:	Customer Name	Person Name	
Address:	123 Main St	321 Center St	
	BH, CA 90210	BH ,CA 90210	
	USA	USA	
Telephone:	8885551234		
Email:	email@domain.com		
Cust ID:	ABC123		
TRANSACTION DATA:			
Last Four:	5454	AVS Response:	N
Batch:	458	CVV Response:	
Order Total:	\$5.50	IP Address:	65.103.239.179
Transaction Source:	session	Error Message:	
FORM ITEMS:			
<b>Item 1</b>			
cost	5.50		
desc	Monthly Payment		
qty	1		
RECURRING INFORMATION:			
Start Date:	1/13/2010		
Recipe Name:	1st		
Remaining Reps:	924		
Recurring Total:	14.00		
On Hold:	Yes		
Recurring History:	<input type="button" value="go"/>		

**The Recurring Information Area**

When a transaction is set to be a recurring transaction this area will display.

- **Start Date** - The day of when a transaction was set as a recurring transaction. This date may or may not coincide with the date of the original transaction.
- **Recipe Name** - This value is the name of the recurring recipe that is set to the transaction at that time.
- **Remaining Reps** - The number of repetitions left that the transaction will continue to rebill until it cycles to zero or is set to zero.
- **Recurring Total** - The amount that will be charged for future transactions.
- **On Hold** - This will display a red YES or a green NO indicating whether the transaction is on hold to prevent future billings or not. By clicking on the value, you can toggle between on hold (YES) or no on hold (NO).
- **Recurring History** - Clicking this "GO" button will pull open the history of the recurring at-

tempts on the specific transaction.

- **Edit Recurring User Info** - This "GO" button opens the interface where the cardholder's information can be modified for future recurring transactions.
  - **Edit Recurring Item Info** - This "GO" button opens the window that allows a merchant to modify the amount and description of recurring transactions.
- 4.
- When changing the recipe or number of remaining repetitions, a merchant is taken to the Recurring Detail page (See Figure 4.15).

**Figure 4.15. Recurring Detail Example: Recurring Transaction**

### Recurring Setup

---

<p><b>Parent Transaction</b>  <b>XID:</b> 103018  <b>Date:</b> 2010-03-24 11:03:52</p>	<p><b>Schedule</b>  <b>Recipe:</b> Monthly1  <b>Interval:</b> month  <b>Next Run:</b> 2010-04-01  <b>Remaining Reps:</b> 10</p>
--	---

### Billing Items

---

Description	Quantity	Price	Total
Monthly Charge	1	\$5.55	\$5.55

---

[Edit Setup](#)

### Billing Information

---

<p><b>Contact Info</b>  <b>Name:</b> Customer Name  <b>Phone:</b> 8885551234  <b>Email:</b> email@domain.com  <b>User ID:</b> Test123ABC</p>	<p><b>Payment Info</b>  <b>Type:</b> Card  <b>Card Type:</b> MasterCard  <b>Last Four:</b> 5454  <b>Expiration:</b> 6/2010</p>
<p><b>Billing Address</b>                  123 Main St                  BH Ca, 90210                  USA</p>	<p><b>Shipping Address</b>                  ,</p>

---

[Edit Billing Info](#)

- Click into the "REPETITIONS" field, delete the value, and enter the new number of repetitions (set it to "0" to stop future recurrings). Be sure to click the "GO" button in the "UPDATE" column.

OR

- Click into the "RECIPE NAME" field, delete the value, and enter the new recipe name (a merchant can click the "View/Select Recipe" button to open the list of current recipes including a link to build a new recipe). There must always be a valid recipe name in this field

if a transaction has ever been a recurring transaction - even if the transaction is no longer recurring. Be sure to click the "GO" button in the "UPDATE" column.

OR

- A merchant can modify both of these fields to modify a recurring transaction. Be sure to click the "GO" button in the "UPDATE" column.
- The page also gives a merchant a way to access the "Edit Recurring User Info" and "Edit Recurring Item Info" interfaces explained below.
- When changing the customer's billing information in the "Edit Recurring User Info" interface, the User Info Editing window is opened (*See Figure 4.16*).

**Figure 4.16. Recurring User Info Editing Interface Example**

**EDITING CUSTOMER INFORMATION**

USER INFO	
First Name	Customer
Last Name	Name
Address	123 Main St
City	BH
State	CA
Zip	90210
Country	USA
Ship First Name	Person
Ship Last Name	Name
Ship Address	321 Center St
Ship City	BH
Ship State	CA
Ship Zip	90210
Ship Country	USA
Email	email@domain.com
Cust ID	ABC123
Phone	8885551234

CARD INFO	
Current Card Type	MasterCard
Current Last Four	5454
New Card Number	
New Exp Month	7
New Exp Year	2016

Next -->

- a. Click into the field that needs to be modified, delete the value, and enter the new value. Remember, any changes to any of the fields in the "CARD INFO" section require that the credit card number is entered - even if the card number is the same card number currently on file.
- b. The page will display featuring the changes in red. (See Figure 4.17)

**Figure 4.17. Recurring User Info Changed Example**

**EDITING CUSTOMER INFORMATION**

USER INFO		
	New Settings	Old Settings
First Name	Customer	Customer
Last Name	Name	Name
Address	789 Main St	123 Main St
City	BH	BH
State	CA	CA
Zip	90210	90210
Country	USA	USA
Ship First Name	Person	Person
Ship Last Name	Name	Name
Ship Address	321 Center St	321 Center St
Ship City	BH	BH
Ship State	CA	CA
Ship Zip	90210	90210
Ship Country	USA	USA
Email	email@domain.com	email@domain.com
Cust ID	ABC123	ABC123
Phone	8885551234	8885551234

CARD INFO		
	New Settings	Old Settings
Card Type	MasterCard	MasterCard
Last Four		5454
Card Number	5454545454545454	
Exp Month	10	7
Exp Year	2020	2016

- c. Either click the "Go Back" button to edit any other data or click the "Submit Changes" button to update the information, at which point the success page will display (See Figure 4.18).

**Figure 4.18. Successful Update Page Example**

## Recurring Setup

### Parent Transaction

**XID:** 103019  
**Date:** 2010-03-24 11:27:04

### Schedule

**Recipe:** testrec  
**Interval:** day  
**Next Run:** 2010-03-24  
**Remaining Reps:** 1

## Billing Items

Description	Quantity	Price	Total
Item 1edited	1	\$5.00	\$5.00

[Edit Setup](#)

## Billing Information

### Contact Info

**Name:** Customer Name  
**Phone:** 8885551234  
**Email:** email@domain.com  
**User ID:** Test123ABC

### Payment Info

**Type:** Card  
**Card Type:** MasterCard  
**Last Four:** 5454  
**Expiration:** 6/2010

### Billing Address

123 Main St  
BH Ca, 90210  
USA

### Shipping Address

[Edit Billing Info](#)

[Go Back](#)

[Back To Detail](#)

- When changing the transaction information in the "Edit Recurring Item Info" interface, the Items Editing window is opened (See *Figure 4.19*).

**Figure 4.19. Recurring Items Editing Interface Example**



**EDITING RECURRENT TRANSACTION ITEMS**

RECUR RECIPE ID: 9791

RECIPE TRANSACTION ID: 30404349

ITEMS				
#	Description	Quantity	Price	Delete
1	Monthly Service Fee	1	14.00	<input type="checkbox"/>
New				Clear
New				Clear
New				Clear
New				Clear
New				Clear

Update    Reset  
Go Back

- To delete an item, click the checkbox in the "Delete" column and enter a new item (including Description, Quantity and Price) - OR - Modify the current item by clicking in to the field and changing the value.
- The "Clear" button will empty any items entered into that row. The "Reset" button clears any data from any of the rows listed as "New".
- Once all of the information is edited correctly, click the "Update" button and a success message will display (*See Figure 4.20*).

**Figure 4.20. Recurring Items Changed Example**

**EDITING RECURRENT TRANSACTION ITEMS**

RECUR RECIPE ID: 9791

RECIPE TRANSACTION ID: 30410640

ITEMS				
#	Description	Quantity	Price	Delete
1	Additional Management Fee	1	2.95	<input type="checkbox"/>
2	Monthly Service Fee	1	7.00	<input type="checkbox"/>
New				Clear
New				Clear
New				Clear
New				Clear
New				Clear

Successful update. 2 items modified.

Please click on Back To Admin below to return to the Admin Page.

Update    Reset  
Go Back

## Modifying Recurring Transaction Information Using XML

### The RecurUpdate

This request allows you to modify the number of remaining repetitions for a recurring transaction.

**Table 4.3. XMLTrans2.cgi RecurUpdate Example**

```
<?xml version="1.0"?>
<GatewayInterface>
  <VendorIdentification><!-- Can also be API Credentials -->
    <VendorId>1</VendorId>
    <VendorPassword>test</VendorPassword>
    <HomePage>text</HomePage>
  </VendorIdentification>
  <!-- Other than OperationXID, all of the child elements of RecurUpdate
  are individually optional but you must pass -->
  <!-- one of Recipe, RemReps, CustomerData, OrderItems or Total -->
  <RecurUpdate>
    <OperationXID>12345</OperationXID>
    <!-- Optional.-->
    <RemReps>123</RemReps>
    <!-- Optional.-->
    <Recipe>Recipe Name</Recipe>
    <!-- Optional. Will update customer info tied to recurring transaction if passed-->
    <CustomerData>
      <Email>demo@demo.com</Email>
      <CustId>12345</CustId> <!-- Optional -->
      <BillingAddress>
        <Address1>test</Address1>
        <FirstName>John</FirstName>
        <LastName>Smith</LastName>
        <City>Bountiful</City>
        <State>UT</State>
        <Zip>84032</Zip>
        <Country>USA</Country>
        <Phone>801-555-1212</Phone>
      </BillingAddress>
      <!-- Optional -->
      <ShippingAddress>
        <Address1>test</Address1>
        <FirstName>John</FirstName>
        <LastName>Smith</LastName>
        <City>Bountiful</City>
        <State>UT</State>
        <Zip>84032</Zip>
        <Country>USA</Country>
      </ShippingAddress>
    </CustomerData>
    <!-- Optional. Will update customer info tied to recurring transaction if passed-->
    <AccountInfo>
      <!-- For Credit card transaction. -->
      <CardAccount>
        <AccountNumber>5454545454545454</AccountNumber>
        <ExpirationMonth>01</ExpirationMonth>
        <ExpirationYear>2000</ExpirationYear>
        <CVVNumber>123</CVVNumber><!-- Optional -->
      </CardAccount>
      <!-- For EFT transactions. -->
      <CheckAccount>
        <AccountNumber>123456</AccountNumber>
        <ABA>324377516</ABA>
      </CheckAccount>
    </AccountInfo>
    <!-- Only one of OrderItems or Total elements may be passed in but neither are required -->
    <OrderItems>
      <Item>
        <Description>item1</Description>
        <Cost>5</Cost>
        <Qty>1</Qty>
      </Item>
    </OrderItems>
```

```
<!-- To use the Total element the original transaction
can only have one item associated with it -->
<Total>5.00</Total>
</RecurUpdate>
</GatewayInterface>
```

## The RecurUpdateResponse

This request will return the following response:

**Table 4.4. XMLTrans2.cgi RecurUpdateResponse Example**

```
<?xml version="1.0" standalone="yes"?>
<ItransactInterface>
  <RecurUpdateResponse>
    <Status>ok</Status>
    <ErrorCategory></ErrorCategory>
    <ErrorMessage></ErrorMessage>
    <TimeStamp>20040621154341</TimeStamp>
    <TestMode>0</TestMode>
    <RecurDetails>
      <RemReps>10</RemReps>
      <RecipeName>daily</RecipeName>
      <RecurTotal>1.00</RecurTotal>
    </RecurDetails>
  </RecurUpdateResponse>
</ItransactInterface>
```

## The RecurDetails

This request allows you to query for details on an existing recurring transaction. Currently, this request will return the number of remaining repetitions, the recipe name and total.

**Table 4.5. XMLTrans2.cgi RecurDetails Example**

```
<?xml version="1.0"?>
<ITransactInterface>
  <VendorIdentification>
    <VendorId>XXXXX</VendorId>
    <VendorPassword>PASSWORD</VendorPassword>
    <HomePage>http://www.merchant.com</HomePage>
  </VendorIdentification>
  <RecurDetails>
    <OperationXID>12345</OperationXID>
    <RemReps>123</RemReps>
  </RecurDetails>
</ITransactInterface>
```

## The RecurDetailsResponse

This request will return the following response:

**Table 4.6. XMLTrans2.cgi RecurDetailsResponse Example**

```
<?xml version="1.0" standalone="yes"?>
<ItransactInterface>
  <RecurDetailsResponse>
    <Status>ok</Status>
    <ErrorCategory></ErrorCategory>
    <ErrorMessage></ErrorMessage>
    <TimeStamp>20040621154341</TimeStamp>
    <TestMode>0</TestMode>
    <RecurDetails>
      <RemReps>10</RemReps>
      <RecipeName>daily</RecipeName>
      <RecurTotal>1.00</RecurTotal>
    </RecurDetails>
  </RecurDetailsResponse>
</ItransactInterface>
```

## Placing Recurring Transactions On Hold and Off Hold

Placing a transaction on a temporary hold to prevent billings is simple.

1. Log into the Control Panel and open the Transaction Listing for the day when the original transaction was processed. Locate the transaction that needs to be modified and click on the XID number to open the Transaction Detail request screen.

**Figure 4.21. Transaction Listing Example**

(There may be a 10-15 minute delay before new transactions appear.)

**ACC Live Test Co. (66646) Transaction Report**

Page 1 of 1 First Prev Next Last

Download data set in  format. [Click the XID for transaction details.](#) Estimated Open Batch Total

DATE & TIME	XID	PXID	CXID	ACTION	STATUS	AVS	CVV	TYPE	LAST FOUR	FIRST	LAST	AUTH #	BATCH	AMOUNT	RECUR	OPTIONS
4/17/2007 13:34:47	20840479			Credit	Ok				5454	test	Smith		21	\$5.00		<input type="button" value="PD"/>
4/18/2007 11:14:37	20850033			Order	Ok	NM			5454	Joel	Dare	TAS566	22	\$10.00		<input type="button" value="PD"/>
5/7/2007 13:19:16	21058688		23742992 23755897	Order	Ok	NM			5454	Demo	Buyer	TAS486	25	\$4.00		<input type="button" value="PD"/>
5/16/2007 10:38:16	21158932			Order	Fail				5454	†				\$1.00		<input type="button" value="PD"/>
5/16/2007 10:39:38	21158964			Order	Fail				5454	†				\$1.00		<input type="button" value="PD"/>
5/16/2007 11:08:32	21159428		21206180	Order	Fail				5454	John	Smith			\$10.00		<input type="button" value="PD"/>
5/16/2007 12:27:26	21160635		21160648 21322946	Order	Ok					John	Smith			\$10.00		<input type="button" value="PD"/>
5/16/2007 12:28:01	21160648	21160635		Order	Fail				-	John	Smith			\$10.00		<input type="button" value="PD"/>
5/21/2007 12:17:36	21206009		22257882 23802232 23802235	Order	Ok	NM			5454	Jack	Tester	TAS197	28	\$4.75		<input type="button" value="PD"/>
5/21/2007 12:18:25	21206018		21636413 21626587 21605106 21616373 21647665	Order	Ok	NM			1111	Rick	Tester	TAS218	28	\$10.75		<input type="button" value="PD"/>
5/21/2007 12:19:00	21206024		21949437	Order	Ok	U			0005	John	Demo	AXS565	28	\$10.75		<input type="button" value="PD"/>
5/21/2007 12:23:05	21206070			Order	Fail				8431	Bill	Test-Demo			\$0.20		<input type="button" value="PD"/>
5/21/2007 12:23:55	21206076			Order	Ok	NM			4444	Bill	Test-Demo	TAS259	28	\$5.20		<input type="button" value="PD"/>
5/21/2007 12:24:46	21206088			Order	Ok	NM			1117	Jim	Test	TAS274	28	\$5.20		<input type="button" value="PD"/>
5/21/2007 12:26:02	21206099		21206105	Order	Ok	NM			1117	Jim	Test	TAS291	28	\$5.20		<input type="button" value="PD"/>
5/21/2007 12:26:25	21206105	21206099		Void	Ok				1117	Jim	Test		28	\$5.20		<input type="button" value="PD"/>
5/21/2007 12:28:12	21206133		21206168	Order	Ok	NM			1111	Dave	Tester	TAS996	28	\$10.00		<input type="button" value="PD"/>
5/21/2007 12:29:16	21206151			Order	Ok	NM			5454	Steve	Demotester	TAS318	28	\$10.00		<input type="button" value="PD"/>
5/21/2007 12:29:51	21206157			Order	Ok	NM			5454	Steve	Demotester	TAS562	28	\$6.00		<input type="button" value="PD"/>
5/21/2007 12:30:40	21206168		21206133	Credit	Ok				1111	Dave	Tester		28	\$7.00		<input type="button" value="PD"/>
5/21/2007 12:31:28	21206180	21159428		Order	Ok	NM			5454	John	Smith	TAS590	28	\$7.85		<input type="button" value="PD"/>

Page 1 of 1 First Prev Next Last

2. From the Transaction Detail request screen, click the "Get Detail" button.

**Figure 4.22. Transaction Detail Access Window Example**



3. In the Transaction Detail screen, click on the "No" link in the "On Hold" row of the Recurring Information area.

**Figure 4.23. Transaction Detail Example**

Transaction Detail for XID 32428094

DATE & TIME	XID	P-XID	C-XID	ACTION	STATUS	INSTR	FIRST NAME	LAST NAME	AUTH #	AMOUNT	OPTIONS	RECUR
1/13/2010 10:12:15	32428094		32434070	Order	Ok		Customer	Name	TAS995	\$5.50	<input type="button" value="go"/>	<input type="button" value="go"/>

CUSTOMER INFORMATION:		SHIPPING ADDRESS:	
Name:	Customer Name	Person Name	
Address:	123 Main St	321 Center St	
	BH, CA 90210	BH ,CA 90210	
	USA	USA	
Telephone:	8885551234		
Email:	email@domain.com		
Cust ID:	ABC123		

TRANSACTION DATA:			
Last Four:	5454	AVS Response:	N
Batch:	458	CVV Response:	
Order Total:	\$5.50	IP Address:	65.103.239.179
Transaction Source:	session	Error Message:	

FORM ITEMS:	
<b>Item 1</b>	
cost	5.50
desc	Monthly Payment
qty	1

RECURRING INFORMATION:	
Start Date:	1/13/2010
Recipe Name:	1st
Remaining Reps:	924
Recurring Total:	14.00
On Hold:	No
Recurring History:	<input type="button" value="go"/>

4. A window will pop up to verify that you want to place the transaction on hold. Click "OK". The setting will change from "No" to "Yes". This will temporarily prevent future recurrings (if done anytime prior to 11:59 PM Mountain time on the day before the next scheduled transaction) for as long as you deem necessary.

**Figure 4.24. Recurring Detail Example: Recurring Transaction**

Transaction Detail for **XID 32428094**

DATE & TIME	XID	P-XID	C-XID	ACTION	STATUS	INSTR	FIRST NAME	LAST NAME	AUTH #	AMOUNT	OPTIONS	RECUR
1/13/2010 10:12:15	32428094			Order	Ok		Customer	Name	TAS995	\$5.50	<input type="button" value="go"/>	<input type="button" value="go"/>

CUSTOMER INFORMATION:		SHIPPING ADDRESS:	
Name:	Customer Name	Person Name	
Address:	123 Main St	321 Center St	
	BH, CA 90210	BH ,CA 90210	
	USA	USA	
Telephone:	8885551234		
Email:	email@domain.com		
Cust ID:	ABC123		
TRANSACTION DATA:			
Last Four:	5454	AVS Response:	N
Batch:	458	CVV Response:	
Order Total:	\$5.50	IP Address:	65.103.239.179
Transaction Source:	session	Error Message:	
FORM ITEMS:			
<b>Item 1</b>			
cost	5.50		
desc	Monthly Payment		
qty	1		
RECURRING INFORMATION:			
Start Date:	1/13/2010		
Recipe Name:	1st		
Remaining Reps:	924		
Recurring Total:	14.00		
On Hold:	Yes		
Recurring History:	<input type="button" value="go"/>		

The transaction can be taken off hold following the same steps as above (but by clicking on the "Yes" and toggling it to "No").

## Canceling Recurring Transactions

Taking a transaction out of the recurring cycle is easy to do.

1. Log into the Control Panel and open the Transaction Listing for the day when the original transaction was processed. Locate the transaction that needs to be modified and click on the XID number to open the Transaction Detail request screen.

**Figure 4.25. Transaction Listing Example**

## Setting Transactions To Recur

(There may be a 10-15 minute delay before new transactions appear.)

**ACC Live Test Co. (66646) Transaction Report**

[Print Listing](#)   [Explanation of Codes](#)   [Close Window](#)

Page 1 of 1   [First](#)   [Prev](#)   [Next](#)   [Last](#)

Download data set in		CSV	format.	Click the XID for transaction details.										Estimated Open Batch Total		
DATE & TIME	XID	PXID	CXID	ACTION	STATUS	AVS	CVV	TYPE	LAST FOUR	FIRST	LAST	AUTH #	BATCH	AMOUNT	RECUR	OPTIONS
4/17/2007 13:34:47	20840479			Credit	Ok				5454	test	Smith		21	\$5.00		
4/18/2007 11:14:37	20850033			Order	Ok	NM			5454	Joel	Dare	TAS566	22	\$10.00		
5/7/2007 13:19:16	21058688		23742992 23755897	Order	Ok	NM			5454	Demo	Buyer	TAS486	25	\$4.00		
5/16/2007 10:38:16	21158932			Order	Fail				5454	1				\$1.00		
5/16/2007 10:39:38	21158964			Order	Fail				5454	1				\$1.00		
5/16/2007 11:08:32	21159428		21206180	Order	Fail				5454	John	Smith			\$10.00		
5/16/2007 12:27:26	21160635		21160648 21322946	Order	Fail					John	Smith			\$10.00		
5/16/2007 12:28:01	21160648	21160635		Order	Fail					John	Smith			\$10.00		
5/21/2007 12:17:36	21206009		22257882 23802232 23802235	Order	Ok	NM			5454	Jack	Tester	TAS197	28	\$4.75		
5/21/2007 12:18:25	21206018		21636413 21626587 21605106 21616373 21647665	Order	Ok	NM			1111	Rick	Tester	TAS218	28	\$10.75		
5/21/2007 12:19:00	21206024		21949437	Order	Ok	U			0005	John	Demo	AXS565	28	\$10.75		
5/21/2007 12:23:05	21206070			Order	Fail				8431	Bill	Test-Demo			\$0.20		
5/21/2007 12:23:55	21206076			Order	Ok	NM			4444	Bill	Test-Demo	TAS259	28	\$5.20		
5/21/2007 12:24:46	21206088			Order	Ok	NM			1117	Jim	Test	TAS274	28	\$5.20		
5/21/2007 12:26:02	21206099		21206105	Order	Ok	NM			1117	Jim	Test	TAS291	28	\$5.20		
5/21/2007 12:26:25	21206105	21206099		Void	Ok				1117	Jim	Test		28	\$5.20		
5/21/2007 12:28:12	21206133		21206168	Order	Ok	NM			1111	Dave	Tester	TAS996	28	\$10.00		
5/21/2007 12:29:16	21206151			Order	Ok	NM			5454	Steve	Demotester	TAS318	28	\$10.00		
5/21/2007 12:29:51	21206157			Order	Ok	NM			5454	Steve	Demotester	TAS562	28	\$6.00		
5/21/2007 12:30:40	21206168	21206133		Credit	Ok				1111	Dave	Tester		28	\$7.00		
5/21/2007 12:31:28	21206180	21159428		Order	Ok	NM			5454	John	Smith	TAS590	28	\$7.85		

Page 1 of 1   [First](#)   [Prev](#)   [Next](#)   [Last](#)

- From the Transaction Detail request screen, click the "Get Detail" button.

**Figure 4.26. Transaction Detail Access Window Example**



- In the Transaction Detail screen, click on the "GO" button in the "RECUR" column.

**Figure 4.27. Transaction Detail Example**



Transaction Detail for XID 32428094

DATE & TIME	XID	P-XID	C-XID	ACTION	STATUS	INSTR	FIRST NAME	LAST NAME	AUTH #	AMOUNT	OPTIONS	RECUR
1/13/2010 10:12:15	32428094			Order	Ok		Customer	Name	TAS995	\$5.50	<input type="button" value="go"/>	<input type="button" value="go"/>

CUSTOMER INFORMATION:		SHIPPING ADDRESS:	
Name:	Customer Name	Person Name	
Address:	123 Main St	321 Center St	
	BH, CA 90210	BH ,CA 90210	
	USA	USA	
Telephone:	8885551234		
Email:	email@domain.com		
Cust ID:	ABC123		

TRANSACTION DATA:			
Last Four:	5454	AVS Response:	N
Batch:	458	CVV Response:	
Order Total:	\$5.50	IP Address:	65.103.239.179
Transaction Source:	session	Error Message:	

FORM ITEMS:	
<b>Item 1</b>	
cost	5.50
desc	Monthly Payment
qty	1

RECURRING INFORMATION:	
Start Date:	1/13/2010
Recipe Name:	1st
Remaining Reps:	924
Recurring Total:	14.00
On Hold:	Yes
Recurring History:	<input type="button" value="go"/>

- On this page, click "Edit Setup".

**Figure 4.28. Recurring Detail Example: Recurring Transaction**

## Recurring Setup

### Parent Transaction

**XID:** 103018  
**Date:** 2010-03-24 11:03:52

### Schedule

**Recipe:** Monthly1  
**Interval:** month  
**Next Run:** 2010-04-01  
**Remaining Reps:** 10

## Billing Items

Description	Quantity	Price	Total
Monthly Charge	1	\$5.55	\$5.55

[Edit Setup](#)

## Billing Information

### Contact Info

**Name:** Customer Name  
**Phone:** 8885551234  
**Email:** email@domain.com  
**User ID:** Test123ABC

### Payment Info

**Type:** Card  
**Card Type:** MasterCard  
**Last Four:** 5454  
**Expiration:** 6/2010

### Billing Address

123 Main St  
 BH Ca, 90210  
 USA

### Shipping Address

[Edit Billing Info](#)

[Go Back](#)

[Back To Detail](#)

The Recurring Setup Edit page will open. into the "REPETITIONS" field, delete the value, enter a zero in that field and click the "GO" button in the "UPDATE" column. This will stop future recur-rings (if done anytime prior to 11:59 PM Mountain time on the day before the next scheduled trans-action).

**Figure 4.29. Recurring Setup Edit Page Example: Recurring Transaction**

**Parent Info**

**Type:** Card  
**Card Type:** MasterCard  
**Last Four:** 5454  
**Expiration:** 6/2010

**Contact Info**

**Name:** Customer Name  
**Phone:** 8885551234  
**Email:** email@domain.com  
**User ID:** Test123ABC

**Select Recipe**

Recipe

**Recurring Repetitions**

Repetitions

**Recurring Items**

Use Current Items

	Description	Quantity	Price	Total
✘	<input type="text" value="Monthly Service Fee"/>	<input type="text" value="1"/>	<input type="text" value="\$5.00"/>	\$5.00
✘	<input type="text"/>	<input type="text" value="0"/>	<input type="text" value="\$0.00"/>	\$0.00

Recurring Amount  
 \$5.00

## Setting Scheduled Transactions

Using a scheduled recipe allows you to run all of the transactions linked to a recipe at a date that can be controlled and scheduled manually using the scheduling tool. Merchants can access the scheduling tool in the Recurring Recipe List (See Figure 4.26). In this example, the recipes named "Group 1" and "Group 2" are Scheduled-Type recurring recipes. To open the Scheduling Tool, click on the "GO" button in the "Schedule" column.

**Figure 4.30. Recurring Transaction Recipe Window Example**

RECURRING TRANSACTION RECIPES					
<a href="#">Add Recipe</a> <a href="#">Recurring Help</a>					
EDIT RECIPE	RECIPE NAME	CREATED	DEFINITION	HISTORY	SCHEDULE
<input type="button" value="go"/>	1stday	1/13/2010 13:54:27	Repeat every month on the 1st day	<input type="button" value="go"/>	
<input type="button" value="go"/>	Bimonthly15	1/13/2010 13:55:22	Repeat every 2 months on the 15th day	<input type="button" value="go"/>	
<input type="button" value="go"/>	Daily	1/13/2010 13:54:51	Repeat every day	<input type="button" value="go"/>	
<input type="button" value="go"/>	friday	1/13/2010 13:55:54	Repeat every week on Friday	<input type="button" value="go"/>	
<input type="button" value="go"/>	GROUP1	1/13/2010 13:56:18	Repeat every	<input type="button" value="go"/>	<input type="button" value="go"/>
<input type="button" value="go"/>	GROUP2	1/13/2010 13:56:37	Repeat every	<input type="button" value="go"/>	<input type="button" value="go"/>
<input type="button" value="go"/>	quarterly	1/13/2010 13:57:08	Repeat every 90 days	<input type="button" value="go"/>	
<input type="button" value="go"/>	YEARLY	1/13/2010 13:57:25	Repeat every 365 days	<input type="button" value="go"/>	

1. The Scheduling Tool window will open (See Figure 4.27).

**Figure 4.31. Scheduling Tool Example**

**SCHEDULING RECIPE 9121**

Date:

ITEMS				
#	Description	Quantity	Price	Delete
New	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="button" value="Clear"/>
New	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="button" value="Clear"/>
New	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="button" value="Clear"/>
New	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="button" value="Clear"/>
New	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="button" value="Clear"/>

- Click on the "Show Calendar" button to open the scheduling calendar (See Figure 4.28).

**Figure 4.32. Scheduling Calendar Tool Example**

Jan		2012					
Mon	Tue	Wed	Thu	Fri	Sat	Sun	
						1	
2	3	4	5	6	7	8	
9	10	11	12	13	14	15	
16	17	18	19	20	21	22	
23	24	25	26	27	28	29	
30	31						

- Scroll to the correct month and year and then click on the appropriate day of the month and the "Date" field in the Scheduling Tool will be populated.
- Enter the description, quantity and price into the appropriate fields (See Figure 4.29).

**Figure 4.33. Scheduling Tool Example**

**SCHEDULING RECIPE 9121**

Date:

ITEMS				
#	Description	Quantity	Price	Delete
New	<input type="text" value="June 30th Billing"/>	<input type="text" value="1"/>	<input type="text" value="49.95"/>	<input type="button" value="Clear"/>
New	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="button" value="Clear"/>
New	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="button" value="Clear"/>
New	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="button" value="Clear"/>
New	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="button" value="Clear"/>

- If any modifications need to be made, either click the "Clear" button in the "Delete" column to clear the row, or click the "Reset" button to clear the entire page.
- Once the appropriate schedule is set, click the "Update" button and a success message will display (See Figure 4.30).

**Figure 4.34. Successful Schedule Example**

**SCHEDULING RECIPE 9121**

Date:

ITEMS				
#	Description	Quantity	Price	Delete
1	<input type="text" value="June 30th Billing"/>	<input type="text" value="1"/>	<input type="text" value="49.95"/>	<input type="checkbox"/>
New	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="button" value="Clear"/>
New	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="button" value="Clear"/>
New	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="button" value="Clear"/>
New	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="button" value="Clear"/>
New	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="button" value="Clear"/>

Successful update. 1 item modified.

Please click on Back To Admin below to return to the Admin Page.

## How Does The Customer Recurring Billing Update Tool Work?

Merchants can opt to let their customers update their own credit card billing information for recurring charges. This tool is comprised of several features that make this possible. The *Customer Recurring Billing Update* tool is only available on recurring recipes that have been enabled for "Allow Customer Update" - and can ONLY be used for credit card transactions. This can be enabled for new and existing recipes. When a recurring transaction is successful or fails, an email including a link for the secure update interface will be generated to the cardholder. Following the simple instructions, they can update their own billing information. If the update was necessary because a recurring attempt had failed, a transaction will be attempted to bring the missed recurring billing up to date. Most merchants choose to use this feature in conjunction with the "Hold on Failure" recurring function - so that when a transaction fails, the recurring is put on hold (to prevent future attempts), and the update link is sent out. A merchant can also resend confirmation emails including the link to their customers from the Transaction Options area of the Transaction Listing. When a customer follows the link, they will be taken to this page:

**Figure 4.35. Customer Recurring Billing Update Verification Example**

The screenshot shows the iTransact, Inc. website interface for editing automatic payments. At the top, there is a green header with the iTransact logo and the text "Automatic Payments". Below this, the main heading is "Edit Automatic Payments for Carl's Computers". To the left of the text is a small icon for Carl's Computers. The text explains that Carl's Computers processes automatic payments through iTransact and asks the user to authenticate by answering two questions. Below this, contact information for Carl's Computers is provided: "123 A St, Salt Lake City UT, 84103, 888.555.1234". A section titled "Enter At Least Two Of The Following Four Values" contains four input fields: "First Name", "Last Name", "ZIP Code", and "Address". Below these fields is a yellow "Authenticate" button. At the bottom right, there is a copyright notice for iTransact, Inc. and links for Home, Legal Notice, and Privacy Policy.

**iTransact, Inc.**  
Automatic Payments

### Edit Automatic Payments for Carl's Computers

Carl's Computers processes automatic payments for its services through iTransact. To set up or modify your automatic payments with Carl's Computers, please authenticate by correctly answering two of the questions below.

If you wish to discontinue automatic payments or if you have any questions, please contact:

Carl's Computers  
123 A St  
Salt Lake City UT, 84103  
888.555.1234

#### Enter At Least Two Of The Following Four Values

First Name

Last Name

ZIP Code

Address

**Authenticate**

Copyright © iTransact, Inc. [Home](#) | [Legal Notice](#) | [Privacy Policy](#)  
iTransact, Inc., is a registered ISO/MSP of Wells Fargo Bank, N.A., Walnut Creek, CA

This verification page is a feature used to verify that the person accessing the Customer Recurring Billing Update Tool is using the correct information for a specific transaction. If the customer enters valid verification information, they will be taken to the following page

**Figure 4.36. Customer Recurring Billing Update Interface Example**

**iTransact, Inc.**  
Automatic Payments

### Edit Automatic Payments for Carl's Computers

Carl's Computers processes automatic payments for its services through iTransact. To set up or modify your automatic payments with Carl's Computers, please select the billing method of choice, and then enter your information.

If you wish to discontinue automatic payments or if you have any questions, please contact:

Carl's Computers  
123 A St  
Salt Lake City UT, 84103  
888.555.1234

Carl's Computers account: ABC1234  
Periodic Amount: \$5.00  
Payment frequency: day  
Current payment method: MasterCard 5454 (ex 8/2015)

Edit your payment information

<p><b>Enter card information</b></p> <p>Cardholder's name <input type="text" value="Customer Name"/></p> <p>Card number <input type="text"/></p> <p>Expiration date month / year <input type="text" value="01"/> / <input type="text" value="2010"/></p> <p>CVV Security number <input type="text"/></p>	<p><b>Enter billing address</b></p> <p>Street name <input type="text" value="123 Main St"/></p> <p>City <input type="text" value="Test Town"/></p> <p>State <input type="text" value="CA"/></p> <p>Zip code <input type="text" value="90210"/></p> <p>Country <input type="text" value="USA"/></p> <p>Email address (to send confirmation) <input type="text" value="email@domain.com"/></p>
--	--

Copyright © iTransact, Inc. [Home](#) | [Legal Notice](#) | [Privacy Policy](#)  
iTransact, Inc., is a registered ISO/MSP of Wells Fargo Bank, N.A., Walnut Creek, CA

## Customer Recurring Billing Update Tool Details

The *Allow Customer Update* feature makes it possible for a merchant to allow cardholders to update their own recurring billing information for credit card transactions attached to recipes where the Customer Billing Update option is enabled. The *Recurring On-Hold* feature provides the ability to put future recurring attempts "on hold" temporarily without removing the value of the remaining repetitions. Proactive cardholders will also be able to update their own card information without requiring the merchant to have to manually make the changes themselves. Merchants will also be able to temporarily stop specific recurring transactions from attempting.

When the *Allow Customer Update* feature is in use, a declined or failed recurring transaction will generate an email to the cardholder instructing them to click a link which will take them to a secure web page where they will be able to update their billing information. The recurring customer confirmation emails for successful transactions will also include a link for updating card information if they want to update the account information that is being used. Additionally, the merchant will be notified of the update link via the "*billing\_update\_token*" in the Recurring Postback, so that if they'd like to be able to provide the customer with the update access through their own system, they'll have the correct security token.

The *Recurring On-Hold* feature allows a merchant to toggle recurring on and off for specific transactions, without changing the number of remaining repetitions for that account. This can be used to prevent future attempts against a recurring transaction temporarily. If the *Hold On Failure* function of the *Recurring On-Hold* feature is enabled in a recipe, a failed recurring credit card billing will place a transaction on-hold to prevent future billing attempts so that account information can be updated (either by the merchant or the customer). When either of these features are triggered by a failed transaction, the merchant will be notified by email in the Recurring Failure email and in the Recurring Postback (using the "*billing\_update\_token*" and "*on\_hold*" parameters). A "Lookup" value for the "*billing\_update\_token*" designating the Customer Billing Update page will be available on a successful recur-initiating transaction for recipes where *Allow Customer Update* is enabled. If a merchant chooses to display the update link on their end, the value of any "*billing\_update\_token*" can be appended to

[https://secure.itransact.com/customers/billing\\_update/](https://secure.itransact.com/customers/billing_update/)



For example, a link using a token with a value of "abc123" would be

*[https://secure.itransact.com/customers/billing\\_update/abc123](https://secure.itransact.com/customers/billing_update/abc123)*

When an update is made to a transaction that is not "on hold", and AVSOnly or Preauth transaction is attempted to validate the updated information.

## Activating These Features

*Recurring On-Hold* is available for any recurring transaction. This can be toggled in the Transaction Details for a specific transaction or through the Recurring History for a specific transaction in the Recurring Transactions interface. Those interfaces display the "On Hold" toggle field with a link to toggle on (YES) or off (NO) the hold setting.

*Allow Customer Update* and *Hold on Failure* can be enabled when a recurring recipe is created in the Recipe Builder by selecting the checkboxes. These features can be enabled independent of each other. If you would like to activate this for existing recipes, you may edit your recipes by logging into your Control Panel, accessing the Recurring Transaction area, and using the "EDIT" feature for the necessary recipes. You will see the checkboxes that you can enable. Activate the desired features and click the "Save Recipe" button.

## How Can I Run A Form-Based Recurring Transaction Update?

### Introduction

This module provides a seamless way for a merchant's website to allow a customer to update their recurring billing information. Depending on the 'on hold' status of the recurring transaction it will also catch a customer up by running a sale transaction. This reduces the amount of work that is required for a merchant to maintain their recurring transactions.

A form is hosted on the merchant's website which allows the customer to enter new billing information. This would most likely be done within a customer interface that requires a login. This form is posted directly to iTransact's site so that the merchant's servers never touch card information reducing PCI compliance issues. For added security, the credentials for this request includes the use of a Payload Signature.

This process appears seamless to customers because of the postback mechanism. All authenticated requests to the module result in a post to the merchant's website to render HTML to be displayed in the browser. This post is done inside of the request thread, and the results of the post are written to the active output stream. In addition to the required fields, 'passback' input fields can be provided so that the merchant's scripts can pass through session identifiers that will be returned in the postback.

### Process Overview

#### Request Authentication

Requests are authenticated using a payload signature. For this module, the only request parameter used for the payload is 'xid'. The reason for this is this module is intended to be used in a browser environment. Requiring any of the other data elements to be used in the payload signature would require the signature to be generated using Javascript or a background http call. This would add un-needed complexity to the module.

The standard use scenario would be for a merchant to generate the payload signature while rendering the customer update form.

Authentication failures will not result in a postback attempt. This includes requests that have no authen-

tication fields, have invalid postback URLs or that have invalid payload signatures. If authentication fails, a page will be rendered which indicates that there was an authentication failure. This page is non-branded and will simply notify the customer that their attempt did not go through.

## Request Validation

Once the request is authenticated, the input fields are validated. If there are any field validation errors, a postback is generated to the error postback URL which includes field specific error information. This allows the postback script to render a form with specific field errors.

card\_number, account\_number and cvv values should be validated client-side using JavaScript if possible since we will not postback these values. If invalid values result in an error postback, then the customer will have to enter these values again. There are several free JavaScript implementations of the Luhn 10 card validation algorithm. We have included some options below although we do not endorse or provide technical support for their use.

- <http://blog.planzero.org/2009/08/javascript-luhn-modulus-implementation/>
- [http://github.com/madrobby/creditcard\\_js](http://github.com/madrobby/creditcard_js)
- <http://javascript.internet.com/forms/val-credit-card.html>

## Billing Info Validation/Recurring Authorization

If a request passes all validations, the module will then look at the 'on hold' state of the recurring transaction. If the transaction is on hold, a Sale transaction will be attempted with the provided billing information. If the transaction is not on hold and card information is being provided, a \$1.00 PreAuth or an AVSOnly transaction will be attempted which will validate the card information provided. If check information is being provided the module will not try any account authentication. The type of transaction used for cards will be dependent on whether AVSOnly is supported for the merchant's processing network. AVSOnly is the preferred method for this type of transaction and is the default. If the request results in a successful Sale transaction, the recurring transaction's 'on hold' state is cleared and the number of remaining recurring repetitions is reduced by one.

If this transaction is not successful a postback is generated to the error postback URL. The presence of either 'internal\_err' or 'proc\_err' designates what type of error occurred. A processor error is normally displayed to the customer so that they have feedback about the problem with the card they attempted to use.

## Email Notification

Depending on the merchant's recurring customer email setting, a confirmation email is sent to the customer for any update that is successful. The email clearly indicates whether or not a sale transaction took place. A confirmation email is always sent to the merchant's 'order' email address on success.

## Interface Postback

All authenticated requests to the module result in a post to the merchant's website to render HTML to be displayed in the browser. This post is done inside of the request thread, and the results of the post are written to the active output stream.

Currently, only one postback attempt is made. If this request fails, a page will be rendered which has basic information useful to the customer. This page is non-branded and will simply notify the customer that their attempt did/didn't go through and indicates that there was a system error. It is the responsibility of the merchant to ensure that their postback pages are up and functional. We recommend using a tool like Nagios or Zenoss to monitor your applications.

## Background Postback

If an update is successful, an additional postback call can be made. This request is sent to the recurring postback URL specified in the merchant's control panel and is run in the background. It is only sent if the merchant has also enabled the recurring postback feature. It will be attempted every 10 minutes for an hour in case the postback address is unresponsive.

## Module Interface

Request content should conform to the 'application/x-www-form-urlencoded' content type. This means that the module only accepts POST requests. If a GET request is initiated against the model a 405 error will be returned with an 'Allow' header with the value of 'POST'. The 'content-type' header should also be specified in the request.

The Module URL is: [https://secure.itransact.com/merchant\\_api/billing\\_update](https://secure.itransact.com/merchant_api/billing_update)

## Payload Signature Generation

### Overview

To generate the `payload_signature` input parameter you are going to generate a string that looks like a HTTP parameter set and sign this string using HMAC-SHA1. You need to sign the `XID`. The HMAC-SHA1 algorithm generates a result that is not web friendly so you have to then Base64 encode it so it can be sent through the request.

### Signature Process

For this section we are going to assume that we are going to issue a request against `XID 99999999`. We have been issued API credentials with a key of `12345678901234567890`.

### Generate The Payload

Note that although we are generating something that looks like part of a HTTP request URL we don't need to URI encode it. The request should be a string without any carriage return or newline characters.

```
xid=99999999
```

### Sign The Payload

After Base64 encoding the output of our HMAC-SHA1 library we end up with a signature of `'QAn-VUdQjREN2hbcsKAok7Ma7/SM='`

## Input Fields For Form-Based Recurring Billing Update Request

The address to POST these request to is [https://secure.itransact.com/merchant\\_api/billing\\_update](https://secure.itransact.com/merchant_api/billing_update)

These are the parameters to use in the POST to the iTransact gateway. All fields are required unless indicated as optional.

**Table 4.7. Recurring Billing Update Input Fields**

Value	Description	Billing Type	Required
aba	Nine digit numeric ABA routing number	Check	Yes (checks)

Setting Transactions To Recur

Value	Description	Billing Type	Required
account_number	Bank account number	Check	Yes (checks)
account_source	Either 'checking' or 'savings'. Default value is checking. This is not available for redicheck processing.	Check	No
account_type	Either 'personal' or 'business'. This is valid only for CheckGateway processing	Check	No
addr	Customer address. Up to 100 characters	Card, Check	Yes
api_username	As displayed in the Gateway Control Panel's Account Settings	Card, Check	Yes
card_number	Credit card account. 14-16 digit numeric value must validate to Luhn algorithm	Card	Yes (cards)
city	Customer city. Up to 25 characters	Card, Check	Yes
ctry	Customer country. Up to 45 characters	Card, Check	Yes
cvv	Cardholder verification value. 3-4 characters ( <i>OPTIONAL</i> )	Card	No
email	Must be a valid email address in standard email format. Up to 255 characters	Card, Check	Yes
error_ret_addr	Must be absolute https address used for Postback URL location for field validation, authentication errors or declined attempts.	Card, Check	Yes
exp_month	Credit card expiration month. Two digit numeric month value [can be 01...12].	Card	Yes (cards)
exp_year	Credit card expiration year. Four digit numeric year value.	Card	Yes (cards)
first_name	Customer first name. Up to 50 characters.	Card, Check	Yes
last_name	Customer first name. Up to 50 characters.	Card, Check	Yes
passback[]	This field allows you to pass additional custom fields that will come through in the return post. Include the open and closed brackets [] characters with each passback request. There should be one input field named 'passback[]' for every input field that you want to have returned to your application. See the passback example below.	Card, Check	No
payload_signature	HMAC-SHA1 signature of the XID parameter	Card, Check	Yes
state	Cardholder state. Up to 25 characters	Card, Check	Yes
sec_code	Standard Entry Category. 3 characters	Check	No
success_ret_addr	Must be absolute https address used for Postback URL location for successful updates.	Card, Check	Yes
xid	The transaction ID (originating or Parent XID) tied to the recurring transaction.	Card, Check	Yes

Value	Description	Billing Type	Required
zip	Customer postal code. Up to 20 characters	Card, Check	Yes

**Table 4.8. Passback[] Examples**

```
<input type='hidden' name='myfield1' value='data1'>
<input type='hidden' name='myfield2' value='data2'>
<input type='hidden' name='passback[]' value='myfield1'>
<input type='hidden' name='passback[]' value='myfield2'>
```

The following is an example of a request being generated from an HTML page.

**Table 4.9. Form-Based Recurring Billing Update Example - Card**

```
<form method=post action="https://secure.itransact.com/merchant_api/billing_update">
<!-- API Key for example is: 9pMx5z2G246W5vwSZ9Et -->
<p>
API Username <input type=text name=api_username value="testaccount_XXXXXXXX">
<p>
Payload Sig <input type=text name=payload_signature value="PUly2kg117rM1+fsEH1mFoZxOe8=">
<p>
XID <input type=text name=xid value="101166">
<p>
Error Return Address <input type=text name=error_ret_addr value="https://www.domain.com/errscript.">
<p>
Success Return Address <input type=text name=success_ret_addr value="https://www.domain.com/script.">
<p>
First Name <input type=text name=first_name value="John">
<p>
Last Name <input type=text name=last_name value="Smith">
<p>
Email <input type=text name=email value="email@domain.com">
<p>
Address <input type=text name=addr value="123 Main ST">
<p>
City <input type=text name=city value="Bountiful">
<p>
State <input type=text name=state value="UT">
<p>
Zip <input type=text name=zip value="84010">
<p>
Country <input type=text name=ctry value="USA">
<p>
Card Number <input type=text name=card_number value="5454545454545454">
<p>
Exp Month <input type=text name=exp_month value="11">
<p>
Exp Year <input type=text name=exp_year value="2020">
<p>
CVV <input type=text name=cvv value="123">
</P>
<br>
<input type=submit name=go value="Process">
</center>
</form>
```

**Table 4.10. Form-Based Recurring Billing Update Example - Check**

```

<form method=post action="https://secure.itransact.com/merchant_api/billing_update">
<!-- API Key for example is: 9pMx5z2G246W5vwSZ9Et -->
<p>
API Username <input type=text name=api_username value="testaccount_XXXXXXXX">
<p>
Payload Sig <input type=text name=payload_signature value="PUly2kg117rM1+fsEH1mFoZxOe8=">
<p>
XID <input type=text name=xid value="101166">
<p>
Error Return Address <input type=text name=error_ret_addr value="https://www.domain.com/errscript.">
<p>
Success Return Address <input type=text name=success_ret_addr value="https://www.domain.com/script.">
<p>
First Name <input type=text name=first_name value="John">
<p>
Last Name <input type=text name=last_name value="Smith">
<p>
Email <input type=text name=email value="email@domain.com">
<p>
Address <input type=text name=addr value="123 Main ST">
<p>
City <input type=text name=city value="Bountiful">
<p>
State <input type=text name=state value="UT">
<p>
Zip <input type=text name=zip value="84010">
<p>
Country <input type=text name=ctry value="USA">
<p>
ABA <input type=text name=aba value="124000054">
<p>
Account Number <input type=text name=account_number value="12345">
<br>
<input type=submit name=go value="Process">
</center>
</form>

```

## Recurring Billing Update Responses

### Recurring Billing Update Response Parameters

These are the parameters that can be included in a recurring billing update response.

**Table 4.11. Standard Recurring Billing Update Response Fields**

Field	Description	Billing Type	Responses
aba	Nine digit numeric ABA routing number	Check	All
account_source	Either 'checking' or 'savings'. Default value is checking. This is not available for redicheck processing.	Check	All
account_type	Either 'personal' or 'business'. This is valid only for CheckGateway processing and only populated if passed through in request.	Check	All
addr	Cardholder address. Up to 100 characters	Card, Check	All
authcode	The approval number issued by card	Card	Success

Setting Transactions To Recur

Field	Description	Billing Type	Responses
	processor/issuer. Up to 8 alpha-numeric characters		
avs_category	Adress verification response category. See table below for possible return values.	Card	Success
avs_response	The raw address verification response code from processor.	Card	Success
card_name	The credit card type submitted.	Card	All (If determinable)
city	Account holder city. Up to 25 characters	Card, Check	All
ctry	Account holder country. Up to 45 characters	Card, Check	All
cvv_response	The CVV verification response from processor.	Card	Success
email	Account holder email address in standard email format. Up to 255 characters	Card, Check	All
err_message	This will be included in any post to the error_ret_addr for any validation, processor, or auth type errors.	Card, Check	Error
exp_month	Credit card expiration month. Two digit numeric month value [can be 01...12].	Card	All
exp_year	Credit card expiration year. Four digit numeric year value.	Card	All
first_name	Account holder first name. Up to 50 characters	Card, Check	All
internal_err	If an internal error occurred while processing the transaction, this value will be "1". It will be blank otherwise.	Card, Check	All
last_four	The last four digits of the credit card account submitted.	Card	All
last_name	Account holder lastst name. Up to 50 characters	Card, Check	All
payload_signature	HMAC-SHA1 signature of the all included parameters signed using merchant's API Key.	Card, Check	All
proc_err	If a processor type error takes place, this value will be "1". It will be blank otherwise.	Card, Check	All
sec_code	Standard Entry Category. 3 characters	Check	All
state	Account holder state. Up to 25 characters	Card, Check	All
total	The dollar amount of the sale transaction or validation. If a preauth was run then this would be '1.00'. If an AVS Only was run then the value will be '0.00'.	Card, Check	All
trans_xid	The Transaction ID of the transaction that was run during the update.	Card, Check	All (If determinable)
xid	The transaction ID (originating or Parent	Card, Check	All

Field	Description	Billing Type	Responses
	XID) tied to the recurring transaction.		
validation_err	If validation type error takes place, this value will be "1". It will be blank otherwise. If equal to "1", there will be a Field Validation Error Indicator included in the response.	Card, Check	All
zip	holder postal code. Up to 20 characters	Card, Check	All
Passback[] Fields	Any values passed as Passback[] variables will be included	Card, Check	All

These are the potential values for avs\_category that may be included in a Recurring Billing Update Response for a credit card update.

**Table 4.12. Potential AVS\_Category Values - Recurring Billing Update Response Values**

Value	Description
address	Street address matched
address_postal	Street address and postal code matched
address_zip5	Street address and five digit postal code matched
address_zip9	Street address and nine digit postal code matched
address_ok_postal_format_error	Street address matched, postal code format error
global_non_participant	International with no AVS support
international_address_not_verified	International with no AVS support
no_match	No street address or postal code match
no_response	No response
not_allowed	Not allowed
postal	Postal code matched
postal_ok_address_format_error	Postal code matched, street address format error
service_not_supported	AVS service not supported for card
unavailable	AVS service not supported for card
zip5	Five digit postal code matched
zip9	Nine digit postal code matched

These are the field validation response parameters than may be included in a Recurring Billing Update Response.

**Table 4.13. Field Validation Error Indicators - Recurring Billing Update Response Fields**

Field	Description	Billing Type
aba_err	Indicates a validation error with the aba field.	Check
account_number_v_err	Indicates a validation error with	Check



Setting Transactions To Recur

Field	Description	Billing Type
	the account_number field.	
account_source_v_err	Indicates a validation error with the account source field.	Check
account_type_v_err	Indicates a validation error with the account type field.	Check
addr_v_err	Indicates a validation error with the address field.	Card, Check
card_number_v_err	Indicates a validation error with the credit card account field.	Card
city_v_err	Indicates a validation error with the city field.	Card, Check
ctry_v_err	Indicates a validation error with the country field.	Card, Check
cvv_v_err	Indicates a validation error with the CVV field.	Card
email_v_err	Indicates a validation error with the email field.	Card, Check
exp_month_v_err	Indicates a validation error with the expiration month field.	Card
exp_year_v_err	Indicates a validation error with the expiration year field.	Card
first_name_v_err	Indicates a validation error with the first name field.	Card, Check
last_name_v_err	Indicates a validation error with the last name field.	Card, Check
sec_code_v_err	Indicates a validation error with the SEC code field.	Check
state_v_err	Indicates a validation error with the state field.	Card, Check
success_ret_addr_v_err	Indicates a validation error with the success_ret_addr field.	Card, Check
xid_v_err	Indicates a validation error with the transaction ID field.	Card, Check
zip_v_err	Indicates a validation error with the postal code field.	Card, Check

---

# Chapter 5. The Recurring Transaction History

## Viewing The Transaction History

Figure 5.1. Recurring Transaction Recipe Window Example

RECURRING TRANSACTION RECIPES					
<a href="#">Add Recipe</a> <a href="#">Recurring Help</a>					
EDIT RECIPE	RECIPE NAME	CREATED	DEFINITION	HISTORY	SCHEDULE
<input type="button" value="go"/>	1stday	1/13/2010 13:54:27	Repeat every month on the 1st day	<input type="button" value="go"/>	
<input type="button" value="go"/>	Bimonthly15	1/13/2010 13:55:22	Repeat every 2 months on the 15th day	<input type="button" value="go"/>	
<input type="button" value="go"/>	Daily	1/13/2010 13:54:51	Repeat every day	<input type="button" value="go"/>	
<input type="button" value="go"/>	friday	1/13/2010 13:55:54	Repeat every week on Friday	<input type="button" value="go"/>	
<input type="button" value="go"/>	GROUP1	1/13/2010 13:56:18	Repeat every	<input type="button" value="go"/>	<input type="button" value="go"/>
<input type="button" value="go"/>	GROUP2	1/13/2010 13:56:37	Repeat every	<input type="button" value="go"/>	<input type="button" value="go"/>
<input type="button" value="go"/>	quarterly	1/13/2010 13:57:08	Repeat every 90 days	<input type="button" value="go"/>	
<input type="button" value="go"/>	YEARLY	1/13/2010 13:57:25	Repeat every 365 days	<input type="button" value="go"/>	

To view all of the customers using a specific recipe, view the Recurring Recipe list, choose the recipe in question, and click the "GO" button in the "HISTORY" column (See Figure 5.1).

Figure 5.2. Recurring Recipe History Example

Transactions Associated with Recipe 'everytuesday'

DATE & TIME	XID	ON HOLD	FIRST	LAST	ORIG AMOUNT	RECUR AMOUNT	RR	HISTORY	EDIT	EDIT
3/19/2009 15:20:44	<u>29210880</u>	<b>Yes</b>	kevin	test	\$1.00	\$1.00	22	<input type="button" value="go"/>	<input type="button" value="Customer"/>	<input type="button" value="Items"/>
10/29/2009 11:36:30	<u>31644975</u>	<b>No</b>	Matt	Try	\$6.00	\$6.00	5	<input type="button" value="go"/>	<input type="button" value="Customer"/>	<input type="button" value="Items"/>
5/30/2007 14:08:17	<u>21292828</u>	<b>No</b>	John	Smith	\$10.00	\$10.00	0	<input type="button" value="go"/>	<input type="button" value="Customer"/>	<input type="button" value="Items"/>

\* RR = Remaining repetitions

Records per page:

The "Yes" or "No" value in the "On Hold" column is a toggle switch that can be use to place a recurring billing on hold or vice versa. The red "Yes" indicates that a transaction is on hold and will not attempt. The green "No" indicates that a transaction is not on hold and will attempt to bill according to the recipe.

To view all of the XIDs for a specific customer, click on the "GO" button in the "HISTORY" column and a page listing the history for a specific customer will display (See Figure 5.2).

**Figure 5.3. Recurring Transaction History Example**

**XID 30706373  
Recurring Transaction History**

DATE & TIME	TYPE	XID	STATUS	AMOUNT
8/6/2009 08:09:27	Order	<u>30706373</u>	Ok	\$10
9/1/2009 12:21:11	Order	<u>30989688</u>	Ok	\$10.00
Remaining repetitions = 0				

This feature (See Figure 5.3) will allow you to view a full list of all transactions run for a customer using the recipe.

---

# Chapter 6. The Recurring PostBack Feature

## Recurring PostBack Feature

Merchants using the recurring transaction features of the gateway may specify a URL to receive transaction postback information. This can be enabled via the "Account Settings" link in the Control Panel. To use this feature, enter the URL to be used for postback information. Then select the check-box for recurring. Each time a recurring transaction is processed through the gateway system, the transaction server will post the transaction results to the designated postback URL. The fields listed below will be available when the postback function is used. Please note that all fields will be posted even if they do not have a value.

- **xid** - This is the value for the transaction ID assigned by the gateway.
- **authcode** - This is the authorization code issued by the credit card processor.
- **avs\_response** - The value of this is the response received from the address verification system.
- **cc\_last\_four** - These are the last four digits of the account number.
- **cc\_name** - This identifies the card type used.
- **cvv2\_response** - The value of this is the response received from the card verification value system.
- **trans\_type** - This should be listed as "order".
- **when** - This is a time stamp in format of "20010509134443" - meaning 05/09/2001 at 13:44:43.
- **status** - This indicates the validity of a transaction. The following responses are possible: "ok", "error", "fail", "begun".
- **error\_message** - If a transaction fails, this value will be the error response.
- **recipe\_name** - This is the name of the recurring recipe being used for the transaction.
- **recipe\_every** - This indicates the how often during a recurring period a transaction is to recur.
- **recipe\_period** - The value for this is "day", "week", "month" or "scheduled".
- **orig\_xid** - This value is the transaction ID for the originating/operating/Parent transaction.
- **rem\_reps** - This shows the number of times that a recurring transaction is set to cycle.
- **start\_date** - This is the date of when the transaction was set as a recurring transaction.
- **\*-desc** - This value is the description of the order item. \* indicates the item number.
- **\*-cost** - This value is the cost of the order item. \* indicates the item number.
- **\*-qty** - This value is the quantity of the order item. \* indicates the item number.
- **\*-X** - This value is an attribute of the order item. \* indicates the item number. X is a user specified item attribute.

- **recur\_desc** - This will have a value if the merchant has passed a recur\_desc field or if the field was modified in the recurring system.
- **recur\_total** - This will have a value if the merchant has passed a recur\_total field or if the field was modified in the recurring system.
- **first\_name** - This is the customer's first name.
- **last\_name** - This is the customer's last name.
- **address** - This is the customer's billing address.
- **city** - This is the customer's billing city.
- **state** - This is the customer's billing state.
- **zip** - This is the customer's billing postal code.
- **ctry** - This is the customer's country.
- **email** - This is the email address entered by the customer.
- **phone** - This is the phone number submitted with the order.
- **saddr** - This is the shipping address.
- **scity** - This is the shipping city.
- **sctry** - This is the shipping country.
- **sfname** - This is the first name of the person which will receive the shipment.
- **slname** - This is the last name of the person which will receive the shipment.
- **sstate** - This is the shipping state.
- **szip** - This is the shipping postal code.
- **cust\_id** - This is the merchant's optional custom id field passed with the transaction.
- **billing\_update\_token** - This parameter is included if the recipe has "Allow Customer Update" enabled. It is the value of the token to be used in a link to the secure billing update page. Should be 60-80 characters.
- **on\_hold** - This parameter is included if the recipe has "Hold on Failure" enabled. This will show a "1" for yes (it's been put on hold) or a "0" for no (it is no longer on hold).