MerchantPartners

Online Commerce Suite™ Integration Guide

Release 5.6

October 2004

MerchantPartners.com LLC
12424 Wilshire Blvd., Suite 1170
Los Angeles, CA 90025
www.merchantpartners.com

© 2003 MerchantPartners.com LLC   All Rights Reserved

Contents

What's In the Guide

The Online Commerce Suite Integration Guide is specialized to meet the needs of your operational, technical, and accounting staff. This guide will help your users to quickly familiarize themselves with the application to make full use of its many powerful tools and to maximize the profitability of your e-commerce operation.

Introduction

Welcome to the Online Commerce Suite system!

Online Commerce Suite is a Web-based payment gateway that allows you to process secure credit card and electronic check payments for goods and services over the Internet. Using the Online Merchant Center™ web-based administrative user interface, you can configure your Online Commerce Suite account, add users, and manage your e-business. Online Commerce Suite provides a comprehensive set of online and downloadable transaction management and accounting reports.

For more information about Online Commerce Suite, refer to the following:

Customer Service:

Phone:	866-242-9933
Email:	customerservice@merchantpartners.com

Documentation:

Web Site:	http://www.onlinemerchantcenter.com/docs/

Online Merchant Center web site:

Web Site:	https://www.onlinemerchantcenter.com

Online Commerce Suite Payment Gateway:

URL:	https://trans.merchantpartners.com/cgi-bin/process.cgi

Online Commerce Suite FTP site:

URL:	ftp.atsbank.com

Setting Up an Online Commerce Suite Account

The first step in setting up your Online Commerce Suite account is to contact Customer Service to complete your registration by telephone. When your account is confirmed and set up, you will receive your five-digit Online Commerce Suite Account ID number (Acct ID) This important number identifies your account in the Online Commerce Suite system and allows the system server to authenticate transactions originating from you. Be sure to include your Online Commerce Suite Acct ID number in all correspondence with Customer Service.

See the companion Manual ***Getting Started with Online Commerce Suite*** for more information on setting up your account.

Integrating your System with Online Commerce Suite

Your e-commerce goals determine the transaction processing method you will use with Online Commerce Suite. Various factors make a difference how you integrate Online Commerce Suite with your e-commerce business, whether you offer products or subscriptions or both. Factors include:

- Do you use Shopping Cart software?
- Did you develop your own Web site?
- Is your Web site is hosted on a secure server?
- Do you want Online Commerce Suite to calculate shipping and tax?
- Do you have your own database to track your inventory and business?

The answers to these questions determine the complexity of your integration with Online Commerce Suite. Depending on your requirements, integration can be very straightforward or may require a sophisticated understanding of HTML, CGI, ASP or other Web technologies.

Integration Overview

The Online Commerce Suite system processes secure on-line credit cards and electronic check payments. The authorization network secures credit card payment authorizations while the automated clearinghouse (ACH) network processes electronic checks. Online Commerce Suite automatically selects the appropriate payment network based upon the requested transaction type (credit card or ACH) and the authorizing network or Federal Reserve Bank region supported by the bank where you have a merchant account. Figure 1 is a conceptual schematic of the Online Commerce Suite system.

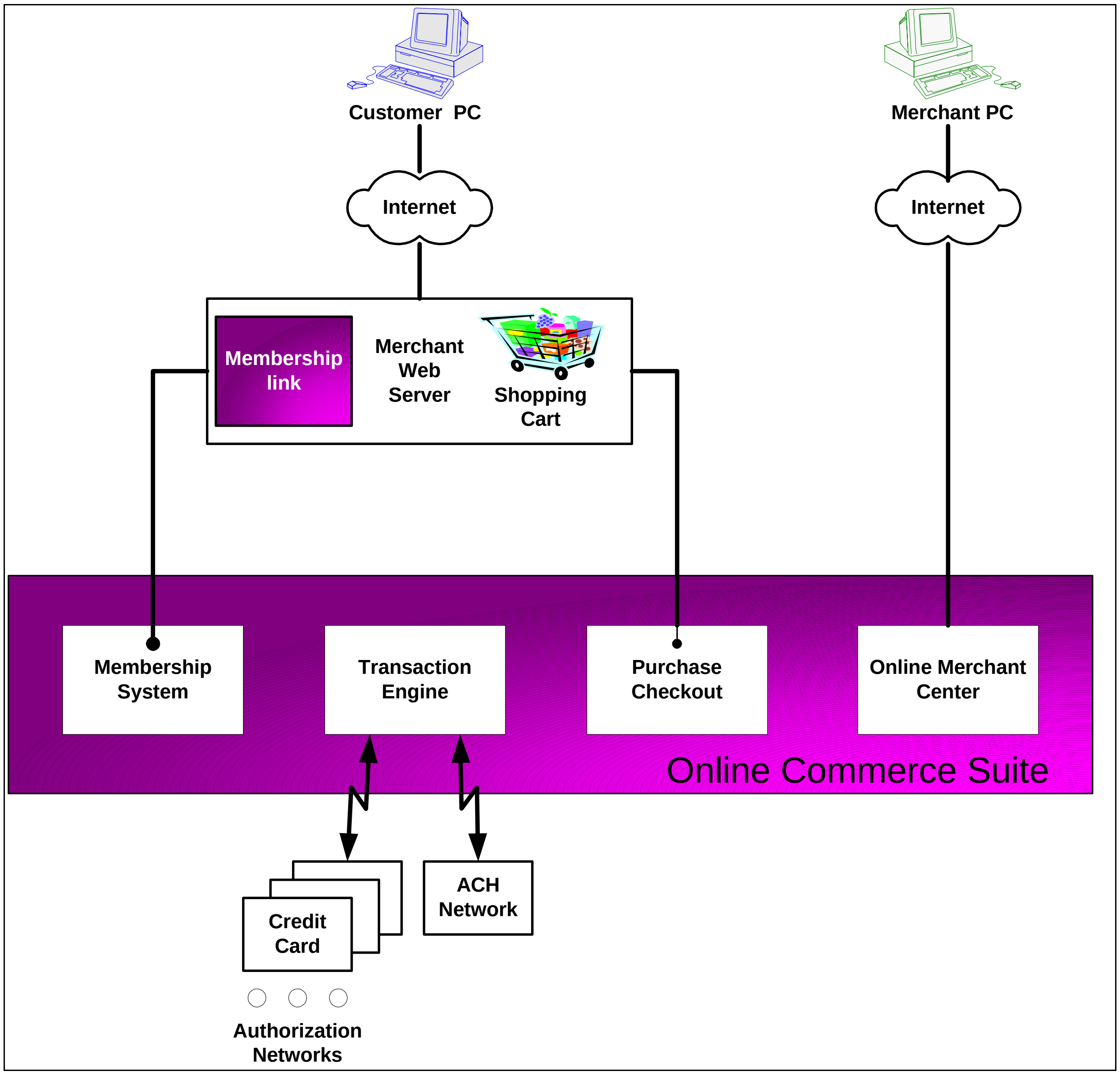


Figure 1 – Online Commerce Suite system

Consumer PC

The on-line customer accesses your Merchant Web Server through the Internet to initiate the shopping experience. Typically, items are selected for purchase and placed in a virtual Shopping Cart. When the customer's purchase decisions are completed and confirmed, the Web Link module is activated. Web Link uses preloaded parameters like price, appropriate tax rate, item weight, and available shipping options to calculate the total amount due for the transaction. This calculated charge is passed back to the on-line customer as an invoice for their approval. The approved invoice activates the Transaction Engine to secure payment authorization through the appropriate authorization network.

Merchant Web Server

The Merchant Web Server supports the customer's shopping and purchasing experiences. It may host its own Shopping Cart and link to the Web Link and Transaction Engine modules for payment processing.

Shopping Cart

Shopping Cart programs are used to collect customer purchase and product pricing information. The Shopping Cart program resides on the Merchant's web server and is typically provided by a third party or ISP. Online Commerce Suite also provides a hosted shopping cart solution (the Web Link) for those merchants who do not have a shopping cart or do not want to host their own cart. The Transaction Engine uses the information collected by the Shopping Cart to determine the amount for account debit authorizations.

Merchant PC

The Merchant connects to the Online Merchant Center module through the Internet. Using this interface, the Merchant may collect transaction, accounting, and customer information reports.

Online Commerce Suite System Modules

Online Commerce Suite modules accomplish the following functions:

Web Link completes the customer purchase experience by collecting payment information from the shopping cart, calculating the amount to be charged, and passing this information to the Transaction Engine for processing.

Transaction Engine processes online, batch, recurring, and membership subscription payment authorization requests. Transactions are processed immediately in real time or in batch. Within seconds, consumers receive an acceptance or decline notification. Funds from accepted credit card transactions are deposited into your merchant bank account, typically within 24 hours. Funds from accepted electronic check (ACH) transactions are deposited into your checking account within six business days.

Online Merchant Center provides Merchant account management functions, virtual terminal transaction processing, and reports.

Transaction Security

Payment processing requires mechanisms using encryption to scramble data to protect customer payment information. Customer's private information, especially credit card numbers and bank account numbers, are securely encrypted as they are transmitted over the Internet.

The three most common data encryption methods are:

1) SSL (Secure Sockets Layer)
2) DES3 (Data Encryption Standard).
3) PGP (Pretty Good Privacy)

SSL Encryption

Secure Socket Layer (SSL) uses a public key to provide encryption between the host server and client browser and is the most secure encryption method. Many browsers including Microsoft Internet Explorer and Netscape Navigator support SSL encryption, but you will need to host your site on a secure server running SSL.

When Internet transmissions are made via SSL, the protocol for the Uniform Resource Locator (URL) address must include HTTPS, rather than HTTP to direct the transmission to the secure SSL port. The SSL public key encryption system works this way. The receiving computer discloses its public key and any other computer can use that public key to encrypt data that it sends to the receiving computer.

While the public key empowers anyone to encrypt a message, decryption is **not** possible on the basis of the public key. Only the receiving computer has the ability to decrypt, therefore, there is no need to distribute or store private keys, which may fall into the wrong hands.

DES3 Encryption

Unlike SSL, DES3 or Triple DES utilizes a private key and Internet transmissions are done using HTTP protocol, not HTTPS. The DES3 private key encryption system works as follows. Both the transmitting computer and the receiving computer must possess the same private key to encrypt and then decrypt the message. The reason this system is less secure than SSL is that the private key must be delivered from one computer to the other and there is the risk of it falling into the wrong hands during this process. The ideal method for private key transmission is by surface mail, but this involves a delay of several days.

Furthermore, once delivered, the private key could still fall into the wrong hands. If you elect to use DES3, sample C++ source code is available for the recommended implementation of the DES3 algorithm.

Pretty Good Privacy

For secure transmissions of batch transactions, you can submit files encrypted using PGP (Pretty Good Privacy), a registered trademark of Network Associates, Inc. PGP is a widely accepted and trusted method of data encryption that utilizes public key encryption technology.

Public key encryption uses two complimentary keys: one public and one private, to implement secure communications. The public key can be distributed freely to other users for encrypting files and cannot be used for decryption. The private key is kept by the user who is to receive the files, and must be used to decrypt files secured with the public key.

Online Transaction Processing

The Online Commerce Suite system allows you to use one of the following transaction processing methods, which include:

- Web Link (hosted shopping cart)
- Quicksale Method

    - Web form
    - Socket Connection
    - Secure Post
- Batch Payment Submission
    - via FTP upload
    - via Email
- Membership

Each method is a distinct way to connect your system and your customer's Web browser to the Online Commerce Suite system and the banking system. You must choose a method when you sign up. We recommend you read the entire guide and examine Table 1 before you decide which method to use.

Table 1 – Product gateway comparison

Method	OCS URL visible?	Transaction Security	Difficulty	Interface
Web Link	Yes	Provided	Moderate	HTML
QuickSale via Web form	Yes	You must provide a secure server	Moderate	HTML
QuickSale via Socket Connection	Hidden	You must provide a secure server	Difficult	HTTP or HTTPS
Quicksale Via Secure Post	Hidden	Inherent in method	Difficult	COM object
Batch Submission Via FTP	Hidden	Need to use PGP	Moderate	FTP
Membership	Yes	Provided	Easy	HTML

Web Link vs. QuickSale method comparison

Web Link	QuickSale
No secure server requirement for Merchant Web site	Secure server requirement
Detailed line item e-mail receipt generated	No line item email receipt (write your own if needed)
Shopping cart unnecessary	Shopping cart optional
Basic shipping and tax calculation by Online Commerce Suite	Write your own tax and shipping
	Integrate with UPS tracking #s
Customer Information page and Payment page can be customized	All pages are can be customized since Merchant writes the Web

Browser Title bar Header Footer	pages.
No recurring transactions	Recurring transactions allowed
Book and Ship	Book and ship
No custom merchant fields	Optional merchant defined fields supported

Membership

Membership is a turnkey solution for managing password protected subscription or Web membership sites. Consumers join to access your Member Only areas. Online Commerce Suite allows you to set up charges on a recurring basis. Technical Support remotely installs necessary software on you server and your programmers provide a few links to appropriate pages on your Web site. Please refer to the *Membership Guide* for more information, or contact Customer Service.

Batch Processing

Batch processing is oriented toward the non-interactive approach to data processing. Your system accumulates a number of transaction requests (a batch), submits them all for processing, and then gets a return batch of transaction results. This is not a good approach if your customer is waiting online, but it is an excellent way to process a large number of recurring billings at the end of the month. Please refer to the *Batch Processing Guide* for more information.

Web Link

Web Link provides an easy way for customers to purchase your products online. It guarantees secure transactions, even if your Web site is not hosted on a secure server.

With the Web Link option, you put product pages with “Buy It” buttons or links on your web site. When your customers click on the Buy It button, they are linked to the Online Commerce Suite hosted shopping cart page.

Online Commerce Suite initiates a secure dialog to obtain customer information and the preferred shipping method. The system displays the finalized order with the total billing amount. Your customer then provides a credit card number or bank account number and Online Commerce Suite processes the transaction.

How It Works

1. Your system displays the page with the Shopping Cart checkout or the Order Form to your customer. The page contains product information and choices to facilitate item selection. Data, in HTML hidden fields, with item price and shipping weight is also included in this page. With this information, Online Commerce Suite calculates the transaction invoice.

2. The customer finalizes the decision on items and quantities and clicks **Submit**. This action initiates the secure transfer of the completed page, including the number of items ordered, weight, and unit price to Online Commerce Suite.

3. Online Commerce Suite displays a page to your customer requesting preferred shipping method and shipping instructions.

4. After the information is provided, the page is securely transferred to Online Commerce Suite and used to calculate the shipping charge, sales tax and total amount due.

5. Online Commerce Suite displays a transaction request page to your customer showing the amount due calculated for this transaction and requests a credit card number or bank account number to debit for the total amount.

6. The customer provides the requested account information and clicks **Submit.** The transaction request data is securely transferred to Online Commerce Suite, which then processes the transaction request.

7. If the transaction is accepted, Online Commerce Suite generates a receipt for the transaction and e-mails it to you and to your customer. You can configure the system to skip the e-mail notification for you, your customer, or both.

8. The transaction result data is sent securely from Online Commerce Suite to your system. At minimum, the data contains the accept/decline flag. Your programmers can extract additional secure transaction data by setting the **usepost** flag.

9. Your system sends a transaction result page from your accepturl or declineurl to Online Commerce Suite. Your programmers can use CGI, ASP or other methods to dynamically create this page.

10. Online Commerce Suite displays the transaction result page in your customer's browser.

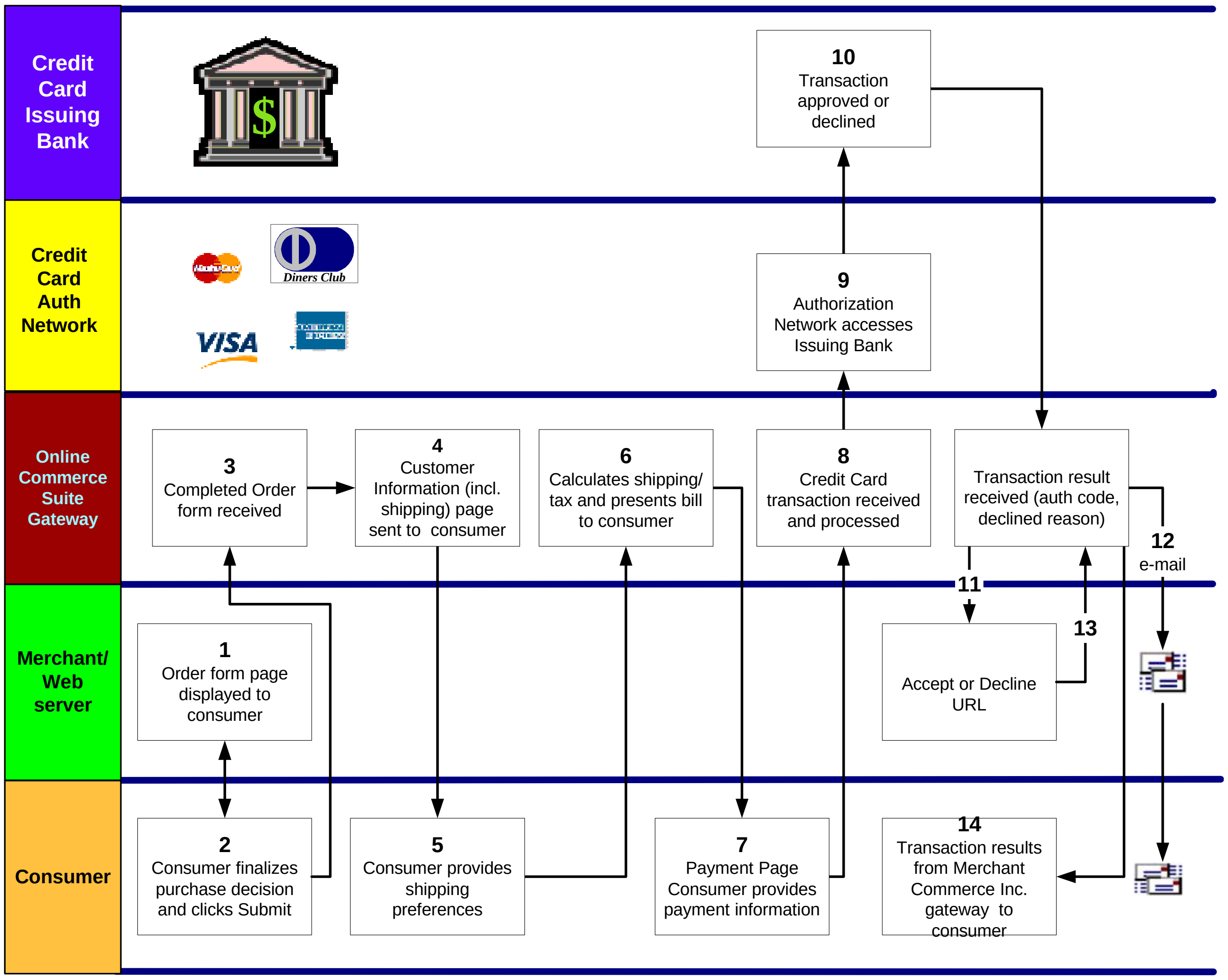

Credit Card Issuing Bank
Credit Card Auth Network
Online Commerce Suite Gateway
Merchant/ Web server
Consumer
Diners Club
VISA
10 Transaction approved or declined
9 Authorization Network accesses Issuing Bank
3 Completed Order form received
4 Customer Information (incl. shipping) page sent to consumer
6 Calculates shipping/ tax and presents bill to consumer
8 Credit Card transaction received and processed
Transaction result received (auth code, declined reason)
12 e-mail
11
13
1 Order form page displayed to consumer
Accept or Decline URL
2 Consumer finalizes purchase decision and clicks Submit
5 Consumer provides shipping preferences
7 Payment Page Consumer provides payment information
14 Transaction results from Merchant Commerce Inc. gateway to consumer

Figure 2 illustrates the Web Link transaction process.

Security

With Web Link processing, Online Commerce Suite secures all customer transactions using SSL. Your e-commerce Web server does not have to be a secure server to take advantage of the SSL security provided by Online Commerce Suite.

Shipping Charges

You can include a tag in your HTML to display shipping information and use the Online Merchant Center Shipping Manager menu to select a shipping calculation method (pounds, kilograms or dollar amount), set up shipping regions, select base shipping method, shipping rates, and premium shipping method. Refer to the Getting Started Guide for more information regarding configuring Shipping Charges

Tax Calculations

You can also include a tag if you want Online Commerce Suite to automatically calculate tax for an item and use the Online Merchant Center Tax Manager menu to set up tax rates by state. Refer to the Getting Started Guide for more information regarding configuring Tax Calculations.

Getting Information About Your Customers

You may get information about your customers and their transactions in three different ways:

1. E-mail automatically generated and sent at the time of the transaction. Credit card and bank account numbers are omitted from e-mail receipts.

2. Data sent at the time of the transaction, which your system can use to update your database. Transaction data is only sent if your programmers have set the **usepost** flag.

3. Logging into your Online Commerce Suite account some time after the transaction, and using the Transaction Menu to generate reports.

What Your Programmers Do

To create both the final order form and the transaction results pages, your programmers can use Web technologies, including CGI scripts, Active Server Pages, Cold Fusion applications, and so on. During the transaction sequence, several Web pages must appear on your customer's browser that are standard Web Link processing screens. You may have your logo and company name included in these standard screens; however, they must have the same basic layout as those used by other merchants using Web Link transaction processing. Your customers will see the Online Commerce Suite URL displayed on their browsers during execution of the secure transaction steps. Your programmers must be proficient in HTML.

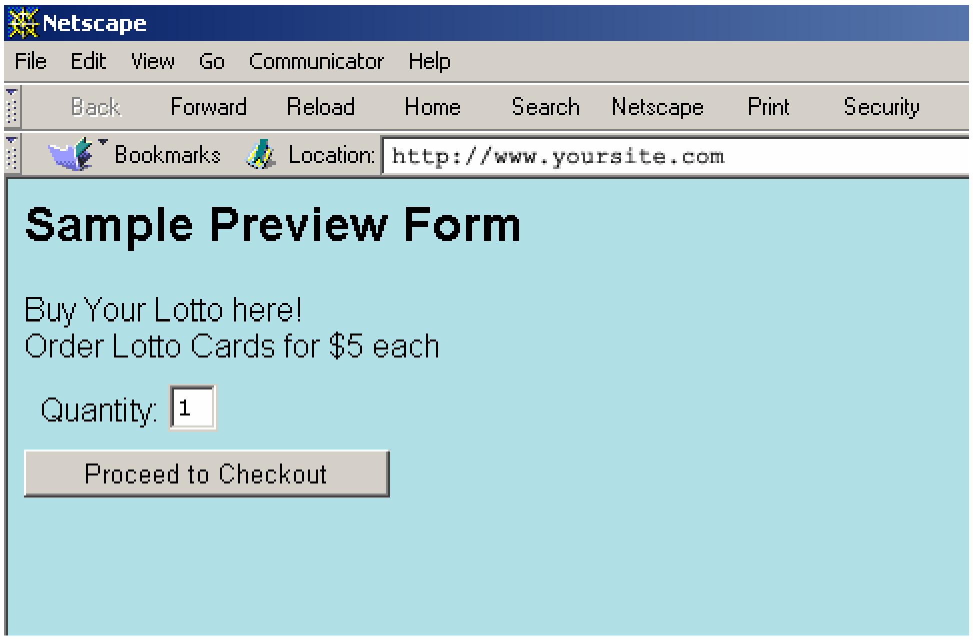
Integrating Your Web Site

The following HTML pages need to be created on your web site:

1. Order form page (required): contains the HTML input tags to post to the Online Commerce Suite payment gateway server.
2. accepturl page (optional): displays to the consumer when a transaction is accepted.

3. declineurl page (optional): displays to the consumer when transaction is declined by either a credit card authorization network or the ACH network.

Sample Preview Form


Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security
Bookmarks Location: http://www.yoursite.com
Sample Preview Form
Buy Your Lotto here!
Order Lotto Cards for $5 each
Quantity: 1
Proceed to Checkout

This sample illustrates one item for sale. To support multiple items, see "Customizing the HTML".

Your Order form HTML page must contain a form statement that posts to the Online Commerce Suite payment gateway. For example:

```
<form method=post action="https://trans.merchantpartners.com/cgi-
     bin/process.cgi">
```

The Order Form HTML contains required and optional input tags. The Merchant Commerce Inc. payment gateway will not respond to Merchant-defined input tags.

Required input tags are:

- acctid
- action
- itemid
- itemname
- itemdesc
- itemprice
- itemweight
- itemquant
- itemtax
- accepturl

- declineurl.

Sample HTML

Required tags are bold.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<body>
<font size="+2"><b><font face="Arial">Sample Preview
      Form</font></b></font>
<form method=post action="https://trans.merchantpartners.com/cgi-
      bin/process.cgi">
<input type=hidden name=action value="preview">
<input type=hidden name=acctid value=TEST0>
<input type=hidden name=accepturl
      value="http://trans.merchantpartners.com/~ats/accepted.html
      ">
<input type=hidden name=declineurl
      value="http://trans.merchantpartners.com/~ats/declined.html
      ">
<input type=hidden name=getshippinginfo value=1>
<input type=hidden name=header value="<h1>www.yoursite.com</h1>">
<input type=hidden name=title value="Title of your page">
<input type=hidden name=footer value="Footer of your page">
<input type=hidden name=body value="bgcolor=#b0e0e6">
<font face="Arial">Buy Your Lotto here!</font><br>
<font face="Arial">Order Lotto Cards for $5 each</font>
<table width=80% cellpadding=4>
<tr><td> <font face="Arial">Quantity:</font>
<input type=hidden name=itemid value="Lotto1">
<input type=hidden name=itemname value="Lotto Card">
<input type=hidden name=itemdesc value="State Lotto cards">
<input type=hidden name=itemprice value="5.00">
<input type=text name=itemquant value="1" maxlength=2 size=2>
<input type=hidden name=itemtax value="1">
<input type=hidden name=itemweight value="0.00">
</td></tr></table>
<input type=submit value="Proceed to Checkout">
</form>
</body>
</html>
```

Customizing the HTML

1. Replace TEST0 with your five-character Merchant ACCTID in the following tag: **`<input type=hidden name=acct value=TEST0>`**
2. For each item, copy and paste the following tags into the body of the page.

   ```
   <td>
   <center>
   &nbsp;<i>Quantity:</i>&nbsp;</font>
   <input type=hidden name=itemid value="Your Product 1">
   <input type=hidden name=itemname value="Your Product 1 Name">
   <input type=hidden name=itemdesc value="Your Product
         Description">
   <input type=hidden name=itemprice value="Your price">
   <input type=text name=itemquant value="1" maxlength=2 size=2>
   <input type=hidden name=itemtax value="1">
   <input type=hidden name=itemweight value="0.00">
   </td>
   ```

3. For each item, replace the itemid, itemname, itemdesc, and itemprice value = fields with your item information. For example, replace Your Product 1 with your itemid in the `<input type=hidden name=itemid value="` Your Product `1">` line. Insert descriptive text or other information.
4. To use the Online Commerce Suite Shipping Manager Customer and Shipping Information forms and fields, insert this tag:

   `<input type=hidden name=getshippinginfo value=1>`

   NOTE: You must set up your Shipping information through the Online Commerce Suite Shipping Manager user interface.
5. To get the Payment page information returned to your Merchant Web server when the transaction is complete, insert this tag:

   `<input type=hidden name=usepost value=1>`

   after

   ```
   <form method=post action=https://trans.merchantpartners.com/cgi-
         bin/process.cgi>
   ```

   NOTE: This tag enables the order form fields to post to the Online Commerce Suite database.
6. To calculate taxes, insert this tag:

   `<input type=hidden name=itemtax value="1">`

   NOTE: You must input the correct tax percentages for each state into the Online Commerce Suite Tax Manager user interface to calculate tax.

Required input fields for the Order form page

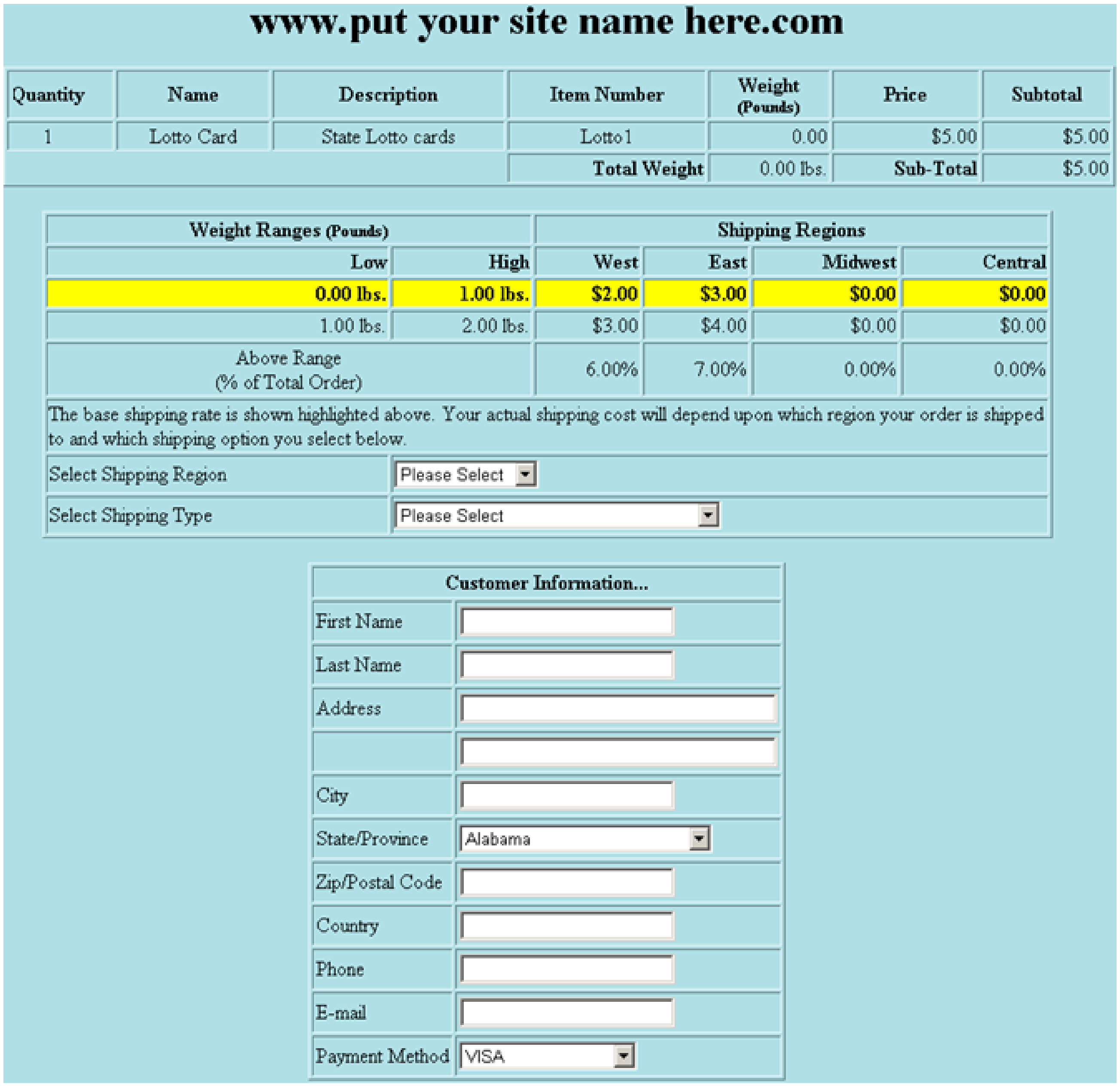
acctid	TEST0 (replace TEST0 with your Online Commerce Suite Merchant ID)
action	Preview
accepturl	Location of the page to display if transaction is accepted. Default is http://trans.merchantpartners.com/~ats/accepted.html
declineurl	Location of the page to display if transaction is declined. Default is http://trans.merchantpartners.com/~ats/declined.html
itemid	model # or SKU (Itemid must precede all other item fields; it is the delimiter between items in the stream.)
itemname	Name of item
itemdesc	Description of item
itemprice	Price in US dollars
itemweight	Weight in pounds
itemquant	Number of items
itemtax	1 = yes, sales tax applies to this item 0 = no, sales tax does not apply to this item

Optional input fields for the Order form page

subid	Merchant Sub ID. If unsure whether you have one, leave blank.
usepost	1 = a POST generated to the accept URL or decline URL 0 = no POST
body	BGCOLOR= or BACKGROUND=
title	HTML <title></title>
header	HTML code appearing at the top of the second preview page
footer	HTML code appearing at the bottom of the page
getshippinginfo	1 = yes, show shipping address fields 0 = no, do not show shipping address fields

Payment page

After the consumer clicks the Order form page Submit button, the following dynamically generated page appears. The title and header are what you coded in the Order form page. The Online Commerce Suite payment gateway URL is secure.



www.put your site name here.com

Quantity	Name	Description	Item Number	Weight (Pounds)	Price	Subtotal
1	Lotto Card	State Lotto cards	Lotto1	0.00	$5.00	$5.00
			Total Weight	0.00 lbs.	**Sub-Total**	$5.00

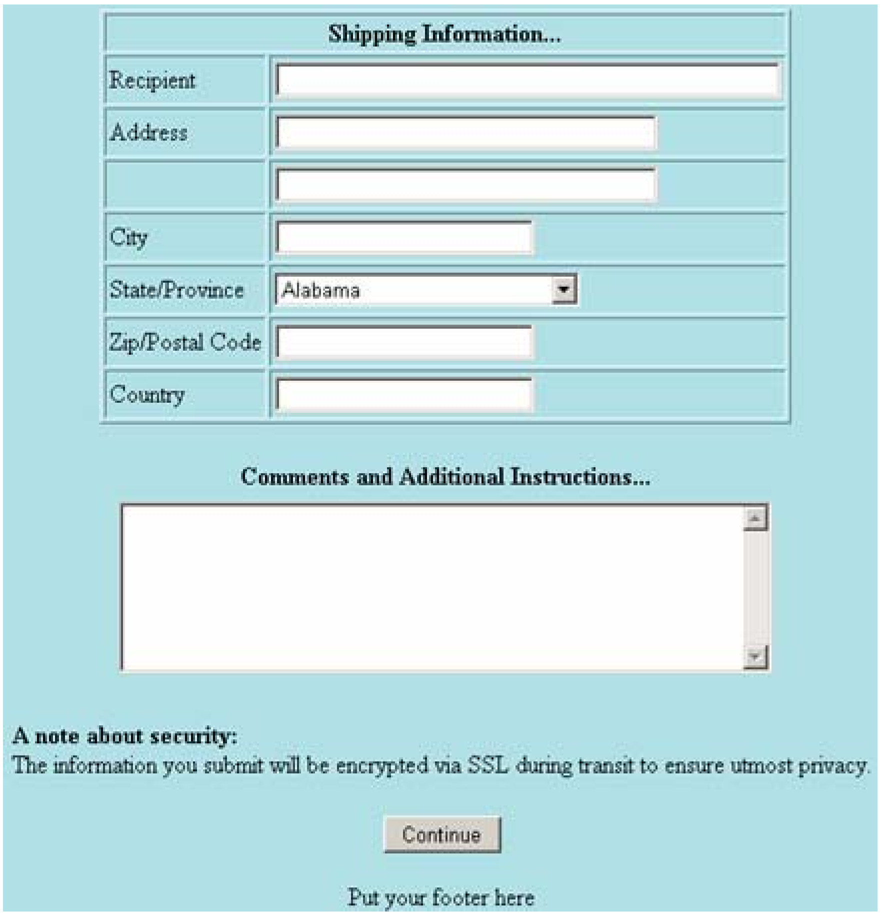
Weight Ranges (Pounds)		Shipping Regions			
Low	**High**	**West**	**East**	**Midwest**	**Central**
0.00 lbs.	**1.00 lbs.**	**$2.00**	**$3.00**	**$0.00**	**$0.00**
1.00 lbs.	2.00 lbs.	$3.00	$4.00	$0.00	$0.00
Above Range (% of Total Order)		6.00%	7.00%	0.00%	0.00%

The base shipping rate is shown highlighted above. Your actual shipping cost will depend upon which region your order is shipped to and which shipping option you select below.

Select Shipping Region	Please Select
Select Shipping Type	Please Select

Customer Information...	
First Name	
Last Name	
Address	
City	
State/Province	Alabama
Zip/Postal Code	
Country	
Phone	
E-mail	
Payment Method	VISA

The consumer scrolls down to enter information in the built in Shipping section. You configure shipping ranges, prices and regions in the Online Merchant Center (OMC) Administrative interface.



Shipping Information...

Recipient	
Address	
City	
State/Province	Alabama
Zip/Postal Code	
Country	

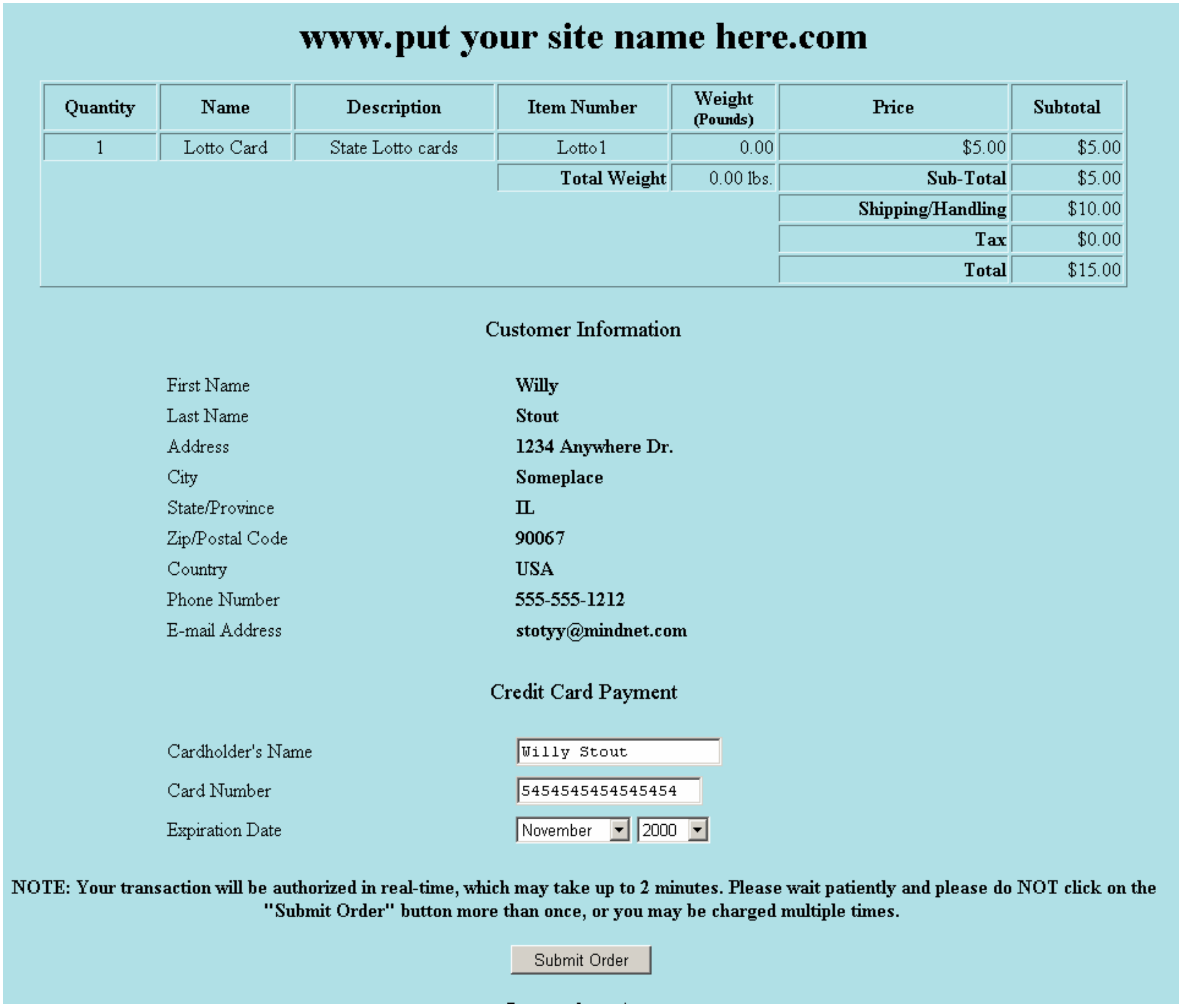
Comments and Additional Instructions...

A note about security:
The information you submit will be encrypted via SSL during transit to ensure utmost privacy.

Put your footer here

The consumer enters the required information and then clicks the Continue button.

The Payment page appears and the consumer enters credit card information and clicks the Submit Order button. The consumer's browser is still pointing to the secure Online Commerce Suite payment gateway URL. The payment gateway contacts the appropriate authorization network and returns an approved or declined response to the consumer.



www.put your site name here.com

Quantity	Name	Description	Item Number	Weight (Pounds)	Price	Subtotal
1	Lotto Card	State Lotto cards	Lotto1	0.00	$5.00	$5.00
			Total Weight	0.00 lbs.	**Sub-Total**	$5.00
					Shipping/Handling	$10.00
					Tax	$0.00
					Total	$15.00

Customer Information

First Name	**Willy**
Last Name	**Stout**
Address	**1234 Anywhere Dr.**
City	**Someplace**
State/Province	**IL**
Zip/Postal Code	**90067**
Country	**USA**
Phone Number	**555-555-1212**
E-mail Address	**stotyy@mindnet.com**

Credit Card Payment

Cardholder's Name	Willy Stout
Card Number	5454545454545454
Expiration Date	November 2000

NOTE: Your transaction will be authorized in real-time, which may take up to 2 minutes. Please wait patiently and please do NOT click on the "Submit Order" button more than once, or you may be charged multiple times.

Return format

Other text type inputs pass consumer data entered into the Online Commerce Suite Customer Information and Payment pages to your CGI script, if you coded a hidden input tag for **usepost**. For example:

```
<input type=hidden name=usepost value="1">
```

If you do not code a **usepost** value=1, the only information your Merchant Web Server receives from the Merchant Commerce Inc. payment gateway is either the accepturl or the declineurl, requesting the appropriate page sent via an HTTP GET. You do **not** get consumer name, ordered items or any other information.

An accepted transaction without usepost=1 returns the following from Online Commerce Suite:

```
Server Software: Stronghold/2.4.1 Apache/1.3.3 C2NetEU/2409
(Unix)
Server Name: trans.merchantpartners.com
Gateway Interface: CGI/1.1
Server Protocol: HTTP/1.0
Server Port: 80
Request Method: GET
HTTP Accept:
Path Info:
Path Translated:
Script Name: /cgi-bin/test.cgi
Query String: Status=Accepted&AuthNo=TEST
Remote Host:
Remote Addr: 216.34.199.102
Remote User:
Auth Type:
Content Type:
Content Length:
```

It's a GET so the returns are in the query string.

If declined because of an invalid credit card, you get:

```
Declined=Invalid Credit Card Number
historyid=7424103
orderid=
```

If you included a usepost=1, the following can be returned from the Online Commerce Suite gateway if you write a script to capture the values:

```
<plaintext>Server Software: Stronghold/2.4.1 Apache/1.3.3
C2NetEU/2409 (Unix)
Server Name: trans.merchantpartners.com
Gateway Interface: CGI/1.1
Server Protocol: HTTP/1.0
Server Port: 80
Request Method: POST
HTTP Accept:
Path Info:
Path Translated:
Script Name: /cgi-bin/test.cgi
Query String:
Remote Host:
Remote Addr: 216.34.199.51
Remote User:
Auth Type:
Content Type: application/x-www-form-urlencoded
Content Length: 1280
Version = 1
Status = Accepted
AuthNo = TEST
OrderID = 3661207
TaxAmount = 31.76
ShippingAmount = 22.00
TotalAmount = 438.76
PostedVars = BEGIN
usegc = 1
debug =
acctid = TEST0
pspid = TEST0
subid =
currency =
accepturl = http://trans.merchantpartners.com/cgi-
bin/test.cgi
declineurl = http://trans.merchantpartners.com/cgi-
bin/showresult.cgi
body = bgcolor=#faf0e6
title = Title of your page
header = <h1>www.yoursite.com here</h1>
footer = Footer of your page
domainname =
securedomainname =
alias =
usepost = 1
itemid = Lotto1
itemname = Lotto Card
itemdesc = State Lotto cards
itemprice = 5.00
itemsig = 0
itemweight = 0
itemquant = 77
itemtype =
itemtax = 1
itemflags =
itemcomments =
itemattributes =
itemGSTHST = 0
itemHSTreduced = 0
itemQST = 0
itemPST = 0
fname = Willy
lname = Stout
phone = 555-555-1212
addr1 = 1234 Anywhere Dr
addr2 =
city = Someplace
state = IL
zip = 90067
country = USA
phone = 555-555-1212
email = stotyy@mindnet.com
comments =
receipttype =
paymethod = mc
shipname =
shipaddr1 =
shipaddr2 =
shipcity =
shipstate =
```

Sample e-mail receipt

```
============================================================
Thank you for ordering from Test Merchant

Your order information appears below.  If you need to get in
touch with us about your order, send an e-mail message to
yourCustomerServiceEmail (or just reply to this message)
Your order reads as follows:

E-mail address:         mconsumer@innuity.com
Billed to:        Willy Stout
                  1234 Anwy

                  Los Angeles
                  CA
                  90025
                  USA

Ship via:         FedEx 2nd Day Air
Ship to:          Mary Consumer
                  12301 Wilshire Blvd.

                  Los Angeles
                  CA
                  90025
                  USA

Phone:                  333-333-3333
Paid By:          MasterCard

Authorization:          TEST
Date and Time placed:   10/12/00 12:00:28 (Pacific)
Order Reference Number: 3661207
------------------------------------------------------------
Quant    Price     Sub-Tot  ID          Name
   77  $5.00     $385.00    Lotto1      Lotto Card

Sub-Total:        $385.00
Shipping:         $22.00
Tax:              $31.76
                  ----------
TOTAL AMOUNT:           $438.76

Thank you for shopping at Test Merchant
============================================================
Please print out this authorization and keep it with your bank records.
```

Quicksale Method with a Web form

Quicksale Method with a Web form gives you almost complete control over what your customers see on their browsers. Your programmers can customize your transaction pages with CGI scripts, Active Server Pages, Cold Fusion applications, and so on. This method requires you to be on a secure server (with SSL installed).

Your system gathers customer information and calculates shipping and tax in a manner controlled by your programmers. Online Commerce Suite does the final transaction—obtaining the credit card number or bank account number from your customer and processing the transaction in a one-step secure dialog with your customer.

Your programmers must be proficient in HTML to integrate your e-commerce Web site.

How It Works

1. Your system displays the transaction request page, which asks your customer to supply either a credit card number or bank account number, to your customer's browser.

2. After your customer submits the account information, the data is securely transferred to Online Commerce Suite, which processes the transaction request.

3. If the transaction is accepted, then Online Commerce Suite generates a receipt for the transaction and e-mails it to you and to your customer. You can configure the system to **not** e-mail either you or your customer, or both.

4. Online Commerce Suite securely sends the transaction result data to your system. At minimum, the data contains the accept/decline flag, the authorization code if accepted and the reason if declined.

5. Your programmers can take advantage of additional secure transaction data that Online Commerce Suite can provide in this step, by setting the **usepost** flag.

6. Your system sends a transaction result page from either your accepturl or your declineurl to Online Commerce Suite. Your programmers can use CGI or other methods to dynamically create this page.

7. Online Commerce Suite passes the transaction result page to your customer's browser.

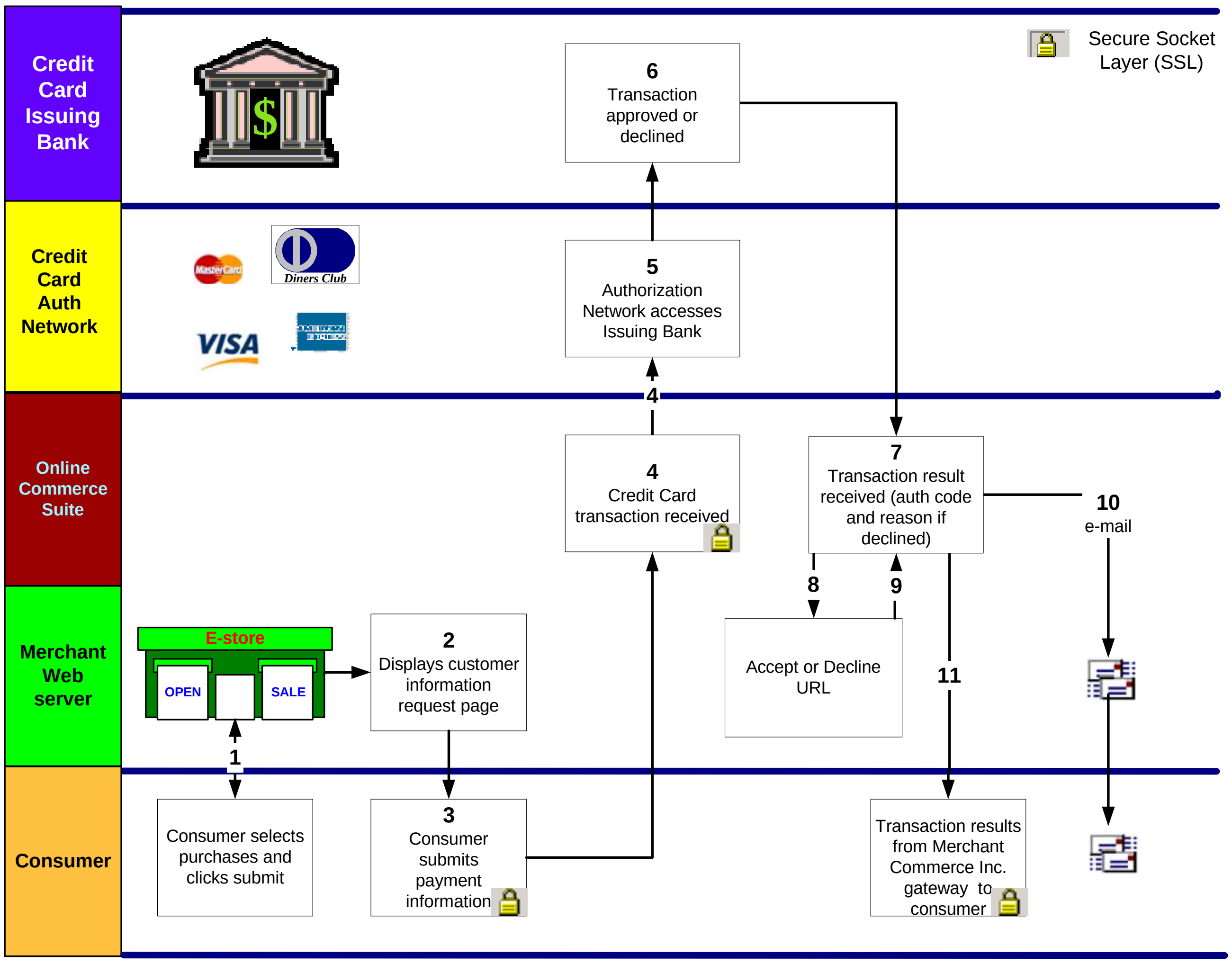

Credit Card Issuing Bank
Credit Card Auth Network
Online Commerce Suite
Merchant Web server
Consumer
Secure Socket Layer (SSL)
6 Transaction approved or declined
5 Authorization Network accesses Issuing Bank
4 Credit Card transaction received
7 Transaction result received (auth code and reason if declined)
10 e-mail
E-store
OPEN
SALE
2 Displays customer information request page
Accept or Decline URL
8
9
11
1
Consumer selects purchases and clicks submit
3 Consumer submits payment information
Transaction results from Merchant Commerce Inc. gateway to consumer

Figure 3 illustrates the Quicksale Method with Web form payment process.

Security

The most sensitive transaction data like credit card number or bank account number is transmitted directly from the customer's browser to the Online Commerce Suite server via SSL. Customer data received by your system prior to your system asking the customer for a credit card number or bank account number (Step 1), which might be used to create the transaction request page, is also transmitted securely because you have installed SSL on your server.

Getting Information About Your Customers

There are four ways to receive information about your customer's order:

1. Data your system obtains from your customer prior to your system asking the customer for a credit card number or bank account number (Step 1). Remember, this data can only be securely obtained if you have a secure server.

2. E-mail automatically generated and sent to you at the time of the transaction. Credit card and bank account numbers are omitted from email receipts.
3. Data included at the time of the transaction, which your system can use to update your database. This data is only sent if your programmers set the **usepost** flag. Sensitive information such as consumer credit card numbers and account numbers are not sent back.
4. Logging into your Online Commerce Suite account some time after the transaction, and using the Transaction menu to generate reports.

What Your Programmers Do

When you use the Quicksale Method with a Web form, your programmers create all the pages leading up to and including the final transaction request page, updating your local database in the process with customer information. This transaction request page is transferred from your system to your customer's browser, where your customer supplies the credit card or bank account number. The transaction request is then submitted from your custom designed transaction request page in your customer's browser directly to Online Commerce Suite. Results are reported to your system, and your programmers provide a page for the final transaction results to display.

To create both the transaction request page and the transaction results pages, your programmers can use Web technologies, including CGI scripts, Active Server Pages, Cold Fusion applications, and so on. Your programmers must be proficient in HTML if you select this transaction processing method.

Data Elements

Tables 2–5 describe required and optional data elements that can be **sent** to the Online Merchant Center payment gateway to complete a transaction with Single payment submission with a Web form processing.

Required input fields for credit card processing

action	The ns_quicksale_cc action instructs Online Commerce Suite to process a credit card order.
acctid	Use TEST0 for testing if you do not have an Account ID. Change to your Account ID for live transaction processing.
accepturl	The consumer is transferred to this URL after the transaction is processed and approved. You may point to a CGI script. Example: http://trans.merchantpartners.com/~ats/accepted.html The script specified in Accepturl and Declineurl will always be called by a connection originating at trans.merchantpartners.com . Do **not** set the return URL to https:
declineurl	The consumer is transferred to this URL after the transaction is processed and declined. You may point to a CGI script. Example: http://trans.merchantpartners.com/~ats/declined.html Do **not** set the return URL to https:
amount	Transaction dollar amount in US dollars in the form of 0.00.
ccname	Consumer name as it appears on the credit card.

ccnum	Consumer's credit card number. Do not include spaces.
expmon	Expiration month (12) of the consumer credit card.
expyear	Expiration year of the consumer's credit card in yyyy format

Optional input fields for credit card processing

subid	Merchant Sub ID. If unsure whether you have one, leave blank.
Authonly	A value of 1 pre-authorizes the credit card. A pre-authorization will "reserve" the amount specified in the amount field, it will not actually bill the consumer's credit card. This process is used for Book and Ship sales transactions, where a Merchant gets an order and at a later date, completes the transfer of funds.
postonly	The Refcode returned from a previous Authonly=1 (pre-authorization transaction). This is the post-authorization step of a Book and Ship sales transaction. See Authonly above.
usepost	To return order form information to a database, specify usepost=1.
ci_companyname	Your company name
ci_billaddr1	Consumer billing address
ci_billaddr2	Second line of the consumer billing address
ci_billcity	Consumer city
ci_billstate	Consumer state or province
ci_billzip	Consumer Zip code or Postal code
ci_billcountry	Consumer country
ci_shipaddr1	Consumer shipping address
ci_shipaddr2	Consumer second line of shipping address
ci_shipcity	Consumer shipping city
ci_shipstate	Consumer shipping state or province
ci_shipzip	Consumer shipping Zip Code or Postal Code
ci_shipcountry	Consumer shipping country
ci_phone	Consumer phone number
ci_email	Consumer email address
ci_memo	Miscellaneous information field
ci_dlnum	Consumer driver's license number
ci_ssnum	Consumer Social Security Number
emailto	E-mail address to send the consumer e-mail receipt. Default is ci_email address.
emailfrom	Return address on consumer's e-mail receipt. Default is *null@atsbank.com*.
emailsubject	Subject line on consumer's receipt email. Default message is *Payment Receipt #xzy*.
emailtext	Consumer's e-mail receipt body text. Default is a generic receipt message.
ci_ipaddress	Consumer's IpAddress
merchantordernumber	Customer's unique alpha-numeric number
cvv2	Credit Card cvv2/cvc2 code

Required input fields for ACH processing

action	ns_quicksale_check instructs Online Commerce Suite to process a check order.
acctid	Use TEST0 for testing if you do not have an Account ID. Change to your Account ID for live transaction processing.
accepturl	The consumer is transferred to this URL after the transaction is approved Point to a CGI script or html page like http://trans.merchantpartners.com/~ats/accepted.html
declineurl	The consumer is transferred to this URL after the transaction is declined. Point to a CGI script or html page like http://trans.merchantpartners.com/~ats/declined.html
amount	Transaction dollar amount in US dollars in the form of 1.00
ckname	Consumer name as it appears on the checking account.
ckaba	Nine-digit numeric value without spaces for checking account routing or ABA number.
ckacct	Variable length numeric value without spaces for checking account number.

Optional input fields for ACH processing

subid	Merchant Sub ID. If unsure if you have one, leave blank.
ci_companyname	Your company name
ci_billaddr1	Consumer billing address
ci_billaddr2	Second line of the consumer billing address
ci_billcity	Consumer city
ci_billstate	Consumer state or province
ci_billzip	Consumer zip code or postal code
ci_billcountry	Consumer country
ci_shipaddr1	Consumer shipping address
ci_shipaddr2	Second line of consumer shipping address
ci_shipcity	Consumer shipping city
ci_shipstate	Consumer shipping state or province
ci_shipzip	Consumer shipping zip or postal code
ci_shipcountry	Consumer shipping country
ci_phone	Consumer phone number
ci_email	Consumer's e-mail address
ci_memo	Miscellaneous information field
ci_dlnum	Consumer driver license number
ci_ssnum	Consumer Social Security number
emailto	E-mail address where to send consumer's e-mail receipt. Default is ci_email
emailfrom	Return address on consumer's e-mail receipt. Default is null@atsbank.com.

emailsubject	Subject line on the consumer's receipt e-mail. Default is Payment Receipt #xzy
emailtext	Text for consumer's e-mail receipt. Default is a generic receipt message.
ci_ipaddress	Consumer's IpAddress
merchantordernumber	Customer's unique alpha-numeric number

Optional recurring billing tags

recur_create	Creates a recurring billing record for a consumer. Set to recur_create=1 to create a recurring billing record.
recur_billingcycle	0=No Recurring Billing Cycle 1=Weekly Recurring Cycle 2=Monthly Recurring Cycle 3=Quarterly Recurring Cycle 4= Semi-Annual Recurring Cycle 5= Annual Recurring Cycle 6=Bi-Weekly Recurring Cycle 7=Bi-Annual Recurring Cycle 8=Quad Weekly (28 day) Recurring Cycle
recur_billingmax	Maximum number of times a consumer's account is debited through recurring billing. For example, setting recur_billingmax =6 bills the consumer 6 times. -1 = Unlimited number of times 0 = No Recurring Billing
recur_start	Number of days after an initial payment where the consumer is debited on a recurring cycle.
recur_amount	Amount the consumer is to be re-debited on the recurring cycle. Do **not** use a dollar sign.

Required input fields for Voids

action	ns_void instructs Online Commerce Suite to issue a void.
acctid	Use TEST0 for testing if you do not have an Account ID. Change to your Account ID for live transaction processing.
subid	Required only If transaction was submitted under a subid.
amount	Transaction dollar amount in US dollars in the form of 1.00
orderkeyid	Orderkeyid of the original transaction
historykeyid	Historykeyid of the original transaction

Required input fields for Credits

action	ns_credit instructs Online Commerce Suite to issue a credit.
acctid	Use TEST0 for testing if you do not have an Account ID. Change to your Account ID for live transaction processing.
subid	Required only If transaction was submitted under a subid.
amount	Transaction dollar amount in US dollars in the form of 1.00
orderkeyid	Orderkeyid of the original transaction
historykeyid	Historykeyid of the original transaction

Web Form Submission Data Formats

To send the transaction to the Online Commerce Suite payment gateway server via HTTP or HTTPS, post the fields to the following URL:

Using HTTP with SSL (Secure Socket Layer):

https://trans.merchantpartners.com/cgi-bin/process.cgi

Using HTTP without SSL (no encryption):

http://trans.merchantpartners.com/cgi-bin/process.cgi

The **accepturl** and **declineurl** are required when using the HTML method**.** They can reference either static HTML pages or CGI scripts

Request Format

The following two examples show how to use HTML to transmit a transaction using HTTP and

HTTPS. The Online Commerce Suite distribution package contains several additional example

HTML files, attached as zip files.

Sample HTML to Submit a Check Transaction Using HTTP(S)

```
<form method=post action=https://trans.merchantpartners.com/cgi-
bin/process.cgi>
<input type=hidden name=action value="ns_quicksale_check">
<input type=hidden name=acctid value="TEST0">
<input type=hidden name=amount value="1.00">
<input type=hidden name=ckname value="Joe Customer">
<input type=hidden name=ckaba value="123456789">
<input type=hidden name=ckacct value="123456789012345">
<input type=hidden name=accepturl
value="http://trans.merchantpartners.com/~ats/accepted.html">
<input type=hidden name=declineurl
value="http://trans.merchantpartners.com/~ats/declined.html">
<input type=submit>
</form>
```

Sample HTML to Submit a Canadian Check Transaction Using HTTP(S)

```
<form method=post action=https://trans.merchantpartners.com/cgi-
bin/process.cgi>
<input type=hidden name=action value="ns_quicksale_check">
<input type=hidden name=acctid value="TEST0">
<input type=hidden name=amount value="1.00">
<input type=hidden name=ckname value="Joe Customer">
```

```
<input type=hidden name=ckaba value=" 12345-123">
<input type=hidden name=ckacct value="123456789012345">
<input type=hidden name=accepturl
value="http://trans.merchantpartners.com/~ats/accepted.html">
<input type=hidden name=declineurl
value="http://trans.merchantpartners.com/~ats/declined.html">
<input type=submit>
</form>
```

Sample HTML to Submit a Credit Card Transaction:

```
<form method=post action=https://trans.merchantpartners.com/cgi-
bin/process.cgi>
<input type=hidden name=action value="ns_quicksale_cc">
<input type=hidden name=acctid value="TEST0">
<input type=hidden name=amount value="1.00">
<input type=hidden name=ccname value="Joe Customer">
<input type=hidden name=ccnum value="5454545454545454">
<input type=hidden name=expmon value="01">
<input type=hidden name=expyear value="2004">
<input type=hidden name=accepturl
value="http://trans.merchantpartners.com/~ats/accepted.html">
<input type=hidden name=declineurl
value="http://trans.merchantpartners.com/~ats/declined.html">
<input type=submit>
</form>
```

Response Format

When a transaction is processed, Online Commerce Suite retrieves the accepted or declined URL directly from the Online Commerce Suite Server, then relays it to the customer's browser. The customer's browser is connected to the Online Commerce Suite secure system during the entire transaction process event. The mechanics of this process are:

1. After a consumer's order is processed, Online Commerce Suite initiates a TCP/IP connection between the Online Commerce Suite server and the Merchant system.
2. An HTTP GET command retrieves the accepted or declined URL.
3. If the Merchant's e-commerce application includes a query string as part of the accepted or declined URL, Online Commerce Suite appends its own responses to the end of the query string. This allows the Merchant to pass as much additional information in the query string as desired.
4. HTTP GET command output is sent to the customer's browser. If the accepted or declined URL references a static HTML page, it is displayed. If

the accepted or declined URL references a CGI script, the script output is displayed.

Online Commerce Suite uses the following formats to return the query string response to the consumer's browser:

Transaction is Accepted

If the transaction is accepted, the following format is used:

http://www.myserver.com/cgibin/mycgi?Status=Accepted&AuthNo=AUTHCODE

The authorization code is the alphanumeric value returned by the credit card processing network, representing a receipt of each individual transaction.

Transaction is Accepted and Consumer Info is Returned

If the **usepost** flag is enabled, the POST sent to the **accepturl** is printed with the name / value pairs below::

```
Version=1
Status=Accepted
AuthNo=######
PostedVars=BEGIN
action=ns_quicksale_check
acctid=TEST0
amount=1.00
ckname=Joe+Customer
usepost=1
accepturl=http%3A//trans.merchantpartners.com/cgi-bin/ret.cgi
declineurl=http%3A//trans.merchantpartners.com/cgi-bin/ret.cgi
PostedVars=END
```

The POST body is encoded as above. Notice that the entire original POST information sent to SPX is listed between `PostedVars=BEGIN` and `PostedVars=END`. All consumer financial information (account numbers) is **not** sent.

Transaction is Declined

If the transaction is declined, the following format is used:

http://www.myserver.com/cgibin/mycgi?Status=Declined&Reason=Reason

The reasons for a declined transaction are numerous and depend on the credit card or check processing network through which your account is routed. Online Commerce Suite does generate a number of internal declined reasons.

They are:

"Invalid first name"

"Invalid last name"

"Invalid checking account number"

"Invalid checking account number, enter only digits"

"Invalid ABA/Routing Number"

"Invalid ABA/Routing Number, must be nine characters in length

"Negative Entry"

Transaction is Declined and Consumer Info is Returned

If the usepost flag is enabled, the POST sent to the declineurl is transmitted in the following format:

Version=1

Status=Decline

Reason= (decline reason goes here)

PostedVars=BEGIN

action=ns_quicksale_check

acctid=TEST0

amount=1.00

ckname=Joe+Customer

usepost=1

accepturl=http%3A//trans.merchantpartners.com/cgi-bin/ret.cgi

declineurl=http%3A//trans.merchantpartners.com/cgi-bin/ret.cgi

PostedVars=END

The POST body is encoded as above. Notice that all of the original POST information sent to SPX is listed between PostedVars=BEGIN and PostedVars=END. All consumer financial information (account numbers) is **not** sent.

Quicksale Method with a Socket Connection

This method gives you 100% control over what your customers see. All Online Commerce Suite server interactions are through your system. The Online Commerce Suite URL never appears on your customer's browser. This method requires advanced programming skills since your programmers create *all* the transaction pages. Your system must be a secure server. Communication between your system and the Online Commerce Suite system is done in single transactions via a direct socket, using HTTP. A socket connection supports interactivity with your customer. Your system submits a single transaction request on behalf of your customer, and gets an immediate transaction result back.

How It Works

1. Your system transfers the transaction request page, which asks your customer to supply either a credit card number or bank account number.
2. After your customer fills in the account information and clicks **Submit**, the transaction request data is transferred back to your system.
3. Your system sends the transaction request data to the Online Commerce Suite server via the direct socket connection. The Online Commerce Suite server then processes the transaction request.
4. The Online Commerce Suite server sends the transaction result data back to your system via the socket connection.
5. Your system creates a transaction result page and transfers it to your customer's browser.

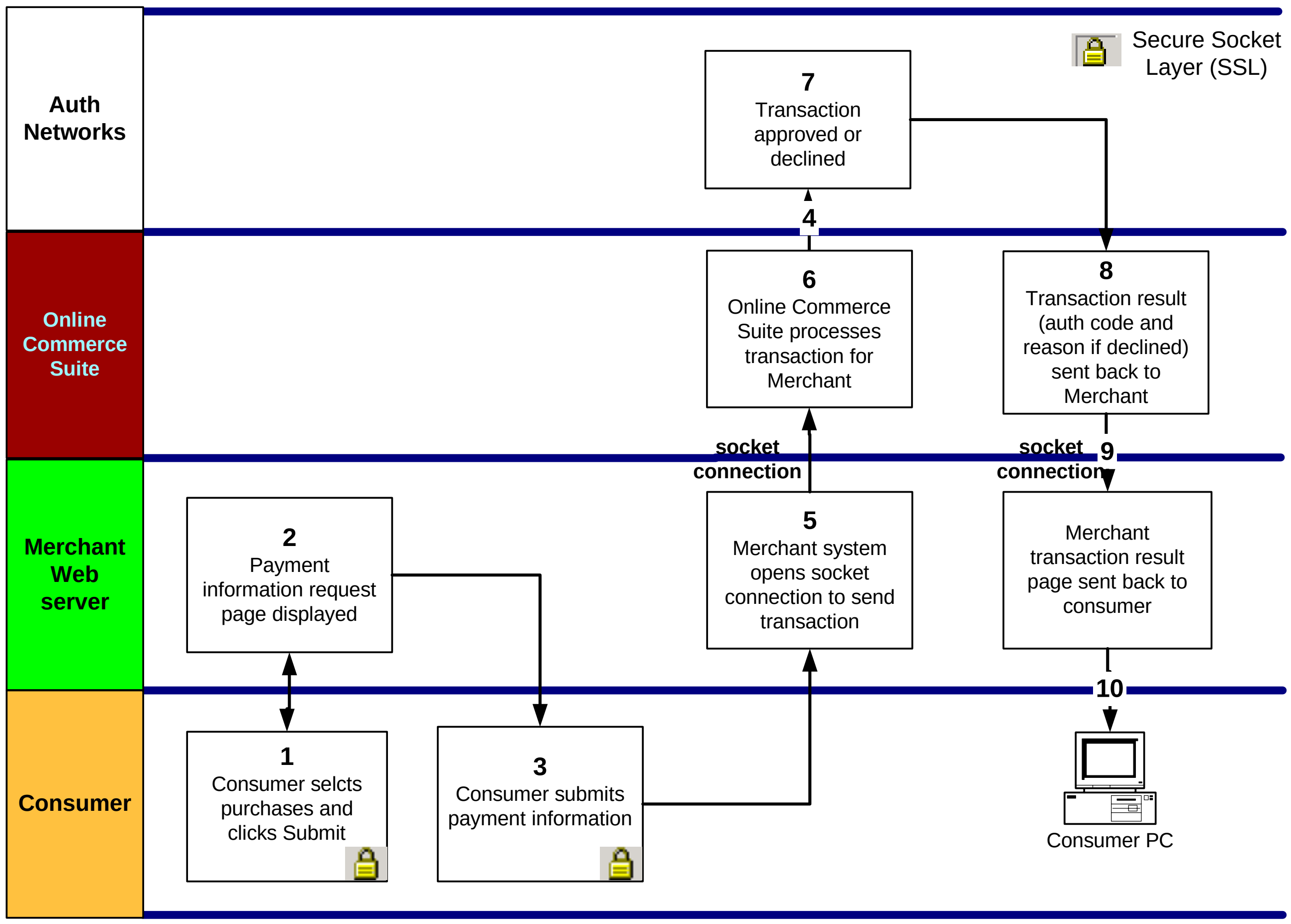

Auth Networks
Online Commerce Suite
Merchant Web server
Consumer
Secure Socket Layer (SSL)
7 Transaction approved or declined
4
6 Online Commerce Suite processes transaction for Merchant
8 Transaction result (auth code and reason if declined) sent back to Merchant
socket connection
9
socket connection
2 Payment information request page displayed
5 Merchant system opens socket connection to send transaction
Merchant transaction result page sent back to consumer
10
1 Consumer selcts purchases and clicks Submit
3 Consumer submits payment information
Consumer PC

Figure 4 illustrates Quicksale Method flow using a socket connection.

Security

All aspects of transaction security are dependent on the Online Commerce Suite Server. If you choose the socket connection method, your server must be secure with SSL to protect your transmissions. You must then decide upon a method to secure transaction request data over the socket connection to the Online Commerce Suite server.

To secure data transmitted between your system and your customer's browser, your system must be a secure server using SSL. However, the SSL provided by your system will **not** secure the transmissions between your system and the Online Commerce Suite server.

Even though these transmissions **are** done over the Internet, they are **not** done over the World Wide Web and do **not** pass through your secure system. They use HTTP independent of your system.

Getting Information About Your Customers

You can receive information about your customer's order in three ways:

1. Data your system obtains from your customer prior to asking for a credit card or bank account number (Step 1). This data can be securely obtained only if you have a secure server.
2. Data included when Online Commerce Suite sends the transaction request data back to your system (Step 4), which your system can use to update your database.
3. Logging into your Online Commerce Suite account some time after the transaction, and using the Transaction menu to generate reports.

What Your Programmers Do

Your programmers have more responsibility than with the other transaction processing methods. They create the entire interface to your customer's browsers and must follow specific protocols in communicating with the Online Commerce Suite server. Your programmers must have advanced skills and proficient in HTTP.

Socket Connection Data Formats

To prepare to transmit a transaction via the direct socket post method, you must first initiate a socket connection to the Online Commerce Suite Payment Gateway URL. Use **Port 80** for http. Use **Port 443** for https (Secure Socket Layer).

Request Format

When you establish a socket connection to the Online Commerce Suite server, you are ready to transmit the transaction information.

The next section contains two sample streams, which illustrate how to transmit an electronic check and a credit card transaction using a direct socket post. To test:

1. Open a TCP/IP session to the host listed above using a Telnet client.
2. When connected, cut and paste the text into the Telnet window replacing checking account or credit card numbers with valid account information.
3. Set the Content-length field to the exact number of characters in the HTTP request body (the line that starts with "action="). For example, if the line contains exactly 102 alphanumeric characters, the entry appears as Content-length: 102.
4. Wait for a response from the server.
5. Terminate all lines by a carriage return/line feed (**CR/LF)**. Be careful not to split credit card or checking account information with line breaks or the server generates an error response.

Sample HTTP Request to Submit an Electronic Check (ACH) Transaction

```
POST /cgi-bin/process.cgi HTTP/1.0
```

```
Content-type: application/x-www-form-urlencoded
Content-length: 101
action=ns_quicksale_check&acctid=TEST0&amount=1.00&ckn
ame=John%20Doe&ckaba=99999999&ckacct=9999999999&
```

Sample HTTP Request to submit a credit card transaction

```
POST /cgi-bin/process.cgi HTTP/1.0
Content-type: application/x-www-form-urlencoded
Content-length: 111
action=ns_quicksale_cc&acctid=TEST0&amount=1.00&ccname
=John%20Doe&ccnum=5454545454545454&expmon=01&expyear=
2001&
```

To use the direct socket post method to transmit transactions, you must be fully proficient with the HTTP protocol. If you are uncertain as to how to correctly use the direct socket post method, use the HTML (Web Form) Submission Method.

Response Format

After a transaction is submitted via the direct socket submission method, the Online Commerce Suite server responds with the results of the transaction within a few seconds. It is up to your program to interpret the results of the transaction and route the consumer accordingly. Here are sample responses for accepted and declined transactions.

Transaction is Accepted

```
HTTP/1.1 200 OK
Date: Thu, 16 Jul 1998 01:23:15 GMT
Server: Stronghold/2.3 Apache/1.2.6 C2NetUS/2007
Connection: close
Content-Type: text/html
<html><body><plaintext>Accepted=0000000875
```

Transaction is Declined

```
HTTP/1.1 200 OK
Date: Thu, 09 Jul 1998 20:38:16 GMT
Server: Stronghold/2.3 Apache/1.2.6 C2NetUS/2007
Connection: close
Content-Type: text/html
<html><body><plaintext>Declined=Negative Entry
```

Transaction Request is in Error:

```
HTTP/1.1 200 OK
Date: Thu, 09 Jul 1998 20:38:16 GMT
Server: Stronghold/2.3 Apache/1.2.6 C2NetUS/2007
Connection: close
Content-Type: text/html
```

```
<html><body><plaintext>Error=Invalid Merchant
```

Quicksale Method with SecurePost Object

Overview

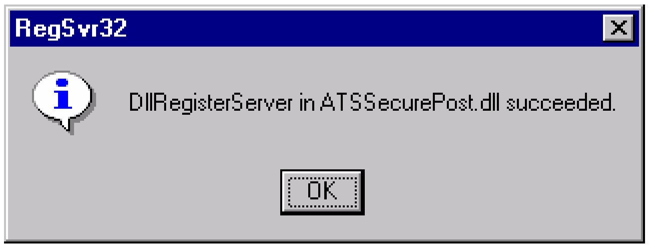
The Online Commerce Suite SecurePost object provides an easy way to integrate credit card or electronic check (ACH) payment processing into your custom applications or Web sites. The SecurePost object is implemented as a COM object capable of running on Windows 95, Windows 98, or Windows NT 4.0 or later. It can be used with any programming language capable of calling a COM object, such as Visual C++, Visual Basic, Active Server Pages (ASP), Microsoft Office applications, and others. The transaction data is transferred over a Secure HTTP (HTTPS) connection using the installed Windows Sockets (WINSOCK) stack and Internet Extensions (WININET). If these components are not already present on your system, the easiest way to get them is to install the latest version of Microsoft Internet Explorer.

Installation

To install the SecurePost object, copy the ATSSecurePost.dll file to the location where you want the object to reside (typically the same location where your application resides, or in the Windows SYSTEM or SYSTEM32 directory), and register it in the Windows Registry by running

```
REGSVR32 ATSSecurePost.dll
```

After executing REGSVR32 from a command-prompt, you should see the following dialog:



DllRegisterServer in ATSSecurePost.dll succeeded.

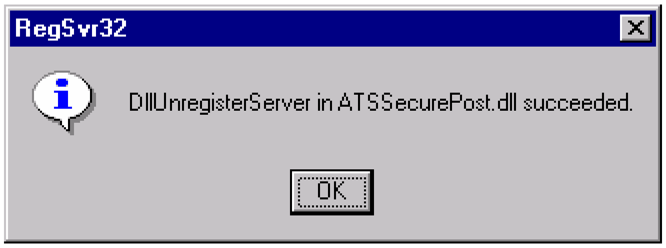
If you get an error message during the registration process, your system contains an old version of the Microsoft Internet Extensions (WININET) libraries. In this case, install the latest version of Microsoft Internet Explorer or its updates.

Uninstalling

To uninstall the SecurePost object, unregister the object by executing the following command, and then delete the ATSSecurePost.dll file:

```
REGSVR32 /u ATSSecurePost.dll
```

If the command is successful, you will see the following dialog:



DllUnregisterServer in ATSSecurePost.dll succeeded.

Interfaces

The SecurePost object exposes a single COM (Component Object Model) interface through a class named *SecurePost Class*. This interface provides several methods for processing transactions, as well as a large number of properties that can be manipulated. There are also several methods that can be used for debugging and testing during the development cycle.

Properties

Name	Type & Size	Default	Required	Purpose and Online Commerce Suite tag
ATSID	String(5)	None	✓[1]	Online Commerce Suite Account ID, always 5 characters long. (ATSID)
ATSSubID	String(5)	None		Online Commerce Suite Sub ID, 5 characters long or empty. (SubID)
CCName	String(32)	None	▪[2]	Credit Card Account Name. (CCName)
CCNumber	String(32)	None	▪	Credit Card Number. (CCNum)
ExpMonth	String(2)	None	▪	Credit Card Expiration Month. (ExpMon)
ExpYear	String(4)	None	▪	Credit Card Expiration Year. (ExpYear)
CVV2	String(3)	None		Credit Card CVV2 Code (CVV2)
CKName	String(32)	None	⅄[3]	Checking Account Name. (CKName)
CKRoutingNumber	String(9)	None	⅄	Checking Account Routing (ABA) Number, always 9 digits long. (CKABA)
CKAccountNumber	String(18)	None	⅄	Checking Account Number. (CKAcct)
RecurBillingCycle	Short Integer	0	★[4]	Billing cycle code for recurring billing. (Recur_Billingcycle) 0=No recurring billing 1=Weekly 2=Monthly 3=Quarterly 4=Semi-Annual 5=Annual 6=Bi-Weekly (14 days) 7=Bi-Annual (2 years) 8=Quad-Weekly (28 days)
RecurBillingMax	Short Integer	-1	★	Maximum number of billing cycles before the

[1] A checkmark ('✓') indicates that this property must be set for all transactions.

[2] A square ('▪') indicates that this property must be set for all credit card transactions.

[3] A triangle ('⅄') indicates that this property must be set for all checking account transactions.

[4] A star ('★') indicates that this property must be set for all recurring transactions.

Name	Type & Size	Default	Required	Purpose and Online Commerce Suite tag
				recurring transaction will automatically be deleted. A value of –1 indicates that there is no maximum, i.e. billing continues indefinitely. (Recur_Billingmax)
RecurStart	Short Integer	-1	★	Number of days between the initial transaction, and the start of the billing cycle. (Recur_Start)
RecurAmount	Short Integer >= 100	0	★	The amount that will be charged during each recurring billing in cents, i.e. $14.95 is specified as 1495. The minimum amount is $1.00. (Recur_Amount)
Amount	Short Integer > 0	0	✓	The amount to be charged to the credit card. (Amount)
CI_CompanyName	String(64)	None		The consumer's company name. (CI_CompanyName)
CI_BillAddr1	String(64)	None	◆[5]	The consumer's billing address. (CI_BillAddr1)
CI_BillAddr2	String(64)	None	◆	The consumer's billing address. (CI_BillAddr2)
CI_BillCity	String(32)	None	◆	The consumer's billing city. (CI_BillCity)
CI_BillState	String(32)	None	◆	The consumer's billing state. (CI_BillState)
CI_BillZip	String(16)	None	◆	The consumer's billing zip code. (CI_BillZip)
CI_BillCountry	String(32)	None	◆	The consumer's billing country. (CI_BillCountry)
CI_ShipAddr1	String(64)	None		The consumer's shipping address. (CI_ShipAddr1)
CI_ShipAddr2	String(64)	None		The consumer's shipping address. (CI_ShipAddr2)
CI_ShipCity	String(32)	None		The consumer's shipping city. (CI_ShipCity)
CI_ShipState	String(32)	None		The consumer's shipping state. (CI_ShipState)
CI_ShipZip	String(16)	None		The consumer's shipping zip code. (CI_ShipZip)
CI_ShipCountry	String(32)	None		The consumer's shipping country. (CI_ShipCountry)
CI_Phone	String(16)	None		The consumer's phone number. (CI_Phone)
CI_Email	String(64)	None		The consumer's e-mail address. (CI_Email)
CI_Memo	String(4096)	None		The consumer's memo. (CI_Memo)
CI_DLNum	String(32)	None		The consumer's Drivers License number. (CI_DLNum)
CI_SSNum	String(32)	None		The consumer's Social Security number. (CI_SSNum)
CI_IPAddress	String(16)	None		The consumer's IP address. (CI_IPAddress)
EmailSubject	String(256)	None		The e-mail message subject of the transaction acknowledgement sent to the consumer. (EmailSubject)
EmailText	String(4096)	None		The e-mail message text of the transaction acknowledgement sent to the consumer. (EmailText)
MerchantReferrerInfo	String(64)	None		The Merchant Referrer Information. (MerchantReferrerInfo)
MerchantOrderNumber	String(64)	None		The Merchant Order Number. If non-empty, this must be a value that is unique for the *ATSID* / *ATSSubID* combination. (MerchantOrderNumber)
MerchantOrderType	String(64)	None		The Merchant Order Type. (MerchantOrderType)
ResultAccepted	Read-Only Boolean			TRUE if the transaction was approved, FALSE if it was declined.
ResultRefCode	Read-Only String			The reference code of the transaction.
ResultOrderID	Read-Only String			The Order ID assigned to the transaction.
ResultTransID	Read-Only			The Transaction ID assigned to the

[5] A diamond ('◆') indicates that this property may be required if the Address Verification System (AVS) option has been enabled for the merchant account.

Name	Type & Size	Default	Required	Purpose and Online Commerce Suite tag
	String			transaction.
ResultAuthCode	Read-Only String			The authorization code of the transaction.
DevMode	Boolean	False		Determines whether the SecurePost object runs in development mode.
AVSOverride	Boolean	False		Overrides the Address Verification System (AVS) fraud protection, allowing transactions that would ordinarily be declined to be processed anyway.
ResultErrorFlag	Read-Only Short Integer			A non-zero value indicates an error occurred during processing.
LastError	Read-Only String			A string describing the error that occurred.
TransactionTimeout	Short Integer >= 5 & <= 600	300		The maximum number of seconds that the SecurePost object will wait for a response from the authorization network. NOTE: If a reply is not received within the timeout period, the SecurePost object will return an error. However, the authorizing network may continue to process the transaction, and eventually approve or decline it. You can retrieve the result later using the *GetTransactionResult* method.

Methods

ProcessSale

Processes a sale using the properties currently set. After the transaction has been processed, you can examine the *ResultErrorFlag*, *ResultAccepted*, *ResultRefCode*, *ResultOrderID*, *ResultTransID*, and *ResultAuthCode* properties to get detailed information about the transaction.

NOTE: Whether the transaction is a credit card or electronic check (ACH) transaction depends on the properties that are set. Setting any of the credit card properties will clear all electronic check properties, and vice versa, so the last property that was set controls the transaction type.

Minimum Required Properties for ProcessSale for Credit Card

Name	Type & Size	Default	Required	Purpose and Online Commerce Suite tag
ATSID	String(5)	None	✓	Online Commerce Suite Account ID, always 5 characters long. (ATSID)
Amount	Short Integer > 0	0	✓	The total amount to be authorized.
CCName	String(32)	None	✓	Credit Card Account Name. (CCName)
CCNumber	String(32)	None	✓	Credit Card Number. (CCNum)
ExpMonth	String(2)	None	✓	Credit Card Expiration Month. (ExpMon)
ExpYear	String(4)	None	✓	Credit Card Expiration Year. (ExpYear)
ResultAccepted	Read-Only Boolean			TRUE if the transaction was approved, FALSE if it was declined.
ResultRefCode	Read-Only String			The reference code of the transaction.
ResultOrderID	Read-Only String			The Order ID assigned to the transaction.
ResultTransID	Read-Only String			The Transaction ID assigned to the transaction.
ResultAuthCode	Read-Only String			The authorization code of the transaction.
ResultErrorFlag	Read-Only Short Integer			A non-zero value indicates an error occurred during processing.
LastError	Read-Only String			A string describing the error that occurred.

ProcessAuth

Processes an authorization using the properties currently set. After the transaction has been processed, you can examine the *ResultErrorFlag*, *ResultAccepted*, *ResultRefCode*, *ResultOrderID*, *ResultTransID*, and *ResultAuthCode* properties to get detailed information about the transaction.

Minimum Required Properties for ProcessAuth

Name	Type & Size	Default	Required	Purpose and Online Commerce Suite tag
ATSID	String(5)	None	✓	Online Commerce Suite Account ID, always 5 characters long. (ATSID)
Amount	Short Integer > 0	0	✓	The total amount to be authorized.
CCName	String(32)	None	✓	Credit Card Account Name. (CCName)
CCNumber	String(32)	None	✓	Credit Card Number. (CCNum)
ExpMonth	String(2)	None	✓	Credit Card Expiration Month. (ExpMon)
ExpYear	String(4)	None	✓	Credit Card Expiration Year. (ExpYear)
ResultAccepted	Read-Only Boolean			TRUE if the transaction was approved, FALSE if it was declined.
ResultRefCode	Read-Only String			The reference code of the transaction.
ResultOrderID	Read-Only String			The Order ID assigned to the transaction.

Name	Type & Size	Default	Required	Purpose and Online Commerce Suite tag
ResultTransID	Read-Only String			The Transaction ID assigned to the transaction.
ResultAuthCode	Read-Only String			The authorization code of the transaction.
ResultErrorFlag	Read-Only Short Integer			A non-zero value indicates an error occurred during processing.
LastError	Read-Only String			A string describing the error that occurred.

ProcessPost (BSTR RefCode)

Processes a post for a previous auth-only transaction. The only properties that are used and must be set are the *ATSID* and *ATSSubID*. In addition, the ProcessPost method requires as an argument the *ResultRefCode* that was returned by the corresponding *ProcessAuth* call. After the transaction has been processed, you can examine the *ResultErrorFlag*, *ResultAccepted*, *ResultRefCode*, *ResultOrderID*, *ResultTransID*, and *ResultAuthCode* properties to get detailed information about the transaction.

Minimum Required Properties for ProcessPost

Name	Type & Size	Default	Required	Purpose and Online Commerce Suite tag
ATSID	String(5)	None	✓	Online Commerce Suite Account ID, always 5 characters long. (ATSID)
ResultAccepted	Read-Only Boolean			TRUE if the transaction was approved, FALSE if it was declined.
ResultRefCode	Read-Only String		✓	The reference code of the transaction. This value is passed as an argument to the ProcessPost method
ResultOrderID	Read-Only String			The Order ID assigned to the transaction.
ResultTransID	Read-Only String			The Transaction ID assigned to the transaction.
ResultAuthCode	Read-Only String			The authorization code of the transaction.
ResultErrorFlag	Read-Only Short Integer			A non-zero value indicates an error occurred during processing.
LastError	Read-Only String			A string describing the error that occurred.

ProcessCredit (BSTR TransID, BSTR OrderID)

Processes a credit for an account. The only properties that are used and must be set are the *ATSID* and *ATSSubID*. In addition, the ProcessCredit method requires as arguments the *ResultTransID* and *ResultOrderID* of the transaction to be credited. After the transaction has been processed, you can examine the *ResultErrorFlag*, *ResultAccepted*, *ResultRefCode*, *ResultOrderID*, *ResultTransID*, and *ResultAuthCode* properties to get detailed information about the transaction.

Minimum Required Properties for ProcessCredit

Name	Type & Size	Default	Required	Purpose and Online Commerce Suite tag
ATSID	String(5)	None	✓	Online Commerce Suite Account ID, always 5 characters long. (ATSID)
ResultAccepted	Read-Only Boolean			TRUE if the transaction was approved, FALSE if it was declined.
ResultRefCode	Read-Only String			The reference code of the transaction.
ResultOrderID	Read-Only String		✓	The Order ID assigned to the transaction. Passed as an argument.
ResultTransID	Read-Only String		✓	The Transaction ID assigned to the transaction. Passed as an argument

Name	Type & Size	Default	Required	Purpose and Online Commerce Suite tag
ResultAuthCode	Read-Only String			The authorization code of the transaction.
ResultErrorFlag	Read-Only Short Integer			A non-zero value indicates an error occurred during processing.
LastError	Read-Only String			A string describing the error that occurred.

ProcessVoid (BSTR TransID)

Processes a void of a previous transaction. The only properties that are used and must be set are the *ATSID*, *ATSSubID,* and *Amount.* In addition, the ProcessVoid method requires as an argument the *ResultTransID* that was returned by the corresponding *ProcessSale*, *ProcessAuth*, or *ProcessPost* call. After the transaction has been processed, you can examine the *ResultErrorFlag*, *ResultAccepted*, *ResultRefCode*, *ResultOrderID*, *ResultTransID*, and *ResultAuthCode* properties to get detailed information about the transaction.

Minimum Required Properties for ProcessVoid

Name	Type & Size	Default	Required	Purpose and Online Commerce Suite tag
ATSID	String(5)	None	✓	Online Commerce Suite Account ID, always 5 characters long. (ATSID)
ResultAccepted	Read-Only Boolean			TRUE if the transaction was approved, FALSE if it was declined.
ResultRefCode	Read-Only String			The reference code of the transaction.
ResultOrderID	Read-Only String			The Order ID assigned to the transaction.
ResultTransID	Read-Only String		✓	The Transaction ID assigned to the transaction. Passed as an argument.
ResultAuthCode	Read-Only String			The authorization code of the transaction.
ResultErrorFlag	Read-Only Short Integer			A non-zero value indicates an error occurred during processing.
LastError	Read-Only String			A string describing the error that occurred.

OnStartPage

For ASP support – called automatically by the Web server when a page is opened. This method should not be called directly.

OnEndPage

For ASP support – called automatically by the Web server when a page is closed. This method should not be called directly.

ClearAll

Clears the values of all properties, and resets them to their default values.

GenerateRuntimeError(short ErrorCode)

Generates a runtime error for the purpose of debugging the error handling code in your application. The supported ErrorCode values are as follows:

ErrorCode	Error	Description
0	None	No error is generated
1	E_UNEXPECTED	Catastrophic failure
2	E_NOTIMPL	Not implemented
3	E_OUTOFMEMORY	Ran out of memory
4	E_INVALIDARG	One or more arguments are invalid
5	E_NOINTERFACE	No such interface supported

6	E_POINTER	Invalid pointer
7	E_HANDLE	Invalid handle
8	E_ABORT	Operation aborted
9	E_FAIL	Unspecified error
10	E_ACCESSDENIED	General access denied error
11	E_PENDING	The data necessary to complete this operation is not yet available

A full explanation of these common COM error codes and their meaning is beyond the scope of this document. In addition, the error message generated in your programming environment in the absence of your own error handler may vary from that shown in the Description column above.

Finally, not all of these errors can result from interacting with the SecurePost object. The only errors that should ever be generated are E_UNEXPECTED, E_NOTIMPL, and E_INVALIDARG. Support for other errors is included solely for the convenience of the developer writing the error handler.

ForceAccept(BSTR ApprovalCode)

Forces an approval of the next call to ProcessSale or ProcessAuth with the specified approval code. The data properties are not actually transmitted for processing, and the values of the *ResultRefCode*, *ResultOrderID*, and *ResultTransID* properties are not meaningful following the call. This method should be used for testing only.

ForceDecline(BSTR Reason)

Forces a decline of the next call to ProcessSale or ProcessAuth for the specified reason. The data properties are not actually transmitted for processing. This method should be used for testing only.

GetTransactionResult

Retrieves the result of a previously processed transaction given the current *ATSID*, *ATSSubID*, and *MerchantOrderNumber* properties. After the results have been retrieved, you can examine the *ResultErrorFlag*, *ResultAccepted*, *ResultRefCode*, *ResultOrderID*, *ResultTransID*, and *ResultAuthCode* properties to get detailed information about the transaction. If the transaction engine responds to the *GetTransactionResult* request, but the transaction cannot be found, both the *ResultErrorFlag* and *ResultAccepted* properties will be FALSE, and the *ResultAuthCode* property will be blank.

NOTE: In order to uniquely identify the transaction for which the results are to be retrieved, you must have assigned a unique *MerchantOrderNumber* property when the transaction was originally processed. If the *MerchantOrderNumber* property was left blank or wasn't unique, there is no way to retrieve the results.

Error Handling

The SecurePost object validates all properties to ensure that their values are within the permissible range. For example, all Online Commerce Suite Account IDs (the *ATSID* property) are always exactly five characters long, so any attempt to set the property to a value that is shorter or longer than five characters will generate an error of type E_INVALIDARG.

How errors are handled depends entirely on the environment in which the SecurePost object is being used: in C++, you will have to provide an exception handler using a *try* / *catch* block, and in Visual Basic, you will need an error handler using *On Error Goto*. For details, see the examples in the next section.

Examples

All of the examples below send a transaction for a $19.95 credit card sale using a ficticious MasterCard credit card to the credit card processor.

Visual C++

The example below assumes that a wrapper class called ISecurePost was created in the Class Wizard of Visual Studio from the type library or DLL.

```
HRESULT     hr;
IUnknown*   pIUnknown;
IDispatch*  pIDispatch;

const CLSID CLSID_ATSSecurePost =
{0x94A1A587,0x1CF1,0x11D3,{0x99,0x3D,0x00,0xE0,0x29,0x1F,0x9A,0x9C}};

try {
    hr = CoCreateInstance(CLSID_ATSSecurePost, NULL, CLSCTX_INPROC_SERVER, IID_IUnknown,
(LPVOID *) &pIUnknown);

    if (SUCCEEDED(hr))
        {
        hr = pIUnknown->QueryInterface(IID_IDispatch, (LPVOID *) &pIDispatch);

        if (SUCCEEDED(hr))
            {
            ISecurePost     MyObject(pIDispatch);

            try {
                MyObject.SetATSID("TEST0");
                MyObject.SetAmount(1995);
                MyObject.SetCCName("John Doe");
                MyObject.SetCCNumber("5454545454545454");
                MyObject.SetExpMonth("10");
                MyObject.SetExpYear("2002");

                MyObject.ProcessSale();

                if (MyObject.GetResultAccepted())
                    MessageBox(MyObject.GetResultAuthCode(),"Accepted", MB_OK);
                else if (MyObject.GetResultErrorFlag())
                    MessageBox(MyObject.GetLastError(),"Error", MB_OK | MB_ICONERROR);
                else
                    MessageBox(MyObject.GetResultAuthCode(),"Declined", MB_OK);
                }
            catch (...)
                {
                MessageBox(MyObject.GetLastError(),"Error", MB_OK | MB_ICONERROR);
                }
            }

        pIUnknown->Release();
        }
    }
catch (...)
    {
    }
```

Visual Basic

Visual Basic supports both early and late binding of objects. The example shown here uses the late binding by instantiating the object based on the class name of the SecurePost object.

```
Dim MyObject As ATSSECUREPOSTLib.SecurePost

On Error GoTo ErrHandler

Set MyObject = CreateObject("ATS.SecurePost")

MyObject.ATSID = "TEST0"
MyObject.Amount = 1995
MyObject.CCName = "John Doe"
MyObject.CCNumber = "5454545454545454"
MyObject.ExpMonth = "10"
MyObject.ExpYear = "2002"

MyObject.ProcessSale

If MyObject.ResultAccepted Then
    MsgBox "Accepted: " + MyObject.ResultAuthCode
Else
    If MyObject.ResultErrorFlag Then
        MsgBox "Error: " + MyObject.LastError
    Else
        MsgBox "Declined: " + MyObject.ResultAuthCode
    End If
End If

ErrHandler:

Set MyObject = Nothing
```

Active Server Pages

The Active Server Pages example shown here processes a sale, and displays the result of the transaction in HTML:

```
<%
Set MyObject = CreateObject("ATS.SecurePost")

MyObject.ATSID = "TEST0"
MyObject.Amount = 1995
MyObject.CCName = "John Doe"
MyObject.CCNumber = "5454545454545454"
MyObject.ExpMonth = "10"
MyObject.ExpYear = "2002"

MyObject.ProcessSale

%>
<HTML>
<BODY>
<%

If MyObject.ResultAccepted Then
    %>
    Accepted: <%=MyObject.ResultAuthCode%>
    <%
Else
    If MyObject.ResultErrorFlag Then
        %>
        Error: <%=MyObject.LastError %>
        <%
    Else
        %>
        Declined: <%=MyObject.ResultAuthCode%>
        <%
    End If
End If

%>
</BODY>
</HTML>
<%

Set MyObject = Nothing
%>
```

Microsoft Office Applications

The sample shown below can be used from within any Microsoft Office application, such as Microsoft Word or Microsoft Excel. The version of Basic available for scripting Microsoft Office applications is a subset of the full Visual Basic language, and does not support early binding. The Microsoft Office Applications example differs only slightly from the Visual Basic version – in this example, the SecurePost object is declared as a generic object, rather than tying it to a specific type library.

```
Function Microsoft_Office_Example()

Dim MyObject As Object

On Error GoTo ErrHandler

Set MyObject = CreateObject("ATS.SecurePost")

MyObject.ATSID = "TEST0"
MyObject.Amount = 1995
MyObject.CCName = "John Doe"
MyObject.CCNumber = "5454545454545454"
MyObject.ExpMonth = "10"
MyObject.ExpYear = "2002"

MyObject.ProcessSale

If MyObject.ResultAccepted Then
    MsgBox "Accepted: " + MyObject.ResultAuthCode
Else
    If MyObject.ResultErrorFlag Then
        MsgBox "Error: " + MyObject.LastError
    Else
        MsgBox "Declined: " + MyObject.ResultAuthCode
    End If
End If

ErrHandler:

Set MyObject = Nothing

End Function
```

The example shown above is included in this document, and can be executed by pressing Alt-F11 to access Visual Basic, selecting the *Microsoft_Office_Example* function from the *ThisDocument* scope, and pressing F8 to step through the code one line at a time.

Copyright Notice

© 2003, MerchantPartners.com LLC All Rights Reserved.

Online Commerce Suite, Online Merchant Center, Online Check, and Online Charge are registered trademarks of MerchantPartners.com LLC

Appendix A: Transaction Authorization Specification.

Credit Card Approval response format

The transaction approval authorization response message consists of a string of eight fields delimited by the colon ":" character.

Here is an example of the format of the complete approval message:

```
AVSSALE:123456:1234567890123:9:12345678:Y:AUTHNETSPECIFIC:M
```

The following table describes each of the fields returned in the approval response message.

Transaction Approval Authorization Response Format

Field	Description	Value
Transaction Type	Type of transaction submitted	SALE AVSSALE AUTH AVSAUTH POST AVSPOST VOICEPOST VOID CREDIT
Authorization Code	The six digit authorization or approval code provided by the authorizing network	Varies
Reference Number	Additional reference information provided by the authorizing network	Varies
Batch Number	Batch settlement number in which this transaction is included	Number
Transaction ID	Unique number assigned by the Online Commerce Suite to this transaction.	Number
AVS Result Code	Result code generated by the Address Verification System.	See Appendix B: AVS response codes
Auth Net Specific	Miscellaneous auth net message	
CVV2/CVC2 Result Code	One character result code generated by the CVV2/CVC2 system	See Appendix C: CVV2/CVC2 Response Codes

Credit Card Decline response format

The transaction decline authorization response message consists of the string "DECLINE" followed by two fields delimited by the colon ":" character.

Here is an example of the format of the complete approval message:

```
DECLINED:1234567890:TEXT RESPONSE
```

The following table describes each of the fields returned in the approval response message.

Transaction Decline Authorization Reponse

Field	Description	Value	
Transaction Result	Result of the transaction	DECLINE	
Decline Code	10 digit decline code.	**First Digit:**	
		0	Authorizing network declined the transaction
		1	Gateway declined the transaction
		2	Authorizing network returned an error, forcing a decline
		3	Gateway returned an error, forcing a decline
		Digits 2-10	Internal decline number
Text Response	Text message indicating the reason for the decline.	Varies	

Appendix B: AVS Response Codes

The following table defines AVS response codes returned from the Address Verification System.

Response Code	Definition
A	Street addresses matches, but the ZIP code does not. The first five numerical characters contained in the address match. However, the ZIP code does not match.
E	Ineligible transaction. The card issuing institution is not supporting AVS on the card in question.
N	Neither address nor ZIP matches. The first five numerical characters contained in the address do not match, and the ZIP code does not match.

R	Retry (system unavailable or timed out).
S	Card type not supported. The card type for this transaction is not supported by AVS. AVS can verify addresses for Visa cards, MasterCard, proprietary cards, and private label transactions.
U	Address information unavailable. The address information was not available at the issuer.
W	9 digit ZIP code match, address does not. The nine digit ZIP code matches that stored at the issuer. However, the first five numerical characters contained in the address do not match.
X	Exact match (9 digit zip and address) Both the nine digit postal ZIP code as well as the first five numerical characters contained in the address match.
Y	Address and 5 digit zip match. Both the five digit postal ZIP code as well as the first five numerical characters contained in the address match.
Z	5 digit ZIP matches, but the address does not. The five digit postal ZIP code matches that stored at the VIC or card issuer's center. However, the first five numerical characters contained in the address do not match.
FOREIGN CODES:	
B	Street address matches for international transaction. Postal Code not verified due to incompatible formats.
C	Street address and Postal Code not verified for international transaction due to incompatible format.
D	Street address and Postal Code match for international transaction.
P	Postal Code match for international transaction. Street address not verified due to incompatible formats.

Appendix C: CVV2/CVC2 Response Codes

The following table defines CVV2/CVC2 response codes returned from the credit card authorizing network.

Response Code	**Definition**
Space	CVV2 processing not requested
M	CVV2/CVC2 Match
N	CVV2/CVC2 not matched
P	Not processed
S	CVV2 should be printed on the card, but it was indicated that the value was not present
U	Issuer does not support CVV2
X	Service provider did not respond