IBM WebSphere Commerce Payments for Multiplatforms

Cassette for CyberCash Supplement

Version 3.1

IBM WebSphere Commerce Payments for Multiplatforms

Cassette for CyberCash Supplement

Version 3.1

Note

Before using this information and the product it supports, be sure to read the general information under Appendix C, “Notices” on page 65.

Third Edition (July, 2002)

This edition applies to Version 3.1.3 of the Cassette for CyberCash for IBM® WebSphere® Commerce Payements for Multiplatforms and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

Preface

This book is for users and administrators of the Cassette for CyberCash, who are responsible for installation and implementation. This information will help you understand what you need to use the Cassette for CyberCash. Programmers who are responsible for developing applications to manage IBM WebSphere Commerce Payments will find cassette-specific information regarding parameter requirements and cassette-specific states useful. This book serves as a supplement to the *IBM WebSphere Commerce Payments Administrator's Guide for Multiplatforms, Version 3.1* as well as the *IBM WebSphere Commerce Payments Programmer's Guide and Reference for Multiplatforms, Version 3.1*.

Note:
IBM WebSphere Commerce Payments for Multiplatforms (hereafter called WebSphere Commerce Payments) was previously known as IBM WebSphere Payment Manager for Multiplatforms. Starting with version 3.1.3, the payments application was renamed to WebSphere Commerce Payments and references to the product were changed throughout this document. References to the former product may still appear in this document and apply to earlier releases of the product.

The Cassette for CyberCash allows users of the IBM WebSphere Commerce Payments to access the CyberCash, Inc. CashRegister Credit Card Service through the WebSphere Commerce Payments programming and user interfaces. In order to use the Cassette for CyberCash, merchants must have a CyberCash-enabled merchant bank relationship in North America. Overseas merchants may be able to qualify for North American bank relationships. For a list of financial institutions that can provide CyberCash-enabled merchant accounts, please refer to the CyberCash Web site: **http://www.cybercash.com/fi_display/home.html**.

Conventions in this book

Table 1. Conventions used in this document.

Boldface	Indicates the name of the item you need to select, the name of a field, or a string you must enter.
Italics	Indicates book titles or variable information that must be replaced by an actual value.
`Monospace`	Indicates an example, a portion of a file, or a previously entered value.

Terminology

Enabling electronic shopping requires different participants and software components:

acquirer
An organization that provides card authorization and payment capture services for merchants. A merchant normally wants to accept more than one credit card brand, but does not want to deal with multiple bankcard associations and so uses acquirer services. These services include verbal or electronic telephone authorization support, electronic payment transfer to the merchant's account, and CyberCash protocol support for one or more

electronic payment protocols. Acquirer services are paid for by the merchant in the form of a small percentage charge on each transaction.

brand Brand recognition and loyalty are the keys to credit card marketing. Some brands are owned by a single financial organization, which is also the card issuer. Other brands are owned by bankcard associations, consortiums of financial institutions that promote and advertise the brand, establish operating rules, and provide a network for payment authorization and fund transfers.

card processor
An agent for an acquirer to whom merchants send their transaction requests. Provides much of the administrative and organizational infrastructure by which merchants process their transactions. Accessed by the WebSphere Commerce Payments and the Cassette for CyberCash through the CashRegister.

cardholder
A person with a valid payment card account who uses a browser to shop for goods and services on the Internet.

CyberCash CashRegister
The CyberCash CashRegister (or simply ″CashRegister″) is a service running on systems which are physically located at CyberCash, Inc. The CashRegister receives requests from online merchants to process payment card transactions and interfaces with the established financial network to process those transactions.

financial network
The aggregate of card processors, acquirers, card issuers and other institutions or organizations, through which payment card transaction processing is traditionally performed.

gateway
A gateway represents an acquirer or card processor to the CyberCash CashRegister, acting as a ″translator″ between the messages used by the CyberCash CashRegister and those used on the financial network. (Note that merchants never communicate directly with a gateway.) Once a transaction request is received from a merchant, the CashRegister sends some type of request to the processor via the gateway. When the gateway receives the message, it translates the request into the format used on the financial network and then forwards that message to the processor. Responses from the processor are translated and sent back in a form that the CashRegister understands. All of the gateways mentioned in this document are operated by CyberCash, Inc.

issuer The financial institution that provides the cardholder with the credit card. Ultimately, it is the issuer who is responsible for the cardholder debt payment. For example, the issuer balances the risk of a cardholder defaulting against the income from interest payments. The cardholder need not have a relationship with the issuer except for the credit card account, but in practice, most cardholders have at least one credit card from the bank that holds their checking account.

merchant
A person or organization that has goods or services to sell to the cardholder.

merchant server

The merchant server displays products for sale to a global market over the Internet. Shoppers can browse through catalogs, adding items to their shopping cart as they go along, and make their purchases when convenient. Merchants can customize their electronic stores, providing discounts for quantity purchases and seasonal merchandise or targeting consumer groups, such as frequent shoppers. They can also track demographic information from data provided by the shoppers and use it for marketing strategies.

payment server
A payment server handles and stores payment information. All transactions from the consumer using the wallet flow directly to the merchant's payment server. The payment server accepts payments from the cardholder via the Internet and passes this information along to financial institutions for approval. The payment server also maintains records of all transactions. Both the CyberCash CashRegister and the IBM WebSphere Commerce Payments are payment servers.

You should also be familiar with terms used in the credit card industry, including:

Authorize
The cardholder is given permission to make a purchase by the financial institution and the merchant has some guarantee that it will receive funds. It is the validation of the cardholder for a given purchase. The process involves assessing transaction risk, confirming that a given transaction does not raise the account holder's debt above the account credit limit, and reserving the specified amount of credit.

Batch A collection of financial transactions grouped for administrative and record-keeping purposes.

Capture
Funds can be moved or deposited to the merchant's account.

Credit The merchant needs to return money to the cardholder following a valid capture transaction. For example, if goods are returned or are defective, the cardholder receives credit.

Additional information

The WebSphere Commerce Payments product package includes information that describes the cassette-independent functions of WebSphere Commerce Payments:

- The *IBM WebSphere Commerce Payments for Multiplatforms Installation Guide Version 3.1* (WebSphere Commerce Payments Installation Guide) provides information for installing, migrating, and starting the WebSphere Commerce Payments.
- The *IBM WebSphere Commerce Payments for Multiplatforms Programmer's Guide, Version 3.1* (WebSphere Commerce Payments Programmer's Guide) provides details about the WebSphere Commerce Payments API.
- The *IBM WebSphere Commerce Payments for Multiplatforms Administrator's Guide Version 3.1* (WebSphere Commerce Payments Administrator's Guide) contains conceptual information and shows how to configure WebSphere Commerce Payments using the user interface.

This book serves as a supplement to the above books, describing cassette-specific programming or administrative considerations and showing how the Cassette for CyberCash implements WebSphere Commerce Payments generic commands.

All documents are available on the WebSphere Commerce Payments CD-ROMs in Portable Document Format (PDF). For the latest Acrobat reader, see: http://www.adobe.com. On iSeries™ systems, the documentation is compressed into a save file and is only available after WebSphere Commerce Payments has been installed.

The CyberCash CashRegister Service and the Merchant Connection Kit are described in the following documents, all of which are available at **http://www.cybercash.com**:

- *Development Guide*: Provides detailed Merchant Connection Kit integration and API information.
- *Error Message Guide*: Explains how to write code to handle error conditions and provides lists and descriptions of error messages from the CashRegister and Gateway.

In addition to this manual, you can reference the following manual and related Web sites:

- **http://www.ibm.com/payment/** provides more information on the IBM Payment Processing products.
- **http://www.software.ibm.com/commerce/payment/support/serv/index.html** provides current WebSphere Commerce Payments technical information and links to the latest softcopy views of all WebSphere Commerce Payments documentation.
- **http://www.ibm.com/software/webservers/appserv/library.html** provides documentation links for IBM WebSphere Application Server.
- **http://www-4.ibm.com/software/data/db2/library/** provides documentation links for IBM Universal Database.
- **http://www.ibm.com/software/webservers/commerce/payment/** points to the financial institutions that have established payment gateways to handle WebSphere Commerce Payments payment processing needs.
- **http://www.cybercash.com** provides information on CyberCash services.
- **http://amps.cybercash.com** provides the site for CyberCash CashRegister merchant registration.

Chapter 1. Overview of the CyberCash CashRegister Service

This chapter serves as an introduction to the CyberCash CashRegister Service and its online payment card service. Transaction processing models are defined and the relationships and registration process of a CyberCash merchant is discussed. Finally, information on a payment card fraud prevention tool, the Address Verification Service (AVS), is presented.

The CyberCash CashRegister Service is an online payment processing service, offered by CyberCash, Inc. This service allows online merchants to accept payment in a variety of electronic forms, including electronic checks, micropayments and payment cards. The most widely used of these services is the payment card service, which enables online merchants to receive payments for goods and services using information supplied by the cardholder.

The CashRegister payment card service uses traditional credit and debit cards, within the existing North American card processing infrastructure. CyberCash-enabled merchants send their transactions to the CashRegister and in turn, the CashRegister interfaces with *card processors*. Card processors are agents of acquiring financial institutions (called *acquirers*), through Internet-based access points, known as *gateways*. The aggregate of the gateway, card processors, acquirers and card-issuing financial institutions comprise a *financial network*.

The CashRegister acts as a front end to the existing infrastructure of financial networks, financial institutions and payment processors. Payments are made using merchant-collected information, where merchant software asks for card information over a secured connection or an electronic wallet.

For more information on the CyberCash CashRegister Service, see **http://www.cybercash.com**.

Transaction processing models

The CashRegister Service supports three transaction processing models, which are based on the different models supported by the card processors and acquirers to which the CashRegister provides access. The differences between these models are where the authorized transactions are stored before being captured, and the action required to capture funds.

HostCapture/AuthCapture
: Acquirer controls batch processing and all authorization requests immediately capture funds (that is, only sale transactions are supported).

HostCapture/PostAuth
: Acquirer controls batch processing and a separate deposit request is required to capture the funds after a payment is authorized.

Terminal Capture
: Merchant controls batch processing.

The decision about which model a merchant uses is made by agreement between the merchant and the acquiring financial institution.

CyberCash merchants

Before a merchant can accept any credit card payments, regardless of the method by which the transactions are processed, a merchant account with an acquiring financial institution must be established and authorization as a credit card merchant must be completed.

In establishing this account, the merchant and acquirer must agree upon:

- Which credit or debit card brands will be accepted for payment
- Which currencies will be supported
- Which type of processing model will be used (based on the merchant's business model)

Acquirers also need to be informed if the merchant wants to use the CyberCash CashRegister Service to perform online transactions. (The vast majority of processors in the U.S. are enabled for CyberCash transactions.). Once the merchant has been approved for online commerce, the acquirer or processor will forward the required information to CyberCash.

In order to gain access to the CyberCash CashRegister service, the merchant must register online using the CyberCash Integrated Merchant Registration (IMR) process at **http://amps.cybercash.com**. This process assumes that the merchant has already established an account with an acquirer.

Each CyberCash merchant account:

- Represents an account that the merchant has with a specific acquirer through a specific processor.
- Is configured to operate in one of three supported processing models.
- Is configured to process transactions in a single designated currency.
- Is allowed to process some predetermined set of payment card brands.

Upon completing the initial steps of the IMR process, the merchant is given several pieces of information for use in CashRegister transaction processing. The two main values provided to the merchant are:

- The *CyberCash Merchant ID (CCID)*, which identifies the account to the CashRegister Service
- The *Merchant Key*, which is an encryption key that will be used to encrypt each message sent to the CashRegister and decrypt each response from the CashRegister

Note: The IMR also presents the merchant with several other processing options. Among these are *AutoMark* and *AutoSettle*. When accessing the CyberCash CashRegister through the WebSphere Commerce Payments, merchants must *not* select either of these options.

Address verification service

The Address Verification Service (AVS) is a payment card fraud prevention tool that enables mail order and electronic commerce merchants to verify U.S. cardholder shipping addresses against the address that the payment card issuer has on file for the consumer.

The cardholder's billing address, including street address and zip code, are sent in the electronic authorization request message to the issuer. The issuer compares the

street address and zip code to those it has on file and returns an AVS response code to advise you of the comparison status. This information enables decision making that limits risks when shipping merchandise, and risk reduction for the financial institution can result in reduced transaction fees for the merchant. Therefore, the CyberCash CashRegister Service allows the use of this tool by its merchants. It is up to the merchant to decide what risks are allowable if AVS data does not compare favorably.

Note: If you are interested in additional information regarding AVS and merchant chargeback liabilities, contact your acquiring financial institution. See the *CyberCash, Inc, CashRegister Service Development Guide, Version 1.1*, for more information on possible AVS codes.

Chapter 2. CyberCash CashRegister and WebSphere Commerce Payments Concepts

The IBM WebSphere Commerce Payments provides a unified interface through which merchants can use multiple payment protocols in a common way. Each WebSphere Commerce Payments cassette attempts to extract protocol-specific differences so that merchants can ignore disparities between protocols.

This section describes how the Cassette for CyberCash presents the CyberCash CashRegister card-based services through the WebSphere Commerce Payments generic object model and API set. In addition, cassette-specific behaviors and requirements are discussed.

The Cassette for CyberCash implements the payment commands and the payment processing model of the WebSphere Commerce Payments Framework, using the card-based service of the CyberCash CashRegister as the underlying payment type. This implementation:

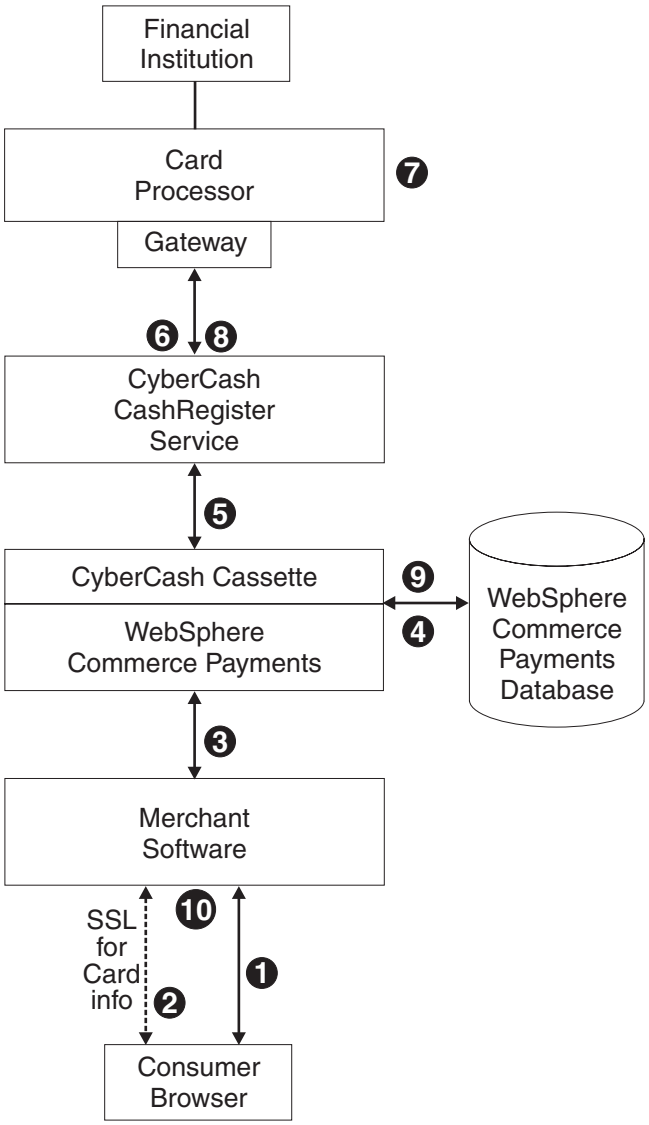
- Enables merchant software that is coded to the WebSphere Commerce Payments API to access CyberCash card-based services with little or no change.
- Supports only the core CyberCash CashRegister card-based services, using merchant-collected card information. Wallet-driven purchases and other CyberCash payment types are not supported.
- Provides intuitive configuration based on key attributes of the CyberCash merchant account.
- Enables all payment administration to be performed using the WebSphere Commerce Payments GUI.

A CyberCash purchase example

The following is an example of how a typical purchase using the Cassette for CyberCash would be processed through the overall system, including the WebSphere Commerce Payments, the Cassette for CyberCash, the CyberCash CashRegister Service and the financial network. This example assumes the HostCapture/AuthCapture processing model, and the use of the AutoApprove and AutoDeposit options of the AcceptPayment command.

Note: Other commands result in different messages being sent to the CashRegister, but the same general flow through the overall system still applies.

.



1. A consumer has been shopping online at a merchant Web site. After choosing several items to purchase, the consumer initiates a purchase, typically by pressing a ″Buy″ button on the shopping page.
2. The merchant software then requests card information from the consumer over a secure (typically SSL-protected) channel. This information includes the credit card number, the card expiration date, the card brand and possibly the cardholder's address.
3. Once this cardholder information has been received, the merchant software invokes the WebSphere Commerce Payments AcceptPayment command with card information and parameters, requesting that the purchase be approved immediately.
4. The WebSphere Commerce Payments and the Cassette for CyberCash record the information they need to execute payment transactions.
5. The Cassette for CyberCash sends an approve and deposit request to the CashRegister.
6. The CashRegister forwards this request to the processor via the gateway.

7. The processor/financial institution processes the request and responds to the CashRegister.
8. The CashRegister records the result, and sends a success response to the cassette.
9. The Cassette for CyberCash, along with the WebSphere Commerce Payments update status in the database and return the success response to the merchant.
10. The merchant software replies to the consumer with an indication that the order is accepted.

WebSphere Commerce Payments object model implementation

This section describes how the Cassette for CyberCash supports the administrative and financial object models that the WebSphere Commerce Payments Framework provides. The mapping of CyberCash-specific information fits very well into the WebSphere Commerce Payments object model.

Administration objects

The WebSphere Commerce Payments administration objects are the entities that comprise the system and merchant configuration under which all financial transactions will be performed. Refer to the *WebSphere Commerce Payments Administrator's Guide for Multiplatforms*, for a complete description of the WebSphere Commerce Payments administration objects. The Cassette for CyberCash augments two of the Framework administration objects with its own attributes and also defines one administration object of its own. Cassette objects are described in detail in Chapter 7, "Using CyberCash objects" on page 49.

Cassette object

The Cassette object represents the cassette itself and contains attributes that apply globally across the cassette. The Cassette for CyberCash extends this object with attributes that tell the cassette how to connect to the CyberCash CashRegister Service.

Account object

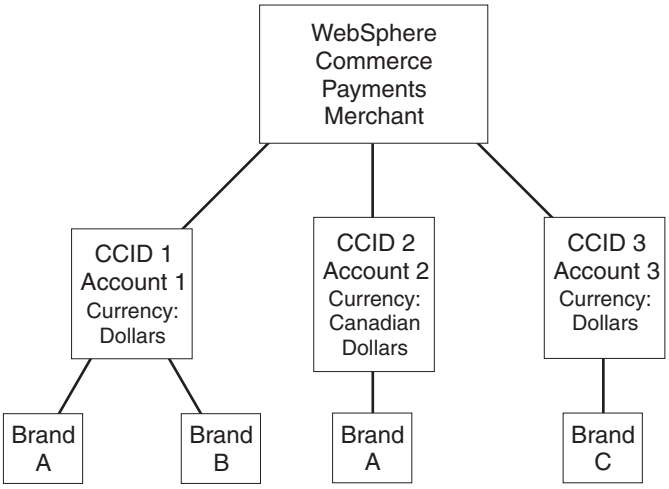
In the WebSphere Commerce Payments object model, the Account object represents a relationship between a given merchant and a given financial institution. This is exactly the type of relationship that each CyberCash merchant account represents. Therefore, the Cassette for CyberCash defines each of its Account objects to represent exactly one CyberCash merchant account. The cassette extends the WebSphere Commerce Payments Account object with attributes that identify and describe the corresponding CyberCash merchant account.

Brand

An important attribute of the CyberCash merchant account is the set of brands that the merchant is allowed to accept for payment. To represent this important element, the Cassette for CyberCash defines a Brand object. Each Brand object represents a single payment card brand that will be accepted for payment by a merchant under the account with which the Brand object is associated.

When a merchant creates a new Order, the brand and currency values specified on the AcceptPayment command will be used to determine which account should be used to process the payment. Since accounts are chosen based on these values, the Cassette for CyberCash requires each WebSphere Commerce Payments merchant's set of CyberCash accounts to have a uniquely defined brand and currency combination. The following figure illustrates this rule. While two different

accounts belonging to the merchant support Brand A, each account supports it under a different currency. Likewise, while two different accounts support the same currency, they each support a different set of brands.



The Cassette for CyberCash implements each Brand object as a MerchantCassetteObject, as defined in the *WebSphere Commerce Payments Programmer's Guide and Reference for Multiplatforms*. Brand objects can be manipulated through the WebSphere Commerce Payments user interface as described in the tutorial in Chapter 4, "Getting Started" on page 25. Through the API, Brand objects are manipulated through the CreateMerchantCassetteObject and DeleteMerchantCassetteObject commands. See "Cassette for CyberCash Commands" on page 41, and "Brand" on page 55, for additional information regarding these commands and how they affect Brand objects.

Financial objects

The WebSphere Commerce Payments financial objects are used to represent the financial transactions executed by merchants. The Cassette for CyberCash provides extensions for each of these financial objects:

- Order objects
- Payment objects
- Credit objects
- Batch objects

For details on how the Cassette for CyberCash extends these payment objects, see Chapter 7, "Using CyberCash objects" on page 49. For descriptions of the payment objects, see the *WebSphere Commerce Payments Administrator's Guide for Multiplatforms*. For programming information, see the *WebSphere Commerce Payments Programmer's Guide and Reference for Multiplatforms*.

Cassette requirements for CyberCash Merchant configuration and operation

When registering as a CyberCash merchant, using the Integrated Merchant Registration (IMR) process, you are presented with a number of processing options. AutoMark and AutoSettle are among these choices. If you plan to access the CyberCash CashRegister via the WebSphere Commerce Payments under this CyberCash merchant account, you must *not* select either of these options.

If you already have a registered, active CyberCash merchant account, then you may need to take one or more of the following steps before using your existing CyberCash merchant account with the WebSphere Commerce Payments and the Cassette for CyberCash:

- Only one WebSphere Commerce Payments should be configured to use any single CyberCash Merchant Identifier (CCID). Using the same CyberCash Merchant Identifier from more than one WebSphere Commerce Payments could produce unpredictable results.
- If you are currently using the AutoMark or the AutoSettle features of the CashRegister, you must disable these before configuring the WebSphere Commerce Payments account.
- Any CyberCash transactions or batches that have been created outside of the WebSphere Commerce Payments will not be recognized by the Cassette for CyberCash, nor will they be visible through the cassette. Therefore, you should ensure that your CyberCash merchant account is in a consistent state (that is, all transactions complete and all batches closed) before initiating your first transaction through the WebSphere Commerce Payments. This will avoid any errors caused by inconsistent views between the CashRegister and the cassette.
- Before using your ″live″ merchant account through the WebSphere Commerce Payments and Cassette for CyberCash, you should create a new test-mode CyberCashID for the merchant account through the CyberCash Merchant Administration Web site. Run some tests to verify the merchant configuration, both at the CashRegister and in the cassette. Once the test account is operating as expected, then you are ready to ″go live″ through the WebSphere Commerce Payments.
- Do not use the CyberCash CashRegister's Payment Administration user interface or any other interface to perform payment processing on transactions that have been created through the WebSphere Commerce Payments while those transactions are in an open batch.
- Do not use the CyberCash CashRegister's Payment Administration user interface or any other interface to initiate batch settlement.
- In order to recover from certain communication failures during batch settlement, the Cassette for CyberCash assumes that the system clock of the local system is within one hour of the CashRegister's system clock (both systems use UTC time). If this is not true, some recovery operations will fail. Therefore, you should ensure that the system clocks of the local system and the CashRegister are set to within one hour of each other

Cassette-specific characteristics and behaviors

This section discusses characteristics of communication parameters, batch processing support and the WebSphere Commerce Payments command set that are unique to the Cassette for CyberCash.

Understanding retry parameters

The Cassette for CyberCash extends the WebSphere Commerce Payments Cassette object with several parameters related to communicating with the CyberCash CashRegister. Four of these parameters control the attempts of the cassette to recover after failed communications or after the CashRegister returns with an indication that a request is pending completion at the gateway or processor. These parameters appear on the CyberCash Cassette Settings screen in the WebSphere Commerce Payments user interface as:

- Read Timeout

- Immediate Retries
- Attempt Interval
- Max Attempts

You can modify any of these values through the user interface (select **Cassettes** under the navigation frame, then select the **CyberCash** cassette icon) or through the ModifyCassette API command. For more information on the ModifyCassette command, see “ModifyCassette” on page 46.

Under direction of these parameters, the cassette will attempt to recover as follows:

- For each command that requires communication with the CyberCash CashRegister, at least one socket connection must be established between the WebSphere Commerce Payments and the CyberCash CashRegister. If the initial connection to the CashRegister for a given command cannot be established, the cassette can immediately retry the connection as many times as you want. The number of immediate connection retries is specified by the `Immediate Retries` parameter (this parameter is called `$MAXRETRIES` on the ModifyCassette API command). Once a socket connection is established to the CashRegister for a given WebSphere Commerce Payments command, immediate retries are disabled for the remainder of that command. All retry logic from that point on will occur as described below. If the initial socket connection is not established and the maximum number of immediate retries is reached, the cassette also invokes the logic described below, but immediate retries remain enabled.
- Once a socket connection is established to the CashRegister, the cassette sends an appropriate request message to the CyberCash CashRegister and then waits for a predetermined period of time for a response. This period of time is specified by the `ReadTimeout` value in the cassette settings. If the cassette receives a response message indicating that the request is complete before the `ReadTimeout` expires, the message exchange is considered complete. Otherwise, this is considered one communication `Attempt`, and the cassette enters ″delayed retry″ logic. Specifically, delayed retries will be used if any of the following conditions occur:
  - The `ReadTimeout` period expires before a response is received.
  - A communication error occurs.
  - The CashRegister responds with a ″pending″ indication.
  - As described above, the initial socket connection has not yet been established and the maximum number of immediate connection retries is reached.
- Once a communication attempt is complete, the cassette will queue the pending request message and then wait for a predetermined amount of time called the `AttemptInterval`. The `AttemptInterval` is also specified through the cassette settings. Once the attempt interval expires, the request is removed from the internal queue and is either retried or recovered. This process is repeated until the request is completed or until the maximum number of communication attempts is reached. The maximum number of communication attempts is specified by the `MaxAttempts` value in the cassette settings.

Be very careful when choosing the `ReadTimeout` value for the CyberCash Cassette object. You should keep the following in mind when choosing the `ReadTimeout` value:

- Setting the value too low will cause unnecessary timeouts and command retries. Some might even time out on each retry attempt, preventing the required exchange of messages with the CashRegister.

- Setting the value too high may cause other WebSphere Commerce Payments commands to be delayed for an excessive amount of time because the internal locks used to ensure data integrity and thread safety are held for the `ReadTimeout` period.

The default `ReadTimeout` setting of 60 seconds should serve the needs of most installations. If the default does not meet your needs, you should tailor this and other retry settings as necessary, keeping the above points in mind.

Because of the complexity involved in settling batches under the `terminalCapture` processing model, the Cassette for CyberCash enforces a minimum `ReadTimeout` value of 60 seconds and a minimum `MaxAttempts` value of 3 when performing the `BatchClose` command. If either of these settings are configured to lesser values, the cassette uses the minimum values. Otherwise, the cassette uses the configured values.

CyberCash CashRegister and Processor-specific limitations

It is important to understand that the Cassette for CyberCash does not attempt to mirror the various syntactical and operational restrictions that the CyberCash CashRegister or the payment card processors impose for transaction parameters. Therefore, in some cases, a value that is allowed by the Cassette for CyberCash as being valid may be rejected by the CashRegister or the processor. In these cases, the error will be reflected as a financial failure with the appropriate CashRegister error information being available through a query of the affected WebSphere Commerce Payments Payment, Credit or Batch object.

Batch processing support

A batch is a collection of payments and credits that are processed as a unit by an acquiring financial institution. Batches are associated with a given merchant and account. After some period of time, or after some predetermined number of payment and credit transactions have been added to a batch, the batch must be settled in order for the financial institution to actually execute the batched transactions (that is, to actually transfer funds from the cardholder's account to the merchants or vice versa). There are two types of batches, both of which are supported by the Cassette for CyberCash:

- *Merchant-controlled batches*, whose contents and settlement policies are controlled by the merchant. This type of batch is used under the TerminalCapture processing model.
- *Acquirer-controlled batches*, where the financial institution decides which batch each transaction belongs to, and when each batch should be settled. This type of batch is used under both types of HostCapture processing models.

For acquirer-controlled batches, *the grouping of transactions into the WebSphere Commerce Payments batch has no direct correlation to the grouping used by the financial institution*. In fact, it is likely that the number of batches that exist in the WebSphere Commerce Payments and at the acquirer do not even match. It is important to note that the representation of acquirer-controlled batches by the WebSphere Commerce Payments is purely an administrative convenience to maintaining payment and credit totals for the merchant.

For merchant-controlled batches, the role of the WebSphere Commerce Payments is very different. In this case, the Cassette for CyberCash determines the contents of each batch and directs the CyberCash CashRegister when to actually settle the

batch. During the process of batch settlement, the cassette ensures that the batch status and totals maintained in the local database and at the CyberCash CashRegister agree.

The Cassette for CyberCash does not allow the merchant software to explicitly request that a batch be opened (that is, the BatchOpen command is not supported by the cassette). Rather, the cassette opens all batches implicitly on an as-needed basis for all processing models. By using this type of scheme, the cassette ensures that there is always one open batch per account.

Settlement (Batch closure), on the other hand, must be explicitly requested by the merchant software using the BatchClose command or the Settle function of the user interface. The Cassette for CyberCash will never close a batch on its own. This means the merchant software must close batches on a periodic basis (again, based on either time or transaction volume) to ensure that batches do not grow too large. Settlement of large batches can take a very long time to complete. When a merchant-controlled batch is successfully closed, the payments and credits associated with the batch move from deposited or refunded state (respectively) to closed state.

Closure of acquirer-controlled batches does not typically affect the state of the batched payments or credits.

When batches belonging to terminalCapture accounts are closed, the batch status field will be set to indicate the final status of the batch settlement. Two values are possible:

Balanced
The batch settled successfully and the batch totals at the financial institution match those at the CashRegister.

Out of Balance
Indicates that either:

- The batch settled successfully, but the batch totals at the financial institution do not match those at the CashRegister.
- Some other event occurred that prevents the Cassette for CyberCash from being able to determine the final balance status of the batch. Very often, this indicates an error from the CashRegister or communication between the cassette and the CashRegister were interrupted.

If your batch settles out of balance, you must follow up through the CyberCash CashRegister's Payment Administration User Interface to determine the final status of the batch.

Note: Since each account is associated with one currency, every batch opened under that account will be limited to the currency of the particular account.

Cassette representation of CyberCash processing models

It is important to understand the host capture and terminal capture processing models supported by the CyberCash CashRegister Service, as described in "Transaction processing models" on page 1. Table 2 on page 13 describes the terms that the Cassette for CyberCash uses to refer to these processing models. *AcquirerProfile* is the value specified in the Account configuration to tell the cassette which processing model is in use. *WebSphere Commerce Payments Terminology* is the term used throughout the remainder of this book, as well as in the WebSphere

Commerce Payments user interface, to refer to these processing models. These terms are already familiar to WebSphere Commerce Payments users.

Table 2. CyberCash Processing Models

Processing Model	Acquirer Profile	WebSphere Commerce Payments Terminology
HostCapture/AuthCapture	1	Acquirer-controlled (AutoDeposit only)
HostCapture/PostAuth	2	Acquirer-controlled
Terminal Capture	3	Merchant-controlled

Support for the WebSphere Commerce Payments command set

For a complete listing of Cassette for CyberCash extensions to the WebSphere Commerce Payments command set, see Chapter 6, “Using WebSphere Commerce Payments Commands with the Cassette for CyberCash” on page 41. The Cassette for CyberCash supports every WebSphere Commerce Payments generic command except for the following:

- BatchOpen
- BatchPurge
- CassetteControl
- Create/Modify/DeleteSystemCassetteObject
- ModifyMerchantCassetteObject
- ReceivePayment

If any of these commands are received by the Cassette for CyberCash, the following primary and secondary return codes will be returned: PRC_COMMAND_NOT_SUPPORTED, RC_NONE.

The following section describes behaviors unique to the Cassette for CyberCash for the listed WebSphere Commerce Payments commands.

Note: References to the *AutoApprove* option means that APPROVEFLAG = 1 or 2. The *AutoDeposit* option means that DEPOSITFLAG = 1.

AcceptPayment

- For Acquirer-controlled (AutoDeposit Only) accounts, if you specify the AutoApprove option, you must also specify the AutoDeposit option. This is required because the processing model only allows authorization with immediate capture.
- For both types of Acquirer-controlled accounts, successful completion of this command with the AutoApprove and AutoDeposit options will cause the resulting Payment object to move into Closed state, since the actual settlement of the deposit is carried out by the acquirer on its own schedule.

When a new order whose pre-existent state is either ORDER_RESET or ORDER_REQUESTED is created via the ACCEPTPAYMENT API, the Order object is put in the ORDER_REFUNDABLE state. This allows the merchant to issue a refund on the order even when no payment exists.

Approve

- For Acquirer-controlled (AutoDeposit Only) accounts, you must specify the AutoDeposit option. This is required because the processing model only allows authorization with immediate capture.

- For both types of Acquirer-controlled accounts, successful completion of this command with the AutoDeposit option will cause the resulting Payment object to move into Closed state, since the actual settlement of the deposit is carried out by the acquirer on its own schedule.

ApproveReversal

- This command is not supported for Acquirer-controlled (AutoDeposit Only) accounts since successful approvals place the resulting Payment object into Closed state. Use the Refund command instead.
- This command does not always result in a reversal being sent to the financial institution. The actual behavior depends upon the support for such reversals by the CyberCash CashRegister and its gateways. In cases where no messages flow to the CashRegister, the reversal is recorded in the WebSphere Commerce Payments database and the authorization is left to expire at the processor. The expiration period varies per processor, but typically lasts several days.
- Both full and partial reversals are supported. If the Amount value is ″0,″ a full reversal is performed and the payment is put into the VOID state. If the Amount is non-″0,″ and this amount is not greater than the original order amount, then a partial reversal is done. The old payment is void and the new payment is sent to the CashRegister for authorization.

CloseOrder

The Delete option may be used only if every Batch containing one or more of the Payments or Credits has already been Closed.

DeleteBatch

A Batch may be deleted only if all the associated Orders with this particular Batch have already been deleted.

Deposit

- For Acquirer-controlled accounts, successful completion of this command places the Payment object into Closed state, since the actual settlement of the Deposit is carried out by the acquirer on its own schedule.
- This command is never allowed for Acquirer-controlled (AutoDeposit Only) accounts since approvals under such accounts require AutoDeposit. Because of this requirement, Payments under this processing model will never be left in Approved state, which is the only state that allows the Deposit command.

DepositReversal

- The DepositReversal command is never allowed for either type of Acquirer-controlled account. Since successful deposits under these processing models place the Payments into Closed state, the Payments will never be left in Deposited state, which is the only state that allows the DepositReversal command.
- Successful completion of this command under Merchant-controlled accounts places the Payment into Void state, in order to conform to the way the CashRegister handles reversals.

Refund

For both types of Acquirer-controlled accounts, successful completion of this command places the resulting Credit object into Closed state, since the actual settlement of the refund is carried out by the acquirer on its own schedule.

RefundReversal

The RefundReversal command is never allowed for either type of Acquirer-controlled account. Since successful refunds under these processing

models place the Credits into Closed state, the Credits will never be left in Refunded state, which is the only state that allows the RefundReversal command.

Cassette for CyberCash financial command summary

Table 3 summarizes the way the Cassette for CyberCash handles each of the WebSphere Commerce Payments financial commands (that is, the commands that carry out financial transactions) for each processing model. Specifically, for each financial command, the table shows:

- Which payment card function will be performed by the command (using terminology more common to the payment card industry)
- How the cassette processes the command:
  - ″Not supported by cassette″ means the cassette does not support that particular command. These commands will always receive return codes PRC_COMMAND_NOT_SUPPORTED, RC_NONE.
  - ″Handled by WebSphere Commerce Payments; no message sent″ means that the command is processed completely within the WebSphere Commerce Payments without communicating to the CashRegister.
  - ″Fails″ with added text indicates that the command is not supported under the processing model for the specified reason.
  - In any other case, the primary CyberCash CashRegister command (or commands) used to accomplish the function will be shown in italics.

Note: The appropriate CashRegister retry and query commands may be used in conjunction with any of the listed commands as necessary to update status or to retry a function.

Table 3. Cassette for CyberCash:. Summary of Payment API Commands

API Command	Payment Card Functions	Acquirer-controlled (AutoDeposit only)	Acquirer-controlled	Terminal Capture
AcceptPayment	No comparable function	Handled by WebSphere Commerce Payments; no message sent	Handled by WebSphere Commerce Payments; no message sent	Handled by WebSphere Commerce Payments; no message sent
AcceptPayment with AutoApprove	Authorize	Fails. AutoDeposit must also be specified	*mauthonly*	*mauthonly*
AcceptPayment with AutoApprove and AutoDeposit	Authorize and capture	*mauthcapture*	*mauthonly*, *postauth*	*mauthonly*, *postauth*
Approve	Authorize	Fails.AutoDeposit must also be specified	*mauthonly*	*mauthonly*
Approve with AutoDeposit	Authorize and capture	*mauthcapture*	*mauthonly*, *postauth*	*mauthonly*, *postauth*
ApproveReversal	Authorize reversal	Not valid; object states don't permit function	Handled by WebSphere Commerce Payments; no message sent if amount is 0; otherwise, approve is processed	Handled by WebSphere Commerce Payments; no message sent if amount is 0; otherwise, approve is processed

Table 3. Cassette for CyberCash: (continued). Summary of Payment API Commands

BatchClose	Close an existing batch	Handled by WebSphere Commerce Payments; no message sent	Handled by WebSphere Commerce Payments; no message sent	*batch-commit*
BatchOpen	Open a new batch	Not supported by cassette (Cassette opens batches internally as needed.)	Not supported by cassette (Cassette opens batches internally as needed.)	Not supported by cassette (Cassette opens batches internally as needed.)
BatchPurge	Purge an existing batch	Not supported by cassette	Not supported by cassette	Not supported by cassette
CancelOrder	No comparable function	Handled by WebSphere Commerce Payments; no message sent	Handled by WebSphere Commerce Payments; no message sent	Handled by WebSphere Commerce Payments; no message sent
CloseOrder	No comparable function	Handled by WebSphere Commerce Payments; no message sent	Handled by WebSphere Commerce Payments; no message sent	Handled by WebSphere Commerce Payments; no message sent
DeleteBatch	No comparable function	Handled by WebSphere Commerce Payments; no message sent	Handled by WebSphere Commerce Payments; no message sent	Handled by WebSphere Commerce Payments; no message sent
Deposit	Capture	Not valid; object states don't permit function	*postauth*	*postauth*
DepositReversal	Capture Reversal	Not valid; object states don't permit function	Not valid; object states don't permit function	*void marked*
ReceivePayment	No comparable function	Not supported by cassette	Not supported by cassette	Not supported by cassette
Refund	Credit	*return*	*return*	*return*
RefundReversal	Credit Reversal	Not valid; object states don't permit function	Not valid; object states don't permit function	*void marked*

Summary of state changes

The following table summarizes the state changes that Order, Payment, Credit and Batch objects undergo as a result of successful completion of each payment command for each processing model. Only those objects whose states actually change as a result of the given operation are shown. Any other existing object states remain unchanged.

API Command	**Acquirer-controlled (Auto-Deposit Only)**	**Acquirer-controlled**	**Merchant-controlled**
AcceptPayment	ORDER_REFUNDABLE	ORDER_REFUNDABLE	ORDER_REFUNDABLE
AcceptPayment with AutoApprove	Not allowed	ORDER_REFUNDABLE PAYMENT_APPROVED	ORDER_REFUNDABLE PAYMENT_APPROVED

AcceptPayment with AutoApprove and AutoDeposit	ORDER_REFUNDABLE PAYMENT_CLOSED	ORDER_REFUNDABLE PAYMENT_CLOSED	ORDER_REFUNDABLE PAYMENT_DEPOSITED
Approve	Not allowed	PAYMENT_APPROVED	PAYMENT_APPROVED
Approve with AutoDeposit	ORDER_REFUNDABLE PAYMENT_CLOSED	ORDER_REFUNDABLE PAYMENT_CLOSED	PAYMENT_DEPOSITED
ApproveReversal	Not allowed	PAYMENT_VOID	PAYMENT_VOID
CancelOrder	ORDER_CANCELED	ORDER_CANCELED	ORDER_CANCELED
CloseOrder	ORDER_CLOSED	ORDER_CLOSED	ORDER_CLOSED
Deposit	Not allowed	ORDER_REFUNDABLE PAYMENT_CLOSED	PAYMENT_DEPOSITED
DepositReversal	Not allowed	Not allowed	PAYMENT_VOID
Refund	CREDIT_CLOSED	CREDIT_CLOSED	CREDIT_REFUNDED
RefundReversal	Not allowed	Not allowed	CREDIT_VOID
BatchClose	BATCH_CLOSED	BATCH_CLOSED	BATCH_CLOSED PAYMENT_CLOSED CREDIT_CLOSED ORDER_REFUNDABLE
DeleteBatch	Deletes the batch	Deletes the batch	Deletes the batch

Protecting sensitive data

As an option, you can prevent sensitive financial data such as credit card numbers and expiry dates from being returned in query results when users enter query commands. A Payment Servlet parameter called `wpm.MinSensitiveAccessRole` can be specified to define the minimum access role a user must have to view sensitive data returned in query command results. (Refer to the *WebSphere Commerce Payments Administrator's Guide* for information on how to set this and other Payment Servlet initialization parameters.)

When a user enters a query through a query command, WebSphere Commerce Payments checks the user's role against the minimum role specified for the `wpm.MinSensitiveAccessRole` parameter and determines whether sensitive data should be returned in full view or masked out. The following table lists the data elements that are considered sensitive by the Cassette for CyberCash:

Table 4. . Sensitive data processed by Cassette for CyberCash

Data	**How data is protected**
$PAN	Cardholder's card number. All but the last 4 digits of the card number are masked with asterisks.
$EXPIRY	Card expiration date. The entire value is masked with asterisks.

Supported minimum sensitive access role values are: clerk, supervisor, merchant administrator, payments administrator, or none. If the `wpm.MinSensitiveAccessRole` parameter is not specified, an access role of clerk is assumed, which allows all users to see sensitive data. If the user's role matches or exceeds the role value, the actual values are displayed for the sensitive data.

For more information about query commands, refer to the *IBM WebSphere Commerce Payments Programmer's Guide and Reference*.

Chapter 3. Installing the Cassette for CyberCash

The WebSphere Commerce Payments must be installed before the Cassette for CyberCash can be installed. The WebSphere Commerce Payments installation will ensure that all prerequisite products are available (this is true for all platforms except iSeries). For detailed information on the WebSphere Commerce Payments Framework, including hardware and software prerequisites, refer to the *IBM WebSphere Commerce Payments Installation Guide*, *Version 3.1*.

Before Installing Cassette for CyberCash

- Read the latest README file, *readme.CyberCash.html*, accessed through documentation links on the IBM WebSphere Commerce Payments Web site: **http://www.ibm.com/software/commerce/payment/support/index.html** and on the Cassette for CyberCash CD-ROM.
- WebSphere Commerce Payments should *not* be running at cassette installation. WebSphere Application Server *should be* running at cassette installation.

  Note: A prior version of the cassette cannot be installed on top of the WebSphere Commerce Payments Version 3.1.3 Framework. If you currently use a prior version of the Cassette for CyberCash, you must install the WebSphere Commerce Payments Version 3.1.3 Cassette for CyberCash software for your cassette data to be migrated and compatible with the WebSphere Commerce PaymentsVersion 3.1.3 Framework.
- Starting with Payment Manager Version 3.1.1, changes were made to WebSphere Commerce Payments directory structure and configuration of WebSphere Application Server relative to that of earlier releases of Payment Manager. For more information about the installed directory structure, see: "WebSphere Commerce Payments directory structure" on page 20.

Note: iSeries does not require that WebSphere Commerce Payments or WebSphere Application Server be ended during installation.

Installing Cassette for CyberCash (Windows® and UNIX platforms)

This section describes the procedure for installing the Cassette for CyberCash on Windows NT®, Windows 2000, Solaris, AIX® and Linux®. Before installing the Cassette for CyberCash software, you should stop the WebSphere Commerce Payments Application Server from the WebSphere Application Server administrative console. This ensures that the configuration files for WebSphere Commerce Payments will be freed to enable the cassette installation program to update the files. (If you are installing more than one type of payment cassette, you must stop the WebSphere Commerce Payments Application Server before installing each cassette.)

- On AIX, Linux and Solaris, you must logon as **root** directly.
- On Windows NT and Windows 2000, you must logon as a user who is a member of the administrator group.

1. Once logged in, insert the insert the CD-ROM containing WebSphere Commerce Payments Cassette for CyberCash.
2. Select the directory for your platform:
   - For Windows NT or Windows 2000, go to the *nt* directory.
   - For Solaris, go to the *solaris* directory.

- For Linux, go to the *linux* directory.
- For AIX, go to the *aix* directory.

3. Enter **InstallCyberCashCassette** to start the installation. Enter the information requested on the installation screens.
4. The *Cassette for CyberCash Readme* screen indicates that the configuration of the Cassette for CyberCash has successfully completed and lets you display the README, if desired.

WebSphere Commerce Payments directory structure

Starting with Payment Manager Version 3.1.1, some changes were made to the WebSphere Commerce Payments directory structure and configuration of WebSphere Application Server relative to that of Payment Manager Version 3.1.0 and earlier releases. Under WebSphere Application Server Version 4.0, WebSphere Commerce Payments now makes use of Web archive (WAR) and enterprise archive (EAR) files. After the WebSphere Commerce Payments framework is installed, a `WPMApplication.ear` file representing the WebSphere Commerce Payments application is found in the *<Payments_installdir>* deployable subdirectory. The EAR file has a sudirectory structure containing a `Payments.war` file, Web files, and other files used to configure WebSphere Application Server . Some files that you may have seen in the WebSphere Commerce Payments installation directory prior to version 3.1.1 are now moved into the EAR file structure and deployed to a WebSphere Application Server directory. In WebSphere Application Server , the WebSphere Commerce Payments EAR file becomes a subdirectory under the `installedApps` subdirectory (for example, on Windows: `<WAS_DIR>\installedApps\IBM_Payments.ear`).

The following Cassette for CyberCash files are moved into the EAR file (or EAR directory for iSeries):

```
<Payments_installdir>\eTillCyberCashClasses.zip
<Payments_installdir>\web\*
```

Under this revised directory structure, if you need to make changes to the WebSphere Commerce Payments (Payment Servlet) initialization parameters, you should refer to the *IBM WebSphere Commerce Payments Administrator's Guide* for instructions on using the WebSphere Application Server administrative console.

Installing Cassette for CyberCash (iSeries)

This section describes the procedure for installing the Cassette for CyberCash for iSeries and adding the cassette to a WebSphere Commerce Payments instance.

Installing the cassette

- Use the Restore License Program **(RSTLICPGM)** CL command to install the Cassette for CyberCash option of the IBM WebSphere Commerce Payments for iSeries product.
- Specify the WebSphere Commerce Payments product number, option 2 for the Cassette for CyberCash and the device from which the product is to be installed. For example:

  ```
  RSTLICPGM LICPGM(5733PY3) DEV(OPT01) OPTION(2).
  ```

Adding a Cassette for CyberCash to a WebSphere Commerce Payments Instance

After installing the Cassette for CyberCash, you need to add the CyberCash cassette to a WebSphere Commerce Payments instance. This process enables the cassette to be used by that instance. Be sure none of the following are running when you begin adding the Cassette for CyberCash to a WebSphere Commerce Payments instance:

- The WebSphere Commerce Payments instance
- The HTTP Server that processes payment requests for the WebSphere Commerce Payments instance
- The Websphere Application Server that processes payment requests for the WebSphere Commerce Payments instance

To add the Cassette for CyberCash to a WebSphere Commerce Payments instance:

- Access the iSeries Tasks page at **http***://system-name*:**2001** where the system-name is the TCP/IP host name of the iSeries system.
- Select the **WebSphere Commerce Payments** icon.
- Select the WebSphere Commerce Payments instance from the drop-down menu.
- Select the **Work Cassettes** menu.
- Select **CyberCash** from the cassette list and press the **Add** button to add the cassette.

  Note: Alternately, you can use the **(ADDPYMCSS)** CL command.

When the process for adding the cassette to a WebSphere Commerce Payments instance has completed, the cassette-specific tables are added to the WebSphere Commerce Payments instance database collection. You are now ready to start the WebSphere Commerce Payments and configure the CyberCash cassette.

Uninstalling Cassette for CyberCash

To remove Cassette for CyberCash, use the process for your operating system:

- "Uninstalling Cassette for CyberCash on Windows NT"
- "Uninstalling Cassette for CyberCash on Windows 2000" on page 22
- "Uninstalling Cassette for CyberCash on Solaris" on page 22
- "Uninstalling Cassette for CyberCash on AIX" on page 22
- "Uninstalling Cassette for CyberCash on Linux" on page 23
- "Uninstalling Cassette for CyberCash for iSeries" on page 24

Uninstalling Cassette for CyberCash on Windows NT

On Windows NT, you can remove the Cassette for CyberCash using the following steps:

1. Use the WebSphere Application Server administrative console to stop the WebSphere Commerce Payments Application Server.
2. Go to the Windows NT Control Panel.
3. Click the **Add/Remove Programs** icon.
4. Select the **IBM WebSphere WebSphere Commerce Payments Cassette for CyberCash**.

5. Click **Add/Remove**.

   Note: This process removes all of the Cassette for CyberCash tables that were installed on your system, including those containing financial transaction data.

Uninstalling Cassette for CyberCash on Windows 2000

On Windows 2000, you can remove the Cassette for CyberCash using the following steps:

1. Use the WebSphere Application Server administrative console to stop the WebSphere Commerce Payments Application Server.
2. Go to the Windows 2000 Control Panel.
3. Click the **Add/Remove Programs** icon.
4. Select the **IBM WebSphere Commerce Payments Cassette for CyberCash**.
5. Click **Add/Remove**.

   Note: This process removes all of the Cassette for CyberCash tables that were installed on your system, including those containing financial transaction data.

Uninstalling Cassette for CyberCash on Solaris

On Solaris, you can remove the Cassette for CyberCash using the following steps:

1. Set your display and xhost. From a command prompt, enter:

   ```
   export DISPLAY <machine_name:0.0>
   xhost + <machine_name>
   ```

2. Use the WebSphere Application Server administrative console to stop the WebSphere Commerce Payments Application Server.
3. Go to a Solaris console window. Be sure you are logged on as user '**root**'.
4. Enter **cd /var/sadm/pkg**.
5. Enter **pkginfo itj***.
6. Record the ″itj″ numbers associated with the IBM WebSphere Commerce Payments 3.1 Cassette for CyberCash.
7. Enter **pkgrm itj<Cassette_for_CyberCash_itjnumber1>**.
8. Database files that were created after installation must be removed manually before a successful reinstall of the WebSphere Commerce Payments can be achieved.
9. Now remove the other package. Enter:

   ```
   pkgrm itj<Cassette_for_CyberCash_itjnumber2>
   ```

   Note: This process removes all of the Cassette for CyberCash tables that were installed on your system, including those containing financial transaction data.

Uninstalling Cassette for CyberCash on AIX

On AIX, you can remove the Cassette for CyberCash using SMIT, following these steps:

1. Use the WebSphere Application Server administrative console to stop the WebSphere Commerce Payments Application Server.
2. Enter **smit** to display the SMIT System Management menu (SMIT main menu).
3. Navigate to the dialog panel that allows you to remove software products.

4. Display your installed software.
5. Select all components that begin with **WebSphere.Commerce.Payments.Cassette.for.CyberCash** and click **OK**.
6. Make sure the Preview Only field is set to **No**. Select **OK** to remove the Cassette for CyberCash.

   Note: This process removes all of the Cassette for CyberCash tables that were installed on your system, including those containing financial transaction data.

Uninstalling Cassette for CyberCash on Linux

On Linux, you can remove the Cassette for CyberCash using the following steps:

1. Set your display and xhost. From a command prompt, enter:

   ```
   export DISPLAY <machine_name:0.0>
   xhost + <machine_name>
   ```

2. Use the WebSphere Application Server administrative console to stop the WebSphere Commerce Payments Application Server.
3. Logon as **root**.
4. Change to the WebSphere Commerce Payments install directory. By default, WebSphere Commerce Payments is installed to `/opt/Payments`.
5. From the command prompt, run this script:

   ```
   ./UninstallCyberCash.sh
   ```

   Note: This process removes all of the Cassette for CyberCash tables that were installed on your system, including those containing financial transaction data.

Removing Cassette for CyberCash from a WebSphere Commerce Payments for iSeries

Removing the Cassette for CyberCash from a WebSphere Commerce Payments instance will remove all CyberCash configuration and transaction data from that instance. Be sure none of the following are running when you begin removing the Cassette for CyberCash from a WebSphere Commerce Payments instance:

- The WebSphere Commerce Payments instance
- The HTTP Server that processes payment requests for the WebSphere Commerce Payments instance
- The Websphere Application Server that processes payment requests for the WebSphere Commerce Payments instance

To remove the Cassette for CyberCash to a WebSphere Commerce Payments instance:

- Access the iSeries Tasks page at **http***://system-name***:2001** where the system-name is the TCP/IP host name of the iSeries system.
- Select the **WebSphere Commerce Payments** icon.
- Select the WebSphere Commerce Payments instance from the drop-down menu.
- Select the **Work Cassettes** menu.
- Select **CyberCash** from the cassette list and press the **Remove** button to add the cassette.

  Note: Alternately, you can use the **(RMVPYMCSS)** CL command.

Uninstalling Cassette for CyberCash for iSeries

To uninstall the Cassette for CyberCash option from the system, use the Delete License Program **(DLTLICPGM)** CL command to delete option 2 from the WebSphere Commerce Payments product. For example:

```
DLTLICPGM LICPGM(5733PY3) OPTION(2).
```

Note: All WebSphere Commerce Payments instances must be ended before uninstalling the Cassette for CyberCash.

Chapter 4. Getting Started

Use the information here to configure the Cassette for CyberCash. At this point, you should have completed the following:

- For Windows and UNIX platforms:
  - Installed and configured the WebSphere Commerce Payments Framework.
  - Installed the Cassette for CyberCash and registered as a CyberCash merchant. (See **http://www.cybercash.com** to register as a CyberCash merchant.)
  - Started the WebSphere Application Server and Web Server (if necessary).
  - Defined a WebSphere Commerce Payments user with administrative authority.
  - Created a merchant and Merchant Administrator for the merchant.
- For iSeries:
  - Installed and configured the WebSphere Commerce Payments Framework.
  - Created a WebSphere Commerce Payments instance.
  - Added the Cassette for CyberCash to the WebSphere Commerce Payments instance and registered as a CyberCash merchant. (See **http://www.cybercash.com** to register as a CyberCash merchant.)
  - Started the Instance.
  - Started the WebSphere Application Server and Web Server (if necessary).
  - Defined a WebSphere Commerce Payments user with administrative authority.
  - Created a merchant and Merchant Administrator for the merchant.
  - Assigned a password to the qpymadm (WebSphere Commerce Payments Administrator) user profile.

To configure a cassette, you must logon to WebSphere Commerce Payments as a Merchant administrator. For information on performing these tasks, see the *IBM WebSphere Commerce Payments Administrator's Guide for Multiplatforms, Version 3.1*.

Note: Whenever a WebSphere Commerce Payments URL is listed in this manual, this implies that the WebSphere Commerce Payments uses a default URL corresponding to one using default HTTP port number 80. If your configuration is such that the WebSphere Commerce Payments is running on a port other than default port number 80, you may need to specify a port number in the URL.

Cassette for CyberCash tutorial

After installing the Cassette for CyberCash, you must configure the cassette before you can process customer transactions. This tutorial will show you how to configure the Cassette for CyberCash. For detailed information on administration, configuration, and payment functions, see the online Help for the WebSphere Commerce Payments user interface.

Using the tutorial software as a model, this chapter demonstrates everything you must do to achieve a fully functioning Cassette for CyberCash. This information walks you through fictitious scenarios that simulate real-world functions. And while you need not complete the entire walk-through, it is important that you complete these tasks to become familiar with the common Cassette for CyberCash tasks:

1. Create an account.

2. Create a brand.

In addition to the required configuration tasks above, we will walk through common payment processing tasks.

Before starting this tutorial

There are a number of configuration steps that require information from your CyberCash merchant account or your financial institution. These items are:

- Your CyberCash ID (CCID), which was provided upon completing the initial phase of the CyberCash IMR process.
- Your CashRegister MerchantKey, which was provided at the same time as the CCID.
- The transaction processing model you will use.
- The currency with which your transactions will be processed.
- The list of payment card brands that you will accept.

Note: For this tutorial, a special test account that uses the Acquirer-controlled (AutoDeposit Only) processing model will be used. Because of the rules surrounding this processing model, only a subset of the functions supported by the Cassette for CyberCash will be used. However, we will also discuss other functions that are available under the other processing models, where appropriate.

Starting the WebSphere Commerce Payments User Interface

Our first task is enabling a merchant to use the Cassette for CyberCash. This must be done by a user with Payments Administrator access.

To start the WebSphere Commerce Payments user interface:

1. Point your browser to **http://<hostname>/webapp/PaymentManager/**, where <hostname> is the machine where the WebSphere Commerce Payments is installed.
2. On the WebSphere Commerce Payments Logon window, type the Payments Administrator's user ID and password and click **Logon**.

Configuring the Cassette for SOCKS

During this tutorial, your WebSphere Commerce Payments will attempt to send messages through the Internet to the CyberCash CashRegister Service. If your computer is behind a firewall and must use a SOCKS server to access sites outside of your internal network, then you may have to perform certain steps to enable the Cassette for CyberCash for SOCKS.

Note: If you already have access to sites on the Internet, then you may skip this step:

1. Click on **Cassettes** in the navigation frame.
2. Click on the **CyberCash** cassette icon.
3. At the next window, you will see several entry fields that allow you to tailor the way the Cassette for CyberCash communicates with the CashRegister Service. Two of these entry fields are related to SOCKS configuration. Enter the following:

SOCKS Host	The hostname of your installation SOCKS server.

SOCKS Port	The TCP port of your installation SOCKS server.

4. Click on **Update** to update your cassette configuration.

Configuring the Cassette for SOCKS (iSeries)

During this tutorial, your WebSphere Commerce Payments will attempt to send messages through the internet to the CyberCash CashRegister Service. If your computer is behind a firewall and must use a SOCKS server to access sites outside of your internal network, then you may have to perform certain steps to enable the Cassette for CyberCash for SOCKS.

If you have already configured the iSeries TCP/IP support to use a SOCKS server, the iSeries will automatically route connections to the SOCKS server. Thus, you will not have to change the configuration for the CyberCash Cassette.

For more information about configuring the iSeries TCP/IP SOCKS support, see OS/400 TCP/IP Configuration and Reference, Version 4.0

You can access this manual from the iSeries on-line library at:
http://publib.boulder.ibm.com/pubs/html/as400/infocenter.htm.

1. Select **TCP/IP**.
2. Select **Getting Started**.

If you choose not to configure SOCKS within iSeries TCP/IP support, then perform the following steps to enable the Cassette for CyberCash for SOCKS.

Note: If you already have access to sites on the Internet, then you may skip this step:

1. Click on **Cassettes** in the navigation frame.
2. Click on the **CyberCash** cassette icon.
3. At the next window, you will see several entry fields that allow you to tailor the way the Cassette for CyberCash communicates with the CashRegister Service. Two of these entry fields are related to SOCKS configuration. Enter the following:

SOCKS Host	The hostname of your installation SOCKS server.
SOCKS Port	The TCP port of your installation SOCKS server.

4. Click on **Update** to update your cassette configuration.

Selecting a WebSphere Commerce Payments merchant and authorizing a cassette

If you haven't already created a merchant, you must do that first and authorize that merchant to use a payment cassette. To create a merchant, you must log into the WebSphere Commerce Payments as an administrator:

1. From the navigation frame, click **Merchant Settings** under the Administration section.
2. From the Merchant Settings window, select the Test store merchant created during the WebSphere Commerce Payments tutorial (See the *IBM WebSphere*

Commerce Payments Administrator's Guide for Multiplatforms, for details on the WebSphere Commerce Payments tutorial), or create a new merchant with merchant number 123456789.

Note: If there are more than 500 merchants in the WebSphere Commerce Payments database, you are prompted to search for a specific merchant or merchants when you access the Merchant Settings window. If you see this prompt, enter 123456789.

3. At the next window, you will be prompted to authorize use of the Cassette for CyberCash:

Authorized cassettes	Check the box next to *CyberCash*. Checking this box authorizes the merchant to use the Cassette for CyberCash.

4. When you have entered the requested information, click **Update (or CreateMerchant)** to save the merchant configuration.
5. You will also have to give the user ID Merchant Administrator authority for this merchant. For instructions on assigning roles, see the *IBM WebSphere Commerce Payments Administrator's Guide*.

Logging in as the merchant administrator

To logoff and login again:

1. Click **Logoff admin** on the navigation frame of the WebSphere Commerce Payments user interface, and you will return to the main WebSphere Commerce Payments Login window.
2. From the main WebSphere Commerce Payments Login window, type the user ID (with Merchant Administrator authority) and the password and click **OK**.

You are now logged in to the WebSphere Commerce Payments user interface with Merchant administrator authority for the Test Store merchant. For the remainder of the tutorial, you will act as the Merchant administrator. Notice that your view of the WebSphere Commerce Payments user interface is now limited to merchant administration functions, whereas as the Payments administrator, you had a global view of both *merchant* and *WebSphere Commerce Payments* administration.

Creating an account

So far, you have enabled one merchant, the Test Store, to use the Cassette for CyberCash. Now, you need to establish an *account* for the Cassette for CyberCash.

An account is a relationship between the merchant and the financial institution which processes transactions for that merchant. There can be multiple accounts for each payment cassette. But for the purposes of this tutorial, you will create one account for the Cassette for CyberCash.

To create an account:

1. Click **Merchant Settings** on the navigation frame of the WebSphere Commerce Payments user interface.
2. From the Merchant Settings window, click the **Cassette for CyberCash** icon in the Test Store.
3. From the Cassette for CyberCash window, click **Accounts**.
4. Click **Add an Account** on the Accounts window.

5. At the next window, you will be prompted to enter the following information (note that the italicized text *must* be entered in these fields for the tutorial):

Account name	Enter *CyberCash Account*. This is the name that you assign to the account. Its only function is to provide display information in the user interface.
Account number	Enter *2*. This is a number that you (that is, the hosting service provider or the merchant administrator) assign to uniquely identify this account in all transaction data. Note that if account number 2 is already defined for the Test Store merchant, then you may choose another number. We will assume for the remainder of this tutorial, however, that you are using account number 2.
Financial Institution name	Enter *CyberCash Bank*. This is the name of the financial institution with which you hold this account. Its only function is to provide display information in the user interface.
CyberCash Merchant ID	Enter *$testCCID$*. For your own CyberCash merchant account, you would enter the CCID value that you received upon completing the CyberCash merchant registration (IMR) process.
Merchant Key	Enter *$testCCKey$*. For your own CyberCash merchant account, you would enter the merchant key value that you received upon completing the CyberCash merchant registration (IMR) process.
Currency	Use the default value (US Dollars). For your own CyberCash merchant account, you would select the currency that you and your financial institution agreed to use.
Acquirer Profile	Use the default value (Acquirer-controlled (AutoDeposit Only)). For your own CyberCash merchant account, you would select this value according to the processing model that you and your financial institution agreed to use: • For HostCapture/AuthCapture, select **Acquirer-controlled (AutoDeposit Only)**. • For HostCapture/PostAuth, select **Acquirer-controlled**. • For TerminalCapture, select **Merchant-controlled**.

6. Click **Create account** to create the new account.

Creating a brand

Now that you have created the account for the Cassette for CyberCash, you need to create a *brand* for the CyberCash Account. A brand is a credit card type, such as VISA or MasterCard. The brand, or brands, that you define for an account is based upon the terms of your account with your acquiring, financial institution. This tutorial uses the fictitious brand ″CyberBrand.″

To create a brand:

1. Click **Merchant Settings** on the navigation frame of the WebSphere Commerce Payments user interface.
2. From the Merchant Settings window, click the **Cassette for CyberCash** icon in the Test Store window.
3. From the Cassette for CyberCash window, click **Accounts**.
4. Click **CyberCash Account** on the Accounts window.
5. From the Test Account window, click **Brands**.

6. Click **Add a brand** on the Brands window. (note that brand names are case sensitive)
7. At the next window, you will be prompted to enter the following information (note that the italicized text *must* be entered in these fields for the tutorial):
8. Enter *CyberBrand* for the brand name.
9. Click **Create brand** to create a brand for the CyberCash Account.

Managing payment processing

As the Merchant administrator, you have global *merchant* authority, which means that you can perform:

- Merchant-specific administration functions
- All payment processing functions

In a real business scenario, you may choose to delegate payment processing tasks to other merchant-defined users who possess limited payment processing authorities (such as, supervisor and clerk). In this tutorial, you, as the Merchant administrator, will perform these tasks.

Having completed all of the WebSphere Commerce Payments and Merchant administration tasks necessary to begin payment processing, you are now ready to start:

- Approving orders
- Depositing payments
- Settling batches
- Issuing credits
- Viewing daily batch totals

For the purposes of this tutorial, you will use the Sample Checkout to create three orders for use in payment processing.

Creating orders using the Sample Checkout

A real business environment features a customer who creates orders using a merchant's Internet storefront and a merchant who processes payments for those orders using the WebSphere Commerce Payments. In order for you to walk through the WebSphere Commerce Payments payment processing functions, you need to create orders that require payment processing. To simulate a merchant's Internet storefront and facilitate order creation, the Cassette for CyberCash supplies a Sample Checkout. To access the Cassette for CyberCash Sample Checkout and create orders:

1. Point your browser to **http://<hostname>/webapp/PaymentManager/SampleCheckout/**, where **<hostname>** is the machine where the WebSphere Commerce Payments is installed.
2. At the Sample Checkout window that appears, you will be prompted to enter the following (note that the italicized text *must* be entered in these fields for the tutorial):

Merchant number	Enter any number to represent a Merchant number.
Order number	Enter any number to represent an Order number.
Amount	Enter any amount to represent the total numeric amount of the order.

Currency	Enter *US dollar*. The currency used to place this order.
Payment method	Choose *CyberCash* as the payment method.
Credit card number	Enter *4111111111111111*.
Expiry Date	Highlight the expiration month and year for your CyberBrand credit card. **Note:** You can choose any future month and year combination for this tutorial.
Brand	Choose *CyberBrand* as the brand (that is, credit card) that you are using.
Street address	Enter the street address of the location of the cardholder.
State or Province	Enter the name or abbreviation of the state of provice of the location of the cardholder.
Cardholder's zip/postal code	Enter the zip/postal code of the cardholder.
Country	Select the country of the location of the cardholder.
Note: When the CyberCash payment method is selected, additional fields are displayed to accept credit card and cardholder information (such as the cardholder address information commonly used in North America as AVS data, which is shown in this table). The Sample Checkout application can be modified to display other countries in the drop-down list for the Country field. (To do this, modify the SampleCheckoutCyberCash.properties file for a given locale. More information about customizing properties files is provided in the *IBM WebSphere Commerce Payments Administrator's Guide*.)	

3. Click **Buy**.

Repeat these steps twice (each time with a different Order number) so that you have three orders for which to process payments.

Approving orders with the Sale function

Since the account being used for this tutorial is defined with the Acquirer-controlled (AutoDeposit Only) processing model, all of our payments must be approved and deposited in a single operation. This is what the ″Sale″ function does, so we will use it to take the next step in our payment processing. After completing this tutorial, we will briefly discuss approval without deposit for the other processing models.

Once you have created three orders using the Sample Checkout, return to the browser window, where the WebSphere Commerce Payments user interface is displayed.

Note: If you used the same browser window to access the Sample Checkout, you will need to point your browser once again to the WebSphere Commerce Payments URL (that is, **`http://<<hostname>>/webapp/PaymentManager/`**) and login as the merchant administrator.

Follow these steps to approve and deposit an order:

1. From the navigation frame, click **Approve** under the Payment Processing section.
2. From the Approve window, check the box next to the order that you want to approve and deposit (select only one order for this exercise) and click **Sale Selected**.
3. At the Approve Results window, you will see the status of your sale request. When processing is complete, success or failure status will appear next to each order submitted for sale.
4. When your sale is complete, click **Return to the Approve Screen**.

Two orders are still awaiting sale. You could have approved them all at once (for their full amounts), by clicking **Sale All** from the Approve window. Instead, you will work with each order individually to better demonstrate the many facets of the Approve function.

Approving orders from the Order window

In this section, you will approve and deposit an order from the Order window (rather than from the Approve window), but you will approve only *part* of the total order amount.

1. From the Approve window, click the **Order number** for one of the remaining orders awaiting approval.
2. From the Order window, you can view order details. Click **Sale** to approve and deposit this order.
3. The Order Approve window, displays the following information:

Currency	The type of currency used to place this order. This is a read-only field.
Order Amount	The total amount of the order expressed in the currency used to place the order. This is a read-only field.
Approved Amount	This field displays zeros since no amount of the order has yet been approved. This is a read-only field.
Deposited Amount	This field displays zeros since no amount has yet been approved or deposited. This is a read-only field.
Sale Amount	This is an entry field that currently contains the total amount of the order.

   Change the sale amount to **3.00** and click **Sale** to approve and deposit this order.

When sale processing is complete, the Order window will display again, with some indication of the success or failure of the sale. You will notice that approval and deposit amounts in the Order window details have been updated to reflect the $3.00 sale that you have just completed. In addition, you will notice at the bottom of the screen that a new payment is now listed under the Payments section. This is the payment that you just approved and deposited.

To view details of the payment, click on the payment number in the Payments section. On the Payment Detail screen, you will see the following information (all fields on this screen are read-only):

Merchant	The merchant name.
State	The current state of the Payment (Closed).
Currency	The currency used for this payment (US Dollars).
Approved Amount	The amount currently approved (3.00).
Deposited Amount	The amount currently deposited (3.00).
Batch Number	The WebSphere Commerce Payments batch into which this payment has been placed.
Account	The account under which this order is being processed (CyberCash Account).
Order URL	In a real merchant's online shopping system, this field might be filled in to point to a corresponding entry in the order entry database.
Time Created	The time that this payment was created.

Time Approved	The time that this payment was approved.
Reference Number	The processor's retrieval reference number received when the payment was approved. Not always present. Varies by processor.
Payment Type	The payment type used for this order (CyberCash).
blank line	Denotes the end of the WebSphere Commerce Payments generic attributes for this payment. All of the remaining attributes are unique to the Cassette for CyberCash.
CyberCash Transaction ID	The identifier by which this payment is known to the CyberCash CashRegister. You will need this identifier if you ever need to contact CyberCash about a problem with this payment.
AuthCode	The authorization code received from the acquirer when this payment was approved. Not always present. Varies by acquirer.
AVS code	The AVS code received from the acquirer when the payment was approved. These codes vary between acquirers. Refer to **www.cybercash.com** for a description of these codes.
Common AVS code	The WebSphere Commerce Payments AVS code that maps to the cassette-specific AVS code received from the acquirer. The common AVS codes can be used by any cassette that supports AVS and relieve the merchant software from having to know which cassette is being used in order to interpret the result.
MStatus	The status code from the last command sent to the CyberCash CashRegister for this payment.
Messages	May contain an ASCII-text informational or error message from the CashRegister, gateway, processor or acquirer. If present, this message was received in the response from the last command sent to the CyberCash CashRegister for this Payment. If the message is preceded with a message code in the form ″CRnnnn″ or ″GWnnnn″, refer to the *CyberCash Error Messages Guide* for a detailed explanation.

Approving multiple orders at one time

Once you have finished viewing the Payment details, return to the Approve window by clicking on **Approve** in the navigation frame. Since you only approved and deposited a portion of the order in the previous step, there are still two order entries in this window; the one that has been partially approved and the one that is still awaiting approval. In this exercise, you will approve and deposit the complete unapproved purchase amount for each of these in one operation. To do this:

1. From the navigation frame, click **Approve** under the Payment Processing section.
2. Click **Sale All** in the Approve window.
3. In the Approve Results window, you will see a progress bar indicating the status of your sale request. When processing is complete, success or failure status will appear next to each order submitted for sale. Upon successful completion of this request, the order which you partially approved and deposited earlier contains a second payment (for the remaining amount). The third order contains one payment for the entire order amount.

4. When this step is complete, click **Return to the Approve Screen**.

Settling batches

In this exercise, you will *settle* the batch that contains the payments you have created so far. The procedure for settling batches is as follows:

1. From the navigation frame, click **Batch Search** under the Payment Processing section.
2. At the Batch Search window, you will be prompted to enter the following information (note that for the purposes of this tutorial, you will only fill in the payment type:*CyberCash*):

Merchant	The name of the merchant whose batch you are searching for. **Note:** If there are fewer than 500 merchants in the WebSphere Commerce Payments database, select the merchant name from the drop-down list. If there are more than 500 merchants in the WebSphere Commerce Payments database, enter the merchant number.
Batch Number	The number that uniquely identifies the batch within the merchant. Assigned when the payment is deposited.
State	The state of the batch: • Open • Closed
Status	The balance status of this batch: • **Balanced:** the batch has been successfully balanced (that is, all totals agree). • **Out of balance:** an unsuccessful attempt has been made to balance this batch (that is, all totals do not agree).
Payment Type	Identifies the payment type, or protocol, used to place the order. Select **CyberCash**.
Batch Open Date	Use the *after* and *before* fields below to search for batches opened during the specified range in time: • **After**: Specify a date to search for all batches opened on and after this date. • **Before**: Specify a date to search for all batches opened on and before this date.
Batch Closed Date	Use the *before* and *after* fields below to search for batches closed during the specified range in time: • **After**: Specify a date to search for all batches closed on and after this date. • **Before**: Specify a date to search for all batches closed on and before this date.

3. Click **Search** to initiate a batch search.
4. Click the batch number to view information about the batch.
5. From the Batch window, you can view useful batch information, including the total number and amount of both payments and credits in the batch. Click **Batch Details** to see a detailed listing of all payments and credits in this batch. You will see the four payments you just created and no credits.
6. Click **Settle** to settle the batch.
7. When processing is complete, you will return to the Batch window, which will contain a success or failure indication for the settlement request. You will notice

that the *Balance Status* field on the screen contains no value (indicated by the ″-″ symbol). Since batches under the processing model of this account do not directly represent the batches maintained by the financial institution, no balancing takes place. This field is only used with the Merchant-controlled processing model.

Issuing a credit

Credits are issued against orders and can be given for any amount. To issue a credit, you need to find the order against which you are issuing the credit:

1. From the navigation frame, click **Order Search** under the Payment Processing section.
2. At the Order Search window, you will be prompted to enter the following information (note that for the purposes of this tutorial, you will not be entering any parameter information in the fields to narrow your search):

Merchant	The name of the merchant whose order you are searching for. **Note:** If there are fewer than 500 merchants in the WebSphere Commerce Payments database, select the merchant name from the drop-down list. If there are more than 500 merchants in the WebSphere Commerce Payments database, enter the name of a merchant.
Order Number	A number assigned by the merchant that uniquely identifies the order.
State	The state of the order: • Refundable • Canceled • Closed
Payment Type	Identifies the payment type, or protocol, used to place the order. Select **CyberCash**.
Order Date	Use the *after* and *before* fields below to search for orders opened during the specified range in time: • **After**: Specify a date to search for all orders opened on and after this date. • **Before**: Specify a date to search for all orders opened on and before this date.
Order Amount	• **Currency**: The currency used to place this order. Select the currency type from the drop-down list. • **Greater than**: Specify a value to retrieve all orders with order amounts that are greater than or equal to the value you specify. • **Less than**: Specify a value to retrieve all orders with order amounts that are less than or equal to the value you specify.

3. Click **Search** to initiate an order search.
4. From the next window, click an order number for an order in Refundable state, to view the details of that order.
5. From the Order window, click **Credit** to create a credit against this order.
6. At the Create Credit window, the following information displays:

Currency	The type of currency used to place this order. This is a read-only field.
Order Amount	The total amount of the order expressed in the currency used to place the order. This is a read-only field.

Approved Amount	This field displays the approved amount. This is a read-only field.
Deposited Amount	This field displays the deposited amount. This is a read-only field.
Credit Amount	This is the total amount of the order.

Enter the credit amount (any amount up to the deposited amount of the order) and click **Credit**.

When credit processing has completed, you will be returned to the Order window and notified of credit success or failure. You will notice on the Order window that the newly created credit appears under **Credits** at the bottom of the window. If you view details, you will see the same CyberCash-specific fields as you saw in the Payment details window.

Viewing batch totals

The last exercise in this tutorial is viewing daily batch totals. The WebSphere Commerce Payments Reports function allows you to view *daily totals* for batches in a closed state. (See the *IBM WebSphere Commerce Payments Adminitrator's Guide* for more information on batch states.) To generate a daily batch totals report:

1. From the navigation frame, click **Reports** under the Payment Processing section.
2. From the Reports window, click **Daily Batch Totals**.
3. At the Batch Totals Report window, you will be prompted to enter the date for which you would like a batch totals report. WebSphere Commerce Payments will compute the batch totals for the date entered and generate a report. *Leave this field blank to generate a report for the current date*.
4. Click **Search** to generate the batch totals report.

The Daily Batch Totals report computes the totals for all batches that were closed on the date specified on the Search screen. These totals will be computed on a per-currency basis, so there will be one line per currency. Note that these totals cover all payments and credits made for all payment types (not just those made through the Cassette for CyberCash).

You have just completed a day in the life of a Payments administrator and a Merchant administrator. While individual business models may vary, this tutorial outlines the basic path to establishing a working WebSphere Commerce Payments and demonstrates fundamental payment processing implemented through the Cassette for CyberCash. For more information on specific fields in the WebSphere Commerce Payments user interface, see the online Help.

Separate approvals and deposits

Because of the restrictions associated with the processing model of the tutorial account (Acquirer-controlled (AutoDeposit Only)), we were unable to perform separate approvals and deposits. These functions are available under accounts that use the Acquirer-controlled or Merchant-controlled processing models. Therefore, a brief description of these functions follows, for use as a guide if you have your own accounts defined for either of these processing models.

Approve

Approval without deposit is performed through the same windows as the Sale function (that is, the Approve or Order windows). Instead of choosing the **Sale**, **Sale Selected**, or **Sale All** buttons as described, use the **Approve**, **Approve Selected** or **Approve All** buttons.

Deposit

Once a Payment has been created and approved via the Approve function, you must use the Deposit function to actually capture the approved funds. As demonstrated in “Approving multiple orders at one time” on page 33, multiple payments can be associated with a single order. Therefore, you may see the same order number appear multiple times in the same list, each time with different payment information. To deposit a payment that has previously been approved:

1. From the navigation frame, click **Deposit** under the Payment Processing section.
2. Check the box next to each of the listed payments that you want to deposit and then click **Deposit Selected**.
3. In the Deposit Results window, you will see a progress bar indicating the status of your deposit request. When processing is complete, success or failure status will appear next to each order submitted for deposit.
4. When this step is complete, click **Return to the Deposit Screen**.

Note that a **Deposit All** button is also available in the Deposit screen, should you want to deposit the full approval amount of all undeposited payments. This operates much like the **Sale All** and **Approve All** buttons that you have already seen.

You may deposit only *part* of a payment, in much the same way that you can approve or sale only part of an order:

1. From the Deposit window, click the payment number for the payment that you want to partially deposit.
2. The Payment window appears, as described in “Approving orders from the Order window” on page 32. Click the **Deposit** button at the bottom of this screen to deposit all or part of the approved amount.
3. On the Deposit Payment screen, change the deposit amount to a value less than the full approval amount and click **Deposit**.
4. When the deposit has been processed, you will return to the Payment window, which will be updated with the new deposit amount.

Testing with a live CyberCash CashRegister

Before running your production payment system with the WebSphere Commerce Payments CyberCash cassette and the CyberCash CashRegister, run various transactions (payments, returns, voids) against all of the credit cards configured for your CyberCash account.

The test credit card numbers, transaction formats and various transaction processing errors to be used during testing can be found in the *Merchant Connection Kit User Guide* in Portable Document Format (PDF) obtained online at: **http://www.cybercash.com/cashregister/support/docs/**

Chapter 5. Cassette for CyberCash Cashier profiles

The Cashier is WebSphere Commerce Payments code that can be invoked by client applications - such as merchant software - to simplify the process of creating WebSphere Commerce Payments orders and payments. The Cashier uses XML documents called *profiles* that describe how orders should be created for a given cassette. This allows the client code writer to concentrate on integrating with the WebSphere Commerce Payments in a generic way rather than having to write code that deals with cassette-specific information.

It is still possible to create WebSphere Commerce Payments orders without using the Cashier; programs can use the client access library or the HTTP/XML interface to use the AcceptPayment and ReceivePayment APIs. However, the use of the Cashier is preferred since it allows the potential for new cassettes to be introduced to the system without the need for rewriting any code. For more information on the Cashier, see the *IBM WebSphere Commerce Payments Programmer's Guide and Reference*.

A Cashier profile represents a description of how WebSphere Commerce Payments orders should be created for a particular payment method. Profiles are XML documents that contain all the information needed by the Cashier to create WebSphere Commerce Payments API requests to create orders for a cassette supporting that payment method. All profiles must include the following data:

- An indication of whether a wallet is used - this flag will be used to determine whether the Cashier should use the AcceptPayment or ReceivePayment command
- Required WebSphere Commerce Payments parameters
- Required cassette parameters
- Specifications for how the Cashier should supply values for each of the above parameters

In addition, profiles may also contain the following optional data:

- An indication of which WebSphere Commerce Payments instance to use for each profile
- Optional WebSphere Commerce Payments parameters
- Optional cassette parameters
- Buy page information that specifies how client code should build buy pages to collect buyer information. For example, the buy page information might contain an HTML form that collects credit card information required by a specific cassette.
- An indication of whether diagnostic information is to be enabled for the profile

Cashier profiles allow parameter values to be specified in four different ways:

1. Hard-coded as constants in the profile
2. Passed as an environment variable on the *CollectPayment()* call
3. Specified as originating from a relational database field
4. Specified as being calculated by Cashier extension code

Following is the Cassette for CyberCash Cashier profile:

Table 5. Cassette for CyberCash Cashier Profile

CyberCash Cashier profile	Function

Table 5. Cassette for CyberCash Cashier Profile (continued)

SampleCheckoutCyberCash.profile	Uses full AVS information

Note: For **iSeries**, the Cassette for CyberCash Cashier profile is in the directory:
`/QIBM/UserData/PymSvr/profiles/`

Chapter 6. Using WebSphere Commerce Payments Commands with the Cassette for CyberCash

This chapter describes for each included WebSphere Commerce Payments API command:

- All CyberCash-specific protocol parameters
- Any special notes related to the Cassette for CyberCash handling of Framework parameters. You can also refer to “Support for the WebSphere Commerce Payments command set” on page 13 for additional comments on cassette-specific parameter behavior.

Note: For any Framework commands that are not listed here, there are no specific CyberCash parameters or unique behaviors. See the *IBM WebSphere Commerce Payments Programmer's Guide and Reference for Multiplatforms*, for a complete listing of generic Framework commands.

Cassette for CyberCash Commands

The following section outlines information specific to the CyberCash protocol for the parameters on WebSphere Commerce Payments commands. This information serves as a supplement to the command information contained in the *IBM WebSphere Commerce Payments Programmer's Guide and Reference*. Look at the keyword tables for each command to determine if there are any special considerations when you use the CyberCash CashRegister Service.

AcceptPayment

- For Acquirer-controlled (AutoDeposit Only) accounts, if you specify the AutoApprove option, you must also specify the AutoDeposit option. This is required because the processing model only allows authorization with immediate capture.
- For both types of Acquirer-controlled accounts, successful completion of this command with the AutoApprove and AutoDeposit options will cause the resulting Payment object to move into Closed state, since the actual settlement of the deposit is carried out by the acquirer on its own schedule.

The ApproveFlag for AcceptPayment can be set to ″0″, ″1″, or ″2″. The default setting is ″0″. An ApproveFlag of ″0″ indicates that the transaction should not be approved.An ApproveFlag of ″1″ indicates that the transaction should be approved automatically. An ApproveFlag of ″2″ indicates that the transaction should be approved asynchronously. See the IBM WebSphere Commerce Payments Programmer's Guide and Reference for more information on Asynchronous Auto Approve.

Required keywords for AcceptPayment command

Required Keywords	Value
$BRAND	Specifies brand of card. Value is specified in ASCII characters, 1- 40 characters long. For example: ″VISA.″ (case sensitive)
$EXPIRY	Specifies card expiration date. The value is specified as a 6 character string, in the form ″YYYYMM.″ For example: ″199907″ (for July, 1999).

Required keywords for AcceptPayment command

$PAN	Specifies card account number. The value is specified as a 1- 19 digit numeric string. For example: ″2222333344445555.″

Optional keywords for AcceptPayment command

Optional Keywords	Value
$AVS.CITY	Specifies the cardholder's city for the Address Verification Service. The value is specified as a 1- 50 character string. For example: ″Johnson City.″
$AVS.COUNTRYCODE	Indicates the cardholder's ISO-3166 country code for the Address Verification Service. Valid values are from ″1″ to ″999.″
$AVS.POSTALCODE	Specifies the cardholder's postal code for the Address Verification Service. The value is specified as a 1-14 character string. For example: ″37614″, ″37614-1255.″
$AVS.STATEPROVINCE	Specifies the cardholder's state or province. The value is specified as a 1- 50 character string. For example: ″TN.″
$AVS.STREETADDRESS	Specifies the cardholder's street address for the Address Verification Service. The value is specified as a 1- 128 character string. For example: ″317 Jodie Drive.″
$CARDHOLDERNAME	Specifies the cardholder's name. This value is specified as a 1- 64 character string.

Note: The $AVS.CITY, $AVS.POSTALCODE, $AVS.STATEPROVINCE, $AVS.STREETADDRESS and $CARDHOLDERNAME parameters are only for use with the U.S. Address Verification Service. Only English ASCII characters are allowed in these fields.

The following table presents cassette-specific processing of Framework parameters.

Cassette-specific processing of parameters for the AcceptPayment command

Keywords	Value
APPROVEFLAG	Integer in ASCII characters. Indicates whether the approvals should be attempted automatically. Default is 0, if using acquirer profile 1. If the ApproveFlag=1 or 2 and is specified for acquirer profile 1, then DepositFlag=1 is required. 0 - Indicates transaction should not be approved. 1 - Indicates transaction should be approved automatically. 2 - Indicates transaction should be approved asynchronously.
DEPOSITFLAG	Valid values are ″0″ and ″1″. If using acquirer profile 1, must be set to ″1″, when APPROVEFLAG=1 or 2. Optional for acquirer profiles 2 and 3. Default =″0.″

Cassette-specific processing of parameters for the AcceptPayment command

PAYMENTAMOUNT	A 32–bit positive integer in ASCII characters.

For valid values, refer to the *IBM WebSphere Commerce Payments Programmer's Guide and Reference*, *Version 3.1*.

Approve

- For Acquirer-controlled (AutoDeposit Only) accounts, you must specify the AutoDeposit option. This is required because the processing model only allows authorization with immediate capture.
- For both types of Acquirer-controlled accounts, successful completion of this command with the AutoDeposit option will cause the resulting Payment object to move into Closed state, since the actual settlement of the deposit is carried out by the acquirer on its own schedule.
- If the authorization request is successful, the Payment object goes into the PAYMENT_APPROVED state. If the CashRegister returns any of these action codes:

  100 Declined for various reasons

  101 Declined due to expired card

  105 Declined due to problem with card

  111 Declined due to problem with card

  107 Decline, pick up card from card holder

  200 Decline, pick up card from card holder

  or some other hard-failures without an action code set, then the Payment object goes into the PAYMENT_DECLINED state. Any other failure results in the Payment object being moved into the PAYMENT_VOID state.

The following table presents cassette-specific processing of Framework parameters.

Cassette-specific processing of parameters for the Approve command

Optional Keywords	**Value**
BATCHNUMBER	Not allowed. Must not be specified since all batches are opened implicitly.
DEPOSITFLAG	Valid values are ″0″ and ″1.″ Must be set to ″1″, for acquirer profile 1. Optional for acquirer profiles 2 and 3. Default value for acquirer profiles 2 and 3 is ″0.″

ApproveReversal

When using acquirer profile 1, this command is always rejected with return codes PRC_VERB_NOT_VALID_IN_PRESENT_STATE and RC_PAYMENT because payments are never left in approved state.

- This command is not supported for Acquirer-controlled (AutoDeposit Only) accounts since successful approvals place the resulting Payment object into Closed state. Use the Refund command instead.
- This command does not result in a reversal being sent to the financial institution. The reversal is recorded in the WebSphere Commerce Payments database and the authorization is left to expire at the processor. The expiration period varies per processor, but typically lasts several days.

- Both full and partial reversals are supported. If the Amount value is ″0,″ a full reversal is performed and the payment is put into the VOID state. If the Amount is non-″0,″ and this amount is not greater than the original order amount, then a partial reversal is done. The old payment is void and the new payment is sent to the CashRegister for authorization.

BatchOpen

This command is not supported. All batches are opened implicitly. If this command is issued with PAYMENTTYPE set to ″CyberCash″, the command will fail with return codes: PRC_COMMAND_NOT_SUPPORTED, RC_NONE.

BatchPurge

This command is not supported. The command will fail with return codes PRC_COMMAND_NOT_SUPPORTED, RC_NONE

BatchClose

This command is fully supported as documented in the*IBM WebSphere Commerce Payments Programmer's Guide and Reference* with the exception of the FORCE parameter. FORCE is not supported and will be ignored if specified.

CassetteControl

This command is not supported. The command will fail with return codes PRC_COMMAND_NOT_SUPPORTED, RC_NONE

CloseOrder

The Delete option of this command may only be used after every Batch associated with this Order has been closed.

CreateAccount

Required keywords for CreateAccount command

Required Keywords	Value
$ACQUIRERPROFILE	Acquirer's processing profile. Valid acquirer profile values are: • 1 - HostCapture/AuthCapture (Acquirer-controlled batch, sale transactions only) • 2 - HostCapture/PostAuth (Acquirer-controlled batch, separate capture required) • 3 - TerminalCapture (Merchant-controlled batch) **Note:** If acquirer profile = 1 and Account apApproveFlag = 1, then the cassette will automatically set apDepositFlag.
$CURRENCY	ISO 3166 currency code associated with this account. 3 digits, 1-999.

Required keywords for CreateAccount command

$CYBERCASHID	The merchant's CyberCash ID (CCID) under which all transactions for this account will be performed. This identifier is obtained through the CyberCash Integrated Merchant Registration process. Character string between 1- 255.
$MERCHANTKEY	The CyberCash Merchant Key provided with the CCID. This identifier is obtained through the CyberCash Integrated Merchant Registration process. 1– 64 characters, *entered exactly as provided* by CyberCash.

CreateMerchantCassetteObject

Manipulates the Cassette for CyberCash Brand object. See "Brand" on page 55 for more information regarding this object.

Required keywords for CreateMerchantCassetteObject command

Required Keywords	Value
OBJECTNAME	″BRAND.″ Tells the cassette that this is to be a brand object.
$BRAND	The credit/debit card brand, exactly as merchants must enter it. 1-40 characters long (case sensitive).

The following table presents cassette-specific processing of Framework parameters.

Cassette-specific processing of parameters for the CreateMerchantCassetteObject command

Required Keywords	Value
ACCOUNTNUMBER	The account number of the account to which this brand object will belong.
CASSETTENAME	″CyberCash″ (case sensitive)

DeleteMerchantCassetteObject

Manipulates the Cassette for CyberCash Brand objects. See "Brand" on page 55 for more information regarding this object.

Required keywords for DeleteMerchantCassetteObject command.

Required Keywords	Value
OBJECTNAME	″BRAND.″ Tells the cassette that this is to be a brand object.
$BRAND	The credit/debit card brand, exactly as merchants must enter it. 1- 40 characters long (case sensitive).

The following table presents cassette-specific processing of Framework parameters.

Cassette-specific processing of parameters for the DeleteMerchantCassetteObject command

Required Keywords	Value
ACCOUNTNUMBER	The account number of the account to which this brand object will belong.
CASSETTENAME	″CyberCash″ (case sensitive)

Deposit

- For Acquirer-controlled accounts, successful completion of this command places the Payment object into Closed state, since the actual settlement of the Deposit is carried out by the acquirer on its own schedule.
- This command is never allowed for Acquirer-controlled (AutoDeposit Only) accounts since approvals under such accounts require AutoDeposit. Because of this requirement, Payments under this processing model will never be left in Approved state, which allows the Deposit command.
- For Merchant-controlled accounts, successful completion of this command places the Payment object into Deposited state.

Cassette-specific processing of parameters for the Deposit command

Optional Keyword	Value
BATCHNUMBER	Not allowed. Must not be specified since all batches are opened implicitly.

DepositReversal

- The DepositReversal command is never allowed for either type of Acquirer-controlled account. Since successful deposits under these processing models place the Payments into Closed state, the Payments will never be left in Deposited state, which is the only state that allows the DepositReversal command.
- Successful completion of this command under Merchant-controlled accounts places the Payment into Void state, in order to conform to the way the CashRegister handles reversals.

ModifyAccount

Optional keywords for ModifyAccount command

Optional Keywords	Value
$MERCHANTKEY	The CyberCash Merchant Key provided with the CCID. This identifier is obtained through the CyberCash Integrated Merchant Registration process. 1 - 64 characters, *entered exactly as provided* by CyberCash.

ModifyCassette

Optional keywords for ModifyCassette command

Optional Keywords	Value

Optional keywords for ModifyCassette command

$ATTEMPTINTERVAL	Number of seconds to wait until trying the next set of retries. If not specified, the default is ″300″ (5 minutes). Integer ≥ 0≤1440.
$CASHREGISTERHOST	Hostname of the CyberCash CashRegister. 1–255 ASCII characters long. If not specified, the default is ″**cr.cybercash.com**.″
$CASHREGISTERPORT	Port number of the CashRegister service. Integer, ≥ 0 ≤ 65535. If not specified, the default is ″80.″
$MAXATTEMPTS	Maximum number of retry sets. Integer, ≥ 0. If not specified, the default is ″3.″ Maximum is 2147483647.
$MAXRETRIES	Maximum number of retries to try for each retry set. Integer ≥ 0. If not specified, the default is ″0.″ Maximum is 2147483647.
$READTIMEOUT	Number of seconds to wait for a response for each CashRegister command sent. Integer ≥ 0≤1440. If not specified, the default is ″60.″
$SOCKSHOST	Hostname of the Socks server through which the WebSphere Commerce Payments should connect to the CashRegister. 1 - 255 ASCII characters long. If not specified, the cassette will not attempt to use a Socks server.
$SOCKSPORT	TCP port number of the Socks server. Integer, ≥ 0 ≤ 65535.

ModifyMerchantCassetteObject

This command is not supported. The command will fail with return codes PRC_COMMAND_NOT_SUPPORTED, RC_NONE

ReceivePayment

This command is not supported. If this command is issued with payment type set to ″CyberCash″, the command will fail with return codes PRC_COMMAND_NOT_SUPPORTED, RC_NONE.

Refund

For both types of Acquirer-controlled accounts, successful completion of this command places the resulting Credit object into Closed state, since the actual settlement of the refund is carried out by the acquirer on its own schedule.

The following table presents cassette-specific processing of Framework parameters.

Cassette-specific processing of parameters for the Refund command.

Optional Keyword	Value
BATCHNUMBER	Not allowed. Must not be specified since all batches are opened implicitly.

RefundReversal

The RefundReversal command is never allowed for either type of Acquirer-controlled account. Since successful refunds under these processing

models place the Credits into Closed state, the Credits will never be left in Refunded state, which is the only state that allows the RefundReversal command.

Notes on specific return codes

The following are specific return code pairs that you might encounter while working with Cassette for CyberCash objects:

Return Code Pairs	Primary Return Code	Secondary Return Code	Description
PRC_PARAMETER_VALUE_ERROR, RC_BRAND	7	208	Invalid value specified on $BRAND parameter. If you receive these codes when creating a new brand, it indicates that an unsupported character was entered in the brand identifier. Brand identifiers may only contain ASCII characters, excluding the double quote (″), single quote (’) and backslash (\).
PRC_CRYPTO_ERROR, RC_CASSETTE_ENCRYPTION_ERROR	56	1008	Cryptographic error: If any of your commands receive these codes, it is likely that the CyberCashID or the MerchantKey value of the associated account is set incorrectly.

For additional information on return codes, see Appendix A, “CyberCash Return Codes” on page 59.

Chapter 7. Using CyberCash objects

The object model of the Cassette for CyberCash closely reflects the generic model of the WebSphere Commerce Payments. This section describes each of the cassette extensions to the various Framework objects, as well as new objects defined exclusively by the cassette.

The WebSphere Commerce Payments query command set allows merchant software to search for and retrieve the data objects maintained in the WebSphere Commerce Payments database. The results of each query call are returned in the form of an XML PSApiResult document. Cassette for CyberCash object extensions appear in these documents as extensions to the generic objects of the Framework. Specifically, the extensions appear as CassetteExtensionObject and CassetteConfigObject elements within the higher-level elements that represent each generic object. For each generic object that the Cassette for CyberCash extends, this section will present a definition of each extension, as well as an example of the XML format that the object takes in a PSApiResult document.

Financial objects used by Cassette for CyberCash

Each of the Framework's generic financial objects is extended by the Cassette for CyberCash.

Order

Cassette properties that belong to a PSOrderObject

Field Name	Description
avsCity	Cardholder's city of residence. This value will only be present if a non-null $AVS.CITY value was specified on the ACCEPTPAYMENT command.
avsCountry	ISO 3166 country code of cardholder's country of residence. This value will only be present if a non-null $AVS.COUNTRY value was specified on the ACCEPTPAYMENT command.
avsPostalCode	Cardholder's postal code. This value will only be present if a non-null $AVS.POSTALCODE value was specified on the ACCEPTPAYMENT command.
avsState	Cardholder's state or province of residence. This value will only be present if a non-null $AVS.STATEPROVINCE value was specified on the ACCEPTPAYMENT command.
avsStreet	Cardholder's street address. This value will only be present if a non-null $AVS.STREETADDRESS value was specified on the ACCEPTPAYMENT command.
brand	Credit/Debit card brand as specified on the ACCEPTPAYMENT command. This value will always be present.
cardholderName	Cardholder's name. This value will only be present if a non-null $CARDHOLDER value was specified on the ACCEPTPAYMENT command.
expiry	Credit/Debit card expiration date in the form yyyymm, as specified on the ACCEPTPAYMENT command. This value will always be present.
pan	Credit/Debit card number as specified on the ACCEPTPAYMENT command. This value will always be present.

Note: For the properties expiry and pan, the data returned can be masked if the user performing the query is not allowed to view sensitive data.

Order object XML example

This XML example shows an Order object and its cassette extensions:

```
<?xml version="1.0"?>
<PSApiResult objectCount="1" primaryRC="0" secondaryRC="0">
  <OrderCollection size="1" withCredits="0" withPayments="0">
    <PSOrder ID="O:855:70946757" amount="500" amountExp10="-2"
approvesAllowed="1" brand="BrandOne" currency="840" merchantAccount="1"
merchantNumber="855" merchantOriginated="1" numberOfCredits="0"
      numberOfPayments="0" orderNumber="70946757"
orderURL="http://merchant.us.ibm.com/ordermanager"
      paymentType="CyberCash" state="order_refundable"
timeStampCreated="938789797000" timeStampModified="938789804000"
  unapprovedAmount="50">
      <CassetteExtensionObject>
        <CassetteProperty propertyId="cardholderName" value="Abraham Lincoln"/>
        <CassetteProperty propertyId="expiry" value="200512"/>
        <CassetteProperty propertyId="pan" value="4111*11"/>
  <CassetteProperty propertyId="avsPostalCode" value="10036"/>
        <CassetteProperty propertyId="avsState" value="DC"/>
        <CassetteProperty propertyId="brand" value="BrandOne"/>
        <CassetteProperty propertyId="avsStreet"
value="1600 Pennsylvania Avenue"/>
        <CassetteProperty propertyId="avsCountry" value="840"/>
        <CassetteProperty propertyId="avsCity" value="Washington"/>
      </CassetteExtensionObject>
    </PSOrder>
  </OrderCollection>
</PSApiResult>
```

Payment

Cassette properties that belong to a PSPaymentObject

Field Name	Description
authCode	Authorization code assigned to the transaction by the Issuer. This is the value returned in the CyberCash CashRegister's `auth-code` parameter. Since some acquirers do not support this code, the field may or may not exist for a given payment or credit.
avsCode	The AVS code received from the acquirer for this payment. This field will be present only when the CashRegister returned an `avs-code` value for this payment.
commonAVSCode	The WebSphere Commerce Payments AVS code that maps to the cassette-specific AVS code received from the acquirer. The common AVS codes can be used by any cassette that supports AVS and relieve the merchant software from having to know which cassette is being used in order to interpret the result.
cyberTransID	The value by which this transaction is identified to the CyberCash CashRegister. (This is the value used as the `order-id` parameter on CashRegister commands.)
referenceNumber	This field is provided by the framework. The Cassette for CyberCash places the processor's retrieval reference code in this field. The CyberCash CashRegister returns this value in the `ref-code` parameter. Since some processors do not support this code, the field may or may not exist for a given payment or credit.

The following table presents the properties of the PSPayment Object that relate to the most recent CyberCash CashRegister command. Therefore, values may be null, even when they had previously contained data.

PSPaymentObject properties that relate to the most recent CyberCash CashRegister command

Field Name	Description
gatewayTransID	The identifier used to identify a given transaction at the gateway. (This is the value returned in the CashRegister's `merch-txn` parameter.)
lastActionCode	The most recent `action-code` value received from the CyberCash CashRegister for this payment. This field will be present only after a CyberCash CashRegister command has been processed by the gateway for this payment.
lastCashRegMsg	A text message generated by either the CyberCash CashRegister, the gateway, or the financial institution. If the message is generated by the CashRegister (that is, obtained from the CashRegister's `MErrMsg` parameter), it will be preceded by the associated message number (the `MErrCode` value). Otherwise, this field reflects the value of the `aux-msg` value (if any).
lastMStatus	The most recent `MStatus` code received from the CyberCash CashRegister for this payment. This field will be present only after a CyberCash CashRegister command has been processed for this payment.

Payment object XML Example: This XML example shows a Payment object and its cassette extensions:

```
<?xml version="1.0"?>
<PSApiResult objectCount="1" primaryRC="0" secondaryRC="0">
<PaymentCollection size="1" withOrders="0">
    <PSPayment ID="P:855:70946757:1" amountExp10="-2" approveAmount="450"
batchNumber="3" currency="840" depositAmount="450" merchantAccount="1"
merchantNumber="855" orderNumber="70946757" paymentNumber="1"
paymentType="CyberCash" referenceNumber="EO97" state="payment_deposited"
timeStampCreated="938789798000" timeStampModified="938789804000">
      <CassetteExtensionObject>
        <CassetteProperty propertyId="authCode" value="EO97"/>
        <CassetteProperty propertyId="avsCode" value="X"/>
        <CassetteProperty propertyId="CommonavsCode" value="0"/>
        <CassetteProperty propertyId="lastActionCode" value="000"/>
        <CassetteProperty propertyId="lastMStatus" value="success"/>
        <CassetteProperty propertyId="cyberTransID"
value="000000855070946757PQYP8X"/>
      </CassetteExtensionObject>
    </PSPayment>
  </PaymentCollection>
</PSApiResult>
```

Credit

Cassette properties that belong to a PSCreditObject

Field Name	Description
authCode	The authorization code received from the acquirer for this payment. This field will be present only when an `auth-code` exists for this credit.
avsCode	The AVS code received from the acquirer for this payment. This field will be present only when an `avs-code` exists for this credit.

Cassette properties that belong to a PSCreditObject

commonAVSCode	The WebSphere Commerce Payments AVS code that maps to the cassette-specific AVS code received from the acquirer. The common AVS codes can be used by any cassette that supports AVS and relieve the merchant software from having to know which cassette is being used in order to interpret the result.
cyberTransID	The value by which this transaction is identified to the CyberCash CashRegister (that is, the value used as the `order-id` parameter on CashRegister commands).
referenceNumber	This field is provided by the framework. The Cassette for CyberCash places the processor's retrieval reference code in this field. The CyberCash CashRegister returns this value in the `ref-code` parameter. Since some processors do not support this code, the field may or may not exist for a given payment or credit.

The following table presents the properties of the PSCredit Object that relate to the most recent CyberCash CashRegister command. Therefore, values may be nulled out, even when they had previously contained data.

PSCreditObject properties that relate to the most recent CyberCash CashRegister command

Field Name	Description
gatewayTransID	The identifier used to identify a given transaction at the gateway. (This is the value returned in the CashRegister's `merch-txn` parameter.)
lastActionCode	The most recent `action-code` value received from the CyberCash CashRegister for this payment. This field will be present only after a CyberCash CashRegister command has been processed by the gateway for this credit.
lastCashRegMsg	A text message generated by either the CyberCash CashRegister, the gateway, or the financial institution. If the message is generated by the CashRegister (that is, obtained from the CashRegister's `MErrMsg` parameter), it will be preceded by the associated message number (the `MErrCode` value). Otherwise, this field reflects the value of the `aux-msg` value (if any).
lastMStatus	The most recent `MStatus` code received from the CyberCash CashRegister for this payment. This field will be present only after a CyberCash CashRegister command has been processed for this credit.

Credit object XML example

This XML example shows a Credit object and its cassette extensions:

```
<?xml version="1.0"?>
<PSApiResult objectCount="1" primaryRC="0" secondaryRC="0">
  <CreditCollection size="1" withOrders="0">
    <PSCredit ID="C:854:2163:2167" amount="550" amountExp10="-2" batchNumber="2"
     creditNumber="2167" currency="840" merchantAccount="1" merchantNumber="854"
orderNumber="2163" paymentType="CyberCash" state="credit_closed"
timeStampCreated="938784344000" timeStampModified="938784347000">
      <CassetteExtensionObject>
        <CassetteProperty propertyId="gatewayTransID" value="402508731"/>
        <CassetteProperty propertyId="lastActionCode" value="007"/>
        <CassetteProperty propertyId="lastCashRegMsg" value="Financial Institution
            Response: return / credit transaction successful.  ">
        <CassetteProperty propertyId="lastMStatus" value="success"/>
        <CassetteProperty propertyId="cyberTransID"
```

```
value="000000854000002163CNE5TC"/>
      </CassetteExtensionObject>
    </PSCredit>
  </CreditCollection>
</PSApiResult>
```

Batch

Cassette properties that belong to a PSBatchObject

Field Name	Description
cyberCashBatchID	The `batch-id` value used by the CyberCash CashRegister to identify this batch. This value exists only for merchant-controlled batches for which a BATCHCLOSE command has been completed.
gatewayBatchID	The `gw-batch-id` used by the gateway to identify this batch. This value exists only for merchant-controlled batches for which a BATCHCLOSE command has been completed.

The following table presents the properties of the PSBatch Object that relate to the most recent CyberCash CashRegister command. Therefore, values may be null, even when they had previously contained data.

PSBatchObject properties that relate to the most recent CyberCash CashRegister command

Field Name	Description
lastActionCode	The most recent `action-code` value received from the CyberCash CashRegister for this batch. This value exists only for merchant-controlled batches for which a BATCHCLOSE command has been completed.
lastCashRegMsg	A text message generated by either the CyberCash CashRegister, the gateway, or the financial institution. If the message is generated by the CashRegister (that is, obtained from the CashRegister's `MErrMsg` parameter), it will be preceded by the associated message number (the `MErrCode` value). Otherwise, this field reflects the value of the `aux-msg` value (if any).
lastMStatus	The most recent `MStatus` code received from the CyberCash CashRegister for this batch. This field exists only for merchant-controlled batches for which a BATCHCLOSE command has been completed.

Batch object XML example

This XML example shows a Batch object and its cassette extensions:

```
<?xml version="1.0"?>
<PSApiResult objectCount="1" primaryRC="0" secondaryRC="0">
  <BatchCollection size="1" withCredits="0" withPayments="0">
    <PSBatch ID="B:855:2" batchNumber="2" batchStatus="batch_balanced"
merchantAccount="1" merchantControl="1" merchantNumber="855"
paymentType="CyberCash" purgeAllowed="0" state="batch_closed"
timeStampClosed="938605197000" timeStampModified="938605204000"
      timeStampOpened="938605193000">
      <BatchTotalCollection size="1">
        <PSBatchTotal amountExp10="-2" creditAmount="550" currency="840"
numberOfCredits="1"
        numberOfPayments="0" paymentAmount="0">
        </PSBatchTotal>
      </BatchTotalCollection>
      <CassetteExtensionObject>
        <CassetteProperty propertyId="cyberCashBatchID" value="8248"/>
```

```
        <CassetteProperty propertyId="lastActionCode" value="500"/>
        <CassetteProperty propertyId="lastMStatus" value="success"/>
        <CassetteProperty propertyId="gatewayBatchID" value="8248"/>
      </CassetteExtensionObject>
    </PSBatch>
  </BatchCollection>
</PSApiResult>
```

Administration objects used by Cassette for CyberCash

The Cassette for CyberCash uses and extends these Framework objects for WebSphere Commerce Payments administration:

- Cassette
- Account
- Brand (extension of Account)

Each administration object is defined by its attributes, or fields. The field names and field descriptions are shown for each administration object.

Framework object

Cassette properties that belong to a PSCassetteObject

Field Name	Description
attemptInterval	Number of seconds to wait until trying the next set of retries
cashRegisterHost	Hostname of the CyberCash CashRegister
cashRegisterPort	Port number of the CashRegister service
maxAttempts	Maximum number of retry sets
maxRetries	Maximum number of retries to try for each retry set
readTimeout	Number of seconds to wait for a response for each CashRegister command sent
socksHost	Hostname of the Socks server through which the WebSphere Commerce Payments should connect to the CashRegister
socksPort	TCP port number of the Socks server

Cassette object XML example

This XML example shows a Cassette object and its cassette extensions:

```
<?xml version="1.0"?>
<PSApiResult objectCount="1" primaryRC="0" secondaryRC="0">
  <CassetteCollection>
    <PSCassette active="1" cassette="CyberCash" changesPending="0"
companyPkgName="ibm" enabled="1" traceSetting="0" valid="1">
      <CassetteExtensionObject>
        <CassetteProperty propertyId="maxRetries" value="0"/>
        <CassetteProperty propertyId="readTimeout" value="60"/>
        <CassetteProperty propertyId="cashRegisterHost"
value="cr.cybercash.com"/>
        <CassetteProperty propertyId="socksHost" value="socks.mynet.com"/>
        <CassetteProperty propertyId="maxAttempts" value="3"/>
        <CassetteProperty propertyId="socksPort" value="1080"/>
        <CassetteProperty propertyId="attemptInterval" value="300"/>
        <CassetteProperty propertyId="cashRegisterPort" value="80"/>
```

```
      </CassetteExtensionObject>
    </PSCassette>
  </CassetteCollection>
</PSApiResult>
```

Account

The Cassette for CyberCash not only extends the Framework Account object with its own attributes, but it also uses this object as an anchor for its own Brand object. In other words, each CyberCash Brand object is associated with one and only one Account object. Both objects are described here.

Cassette properties that belong to a PSCyberCashAccountObject

Field Name	Description
acquirerProfile	Acquirer's Processing Profile. Valid acquirer profile values are: • HostCapture/AuthCapture (Acquirer-controlled batch, sale transactions only) • HostCapture/PostAuth (Acquirer-controlled batch, separate capture required) • TerminalCapture (Merchant-controlled batch)
currency	ISO 3166 currency code associated with this account. Each CyberCash account supports only one currency.
cyberCashID	CCID obtained for the merchant from the CyberCash Integrated Merchant Registration site
merchantKey	Merchant secret encryption key obtained for the merchant from the CyberCash Integrated Merchant Registration site

Brand

Brand is the only administrative object that is uniquely defined by the Cassette for CyberCash. One object exists for each brand supported for each account. The fields described here are represented as cassette property elements in the cassette configuration object. See the XML example below for more detail.

Cassette properties that belong to a PSCyberBrandObject

Field Name	Description
brand	The credit card brand.

Account and Brand object XML example

This XML example shows an Account object that contains two different Brand objects.

```
<?xml version="1.0"?>
<PSApiResult objectCount="1" primaryRC="0" secondaryRC="0">
  <MerchantAccountCollection>
    <PSMerchantAccount active="1" cassette="CyberCash" changesPending="0"
enabled="1" financialInstName="Sixteenth National Bank of Mayberry"
merchantAccount="1" merchantAccountName="Barney's software sales account"
merchantNumber="853" valid="1">
      <CassetteExtensionObject>
        <CassetteProperty propertyId="currency" value="840"/>
        <CassetteProperty propertyId="cyberCashID" value="billy-bob-25"/>
        <CassetteProperty propertyId="acquirerProfile" value="1"/>
        <CassetteProperty propertyId="merchantKey"
value="ABCDefghIJKLmnopQRSTuvwzYZ0123"/>
      </CassetteExtensionObject>
```

```
      <CassetteConfigObject active="1" changesPending="0" enabled="1"
key="BrandOne" objectId="brand" valid="1">
        <CassetteProperty displayType="readOnly" propertyId="brand"
value="BrandOne"/>
      </CassetteConfigObject>
      <CassetteConfigObject active="1" changesPending="0" enabled="1" key="BrandTwo"
       objectId="brand" valid="1">
        <CassetteProperty displayType="readOnly" propertyId="brand"
value="BrandTwo"/>
      </CassetteConfigObject>
    </PSMerchantAccount>
  </MerchantAccountCollection>
</PSApiResult>
```

Address Verification Service (AVS) result codes

When Address Verification Services (AVS) are requested on an AcceptPayment or a ReceivePayment command, subsequent approvals will reflect the results of the AVS check by storing the associated AVS result code in the Payment object. Since other credit card-oriented cassettes also support AVS, but may use different result codes, the WebSphere Commerce Payments Framework provides a set of common AVS result codes that can be used by any cassette that supports AVS. These common codes relieve merchant software from having to be aware of which cassette is being used. For more information on Address Verification Services, see “Address verification service” on page 2.

The following table illustrates the way the Cassette for CyberCash maps the CyberCash-specific AVS result codes to the WebSphere Commerce Payments Framework’s common AVS codes. Note that the CyberCash AVS result codes (defined in the left-most column) are sometimes returned as two letters (for example, both the letter `X` and the letter `Y` can be returned on a complete AVS match).

Table 6. WebSphere Commerce Payments common AVS result codes mapped to Cassette for CyberCash AVS result codes

CyberCash AVS result code (returned from acquirer)	WebSphere Commerce Payments common AVS result code	Explanations
X	0	Both the postal code (that is, the AVS 5–digit and 9–digit) and the street address match.
Y	0	Both the postal code (that is, the AVS 5–digit and 9–digit) and the street address match.
A	1	The street address matches, but the postal code does not match.
W	2	The 5–digit or 9–digit postal codes matches, but the street address does not match.
Z	2	The 5–digit or 9–digit postal codes matches, but the street address does not match.
N	3	Neither the street address nor the postal code matches.

Table 6. WebSphere Commerce Payments common AVS result codes mapped to Cassette for CyberCash AVS result codes (continued)

CyberCash AVS result code (returned from acquirer)	WebSphere Commerce Payments common AVS result code	Explanations
R	4	This constant maps: • Address information unavailable • System unavailable (maybe due to timeout) • Card type not supported • Transaction ineligible AVS return codes
S	4	This constant maps: • Address information unavailable • System unavailable (maybe due to timeout) • Card type not supported • Transaction ineligible AVS return codes
U	4	This constant maps: • Address information unavailable • System unavailable (maybe due to timeout) • Card type not supported • Transaction ineligible AVS return codes
E	4	This constant maps: • Address information unavailable • System unavailable (maybe due to timeout) • Card type not supported • Transaction ineligible AVS return codes

Note: The query command results will contain both the common as well as the cassette-specific codes.

Appendix A. CyberCash Return Codes

Almost all of the error conditions raised by the Cassette for CyberCash are reported exclusively through primary and secondary return codes:

- **Primary Return Codes**: Only Framework-defined primary return codes are used. Refer to the *IBM WebSphere Commerce Payments Programmer's Guide and Reference for Multiplatforms*, for this listing.
- **Secondary Return Codes**: The majority of the secondary return codes generated by the Cassette for CyberCash are defined by the Framework (See the *IBM WebSphere Commerce Payments Programmer's Guide and Reference for Multiplatforms*, for a listing.) The following table lists CyberCash-specific errors and their definitions.

RC_CASHREGISTERHOST	10000	Response refers to the $CASHREGISTERHOST parameter.
RC_CASHREGISTERPORT	10001	Response refers to the $CASHREGISTERPORT parameter.
RC_READTIMEOUT	10002	Response refers to the $READTIMEOUT parameter.
RC_MAXRETRIES	10003	Response refers to the $MAXRETRIES parameter.
RC_ATTEMPTINTERVAL	10004	Response refers to the $ATTEMPTINTERVAL parameter.
RC_MAXATTEMPTS	10005	Response refers to the $MAXATTEMPTS parameter.
RC_CYBERCASHID	10008	Response refers to the $CYBERCASHID parameter.
RC_MERCHANTKEY	10009	Response refers to the $MERCHANTKEY parameter.
RC_ACQUIRERPROFILE	10012	Response refers to the $ACQUIRERPROFILE parameter.
RC_CARDHOLDERNAME	10014	Response refers to the $CARDHOLDERNAME parameter.
RC_BRAND_ADMIN	10016	Response refers to the CyberCashBrand object.
RC_DUPLICATE_BRAND_CURRENCY	10017	The specified brand/currency combination has already been defined for this merchant.
RC_ORDER_HAS_TRANSACTION_IN_ACTIVE_BATCH	10018	The order could not be deleted because a payment or credit belonging to this order is a member of a non-closed batch.
RC_CASSETTE_DECLINED_PICKUP_CARD	10200	The transaction was declined by the payment processor and merchant may call card issuer and pickup card.
RC_CC_CASHREG_RESPONSE_NOT_VALID	11000	The CyberCash cassette received an invalid response message from the CyberCash CashRegister

Appendix B. Cassette for CyberCash Messages

This chapter contains the Cassette for CyberCash-specific messages.

CEPCyberCash0202 An SQL exception occurred while creating an object *object_key* **for Merchant** *merchant number* **:** *text*

Severity: Error

Explanation: An SQL exception occurred while attempting to create an object in the database. This could be due to an error connecting or writing to the database or due to an error in the data being stored.

User Response: Check the connection to the database to make sure that there is not a problem with the communication between the WebSphere Commerce Payments machine and the database server.

CEPCyberCash0203 An SQL exception occurred while updating an object *object_key* **for Merchant** *merchant number* **:** *text*

Severity: Error

Explanation: An SQL exception occurred while updating an object in the WebSphere Commerce Payments database. This could be due to an error connecting to or writing to the database or due to an error in the content of the data being stored.

User Response: Check the connection to the database to make sure that there is not a problem with the communication between the WebSphere Commerce Payments machine and the database server.

CEPCyberCash0204 An SQL exception occurred while reading an object *order_number* **from the database for Merchant** *merchant number***:** *text*

Severity: Error

Explanation: An SQL exception occurred while retrieving a record from the WebSphere Commerce Payments database. This could be due to an error connecting to or reading from the database, or due to an error in the content of the data that was read from the database.

User Response: Check the connection to the database to make sure that there is not a problem with the communication between the WebSphere Commerce Payments machine and the database server.

CEPCyberCash0205 An SQL exception occurred while accessing the database: *text***.**

Severity: Error

Explanation: An SQL exception occurred while accessing the WebSphere Commerce Payments database. This could be due to an error connecting to or accessing the database, or due to an error in the content of the data.

User Response: Check the connection to the database to make sure that there is not a problem with the communication between the WebSphere Commerce Payments machine and the database server.

CEPCyberCash0469 An SQL exception occurred while reading *reading_number***:** *text*

Severity: Error

Explanation: SQL exception occurred while reading a WebSphere Commerce Payments database table. The SQL exception text describes the exception and provides SQL state information that can be looked up in the XOPEN SQL specification. This could be the result of a disruption in communication between the WebSphere Commerce Payments and the database server, or a discrepancy between the table definition and the definition expected by the WebSphere Commerce Payments.

User Response: Refer to the SQL state information to get specific details about the nature of the problem. Test the database server connection and verify that the table definition matches the definition expected by the WebSphere Commerce Payments.

CEPCyberCash0606 An internal error occurred: *exception text***.**

Severity: Error

Explanation: An internal error occurred in the cassette. The exception text will help IBM service identify the location of the problem.

User Response: If some required operation or service is not functioning properly, contact IBM Service.

CEPCyberCash0729 The Cassette has stopped.

Severity: Information

Explanation: The Cassette is no longer accepting requests.

User Response: None.

CEPCyberCash0755 The Cassette has started.

Severity: Information

Explanation: The Cassette is now accepting requests.

User Response: None.

CEPCyberCash0760 Restarted *number* **transactions for WebSphere Commerce Payments** *hostname***.**

Severity: Information

Explanation: The specified number of pending transactions were sent to the Acquirer for WebSphere Commerce Payments *hostname*.

User Response: None.

CEPCyberCash1180 Invalid response received from CyberCash CashRegister: Missing command status field (MStatus)

Severity: Error

Explanation: A required field was not present in the response message from the CyberCash CashRegister service. This indicates that either an incomplete or incorrectly formed CashRegister response message was received by the cassette.

User Response: Reissue the request. If the problem persists, report the error to your IBM service representative.

CEPCyberCash1181 Response received from CyberCash CashRegister not valid: Could not decode return parameter *parameter name***.**

Severity: Error

Explanation: The cassette could not decode the specified response parameter from the CyberCash CashRegister service. This indicates that either an incomplete or incorrectly formed CashRegister response message was received by the cassette.

User Response: If the problem persists, report the error to your IBM service representative.

CEPCyberCash1182 Response received from CyberCash CashRegister not valid: Could not decrypt the message

Severity: Error

Explanation: The cassette could not decrypt a response message from the CyberCash CashRegister service. This indicates a mismatch between the MerchantKey value configured for the merchant's account and the corresponding encryption key at the CyberCash CashRegister.

User Response: Ensure that you have configured the account's CyberCashID and MerchantKey correctly. If the problem persists, report the error to your IBM service representative.

CEPCyberCash1183 Unable to close connection on port *port number* **to the CyberCash CashRegister at** *hostname*

Severity: Error

Explanation: The socket connection that performs the communication between the cassette and the CyberCash CashRegister failed to close successfully.

User Response: If the problem persists, contact your IBM service representative.

CEPCyberCash1184 IOException opening connection to host name *hostName* **using port** *port number*

Severity: Error

Explanation: An IOException occurred when IBM Payment Server attempted to open a socket to the host name specified using the port number indicated. This could be because the port is already in use, because there was a network error, or because the host name indicated either is not valid or not responding.

User Response: Test the connection to the host name specified. Verify that the specified host is still up and running.

CEPCyberCash1185 IOException received while attempting to write an HTTP message to the CyberCash CashRegister: *exception text*

Severity: Error

Explanation: A Java IOException was generated during a write to the CashRegister service. The exception text is that generated by Java's sockets support.

User Response: If the Payment Server has been configured to retry failed operations to the CashRegister, the operation will retry automatically, do nothing. If all retries fail, or no retries are configured, the operation will be marked as being incomplete. The operation can be reissued.

CEPCyberCash1186 IOException received while reading HTTP response message from the CyberCash CashRegister: *exception text*

Severity: Error

Explanation: A Java IOException was generated during a read from the CashRegister service. The exception text is that generated by Java's sockets support.

User Response: If the Payment Server has been configured to retry failed operations to the CashRegister, the operation will retry automatically, do

nothing. If all retries fail, or no retries are configured, the operation will be marked as being incomplete. The operation can be reissued.

Appendix C. Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this publication to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs

and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Department TL3B/Building 062
PO Box 12195
3039 Cornwallis Road
Research Triangle Park, NC 27709-2195

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement or any equivalent agreement between us.

Trademarks

The following terms are trademarks of the IBM Corporation in the United States or other countries or both:

- AIX
- DB2
- IBM
- iSeries

CyberCash, CashRegister and the CyberCash logo are trademarks of CyberCash, Inc.

Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States and/or other countries.

Lotus and Lotus Domino Go Webserver are trademarks of Lotus Development Corporation in the United States and/or other countries.

Microsoft, Windows NT, Windows 2000 and the Windows logo are trademarks or registered trademarks of Microsoft Corporation in the United States and/or other countries.

UNIX is a registered trademark in the United States and/or other countries licensed exclusively through X/Open Company Limited.

Other company, product, and service names may be trademarks or service marks of others.

Glossary

This dictionary defines technical terms used in the documentation for Payment Suite products. It includes IBM product terminology and may include selected terms and definitions from:

- The *American National Standard Dictionary for Information Systems* , ANSI X3.172-1990, copyright 1990 by the American National Standards Institute (ANSI). Copies may be purchased from the American National Standards Institute, 11 West 42nd Street, New York, New York 10036. Definitions are identified by the symbol (A) after the definition.
- The ANSI/EIA Standard—440-A, *Fiber Optic Terminology*. Copies may be purchased from the Electronic Industries Association, 2001 Pennsylvania Avenue, N.W., Washington, DC 20006. Definitions are identified by the symbol (E) after the definition.
- The *Information Technology Vocabulary* developed by Subcommittee 1, Joint Technical Committee 1, of the International Organization for Standardization and the International Electrotechnical Commission (ISO/IEC JTC1/SC1). Definitions of published parts of this vocabulary are identified by the symbol (I) after the definition; definitions taken from draft international standards, committee drafts, and working papers being developed by ISO/IEC JTC1/SC1 are identified by the symbol (T) after the definition, indicating that final agreement has not yet been reached among the participating National Bodies of SC1.
- The *IBM Dictionary of Computing*, New York: McGraw-Hill, 1994.
- Internet Request for Comments: 1208, *Glossary of Networking Terms*
- Internet Request for Comments: 1392, *Internet Users' Glossary*
- The *Object-Oriented Interface Design: IBM Common User Access Guidelines* , Carmel, Indiana: Que, 1992.

The most current *IBM Dictionary of Computing* is available on the World Wide Web at http:\\www.ibm.com/networking/nsq/nsqmain.htm.

The following cross-references are used in this dictionary:

Contrast with:
This refers the reader to a term that has an opposed or substantively different meaning.

See: This refers the reader to (a) a related term, (b) a term that is the expanded form of an abbreviation or acronym, or (c) a synonym or more preferred term.

Obsolete term for:
This indicates that the term should not be used and refers the reader to the preferred term.

A

access control. In computer security, the process of ensuring that the resources of a computer system can be accessed only by authorized users in authorized ways.

account. An account is a relationship between the merchant and the financial institution which processes transactions for that merchant. There can be multiple accounts for each payment cassette.

account name. The name you assign to the account. Its only function is to provide display information in the user interface.

acquirer. In e-commerce, the financial institution (or an agent of the financial institution) that receives from the merchant the financial data relating to a transaction and initiates that data into an interchange system.

Address Verification Service (AVS). Within IBM e-commerce, a credit and debit card scheme used by merchants to authenticate the cardholder. The merchant requests the cardholder's address and uses AVS to confirm that the cardholder is who he says he is.

ADSM. See ADSTAR Distributed Storage Manager.

ADSTAR Distributed Storage Manager (ADSM). An IBM licensed program that provides storage management and data access services in a multi-vendor, distributed computing environment.

applet. An application program, written in the Java programming language, that can be retrieved from a Web server and executed by a Web browser. A reference to an applet appears in the markup for a Web page, in the same way that a reference to a graphics file appears; a browser retrieves an applet in the same way that it retrieves a graphics file. For security reasons, an applet's access rights are limited in two ways: the applet cannot access the file system of the

client upon which it is executing, and the applet's communication across the network is limited to the server from which it was downloaded. Contrast with servlet.

approve. Within IBM e-commerce, a WebSphere Commerce Payments verb. A merchant issues this verb to create a Payment object. For cassettes that implement credit card protocols, this verb will likely map to authorization (see authorize). Other cassettes may implement the approval process differently. For IBM WebSphere Commerce Payments Cassette for SET and Cassette for CyberCash, the **approve** verb results in the creation of a Payment object and authorization to ensure that funds are available to cover payment.

approve all. Selects all orders displayed for approval.

approved amount. The amount of the order approved for payment.

approve selected. Selects the orders that you want to create a payment in the approved state for. You must perform a manual deposit on this payment to move it from approved state to deposit state.

asymmetric. In computer security, pertaining to the use of different keys for encryption and decryption.

authentication. (1) In SETCo., the process that seeks to validate identity or to prove the integrity of the information. Authentication in public key systems uses digital signatures. (2) In computer security, verification that a message has not been altered or corrupted. (3) In computer security, a process used to verify the user of an information system or protected resources.

authorization. (1) In SETCo., the process by which a properly appointed person or persons grants permission to perform some action on behalf of an organization. This process assesses transaction risk, confirms that a given transaction does not raise the account holder debt above the account credit limit, and reserves the specified amount of credit. (When a merchant obtains authorization, payment for the authorized amount is guaranteed provided that the merchant followed the rules associated with the authorization process.) (2) In computer security, the right granted to a user to communicate with or make use of a computer system. (T) (3) An access right. (4) The process of granting a user either complete or restricted access to an object, resource, or function.

authorization reversal. In SETCo., a transaction sent when a previous authorization needs to be canceled (that is, a full reversal performed) or decreased (that is, a partial reversal performed). A full reversal will be used when the transaction cannot be completed, such as when the cardholder cancels the order or the merchant discovers that goods are no longer available, as when discontinued. A partial reversal will be used when the authorization was for the entire order and some of the goods cannot be shipped, resulting in a split shipment.

authorize. In the credit card world, a merchant is guaranteed that cardholder funds are available to cover a transaction by first *authorizing* the transaction. The cardholder's issuer (that is, the bank that issued the card) guarantees payment.

B

balance. Within IBM e-commerce, an attribute of a WebSphere Commerce Payments Batch object. Indicates whether the merchant and financial institution agreed on the contents of the batch when it was closed. See 78 for more details.

balanced. Within IBM e-commerce, an attribute of a WebSphere Commerce Payments Batch object. The batch has been successfully balanced. All totals agree.

balance status. Within IBM e-commerce, an attribute of a WebSphere Commerce Payments Batch object. The balance status of a batch can be balanced or out of balance.

baseline. In SETCo., a baseline product is the specific product within an operating system family that is run against the SET Tests. A vendor must designate a distinct baseline product for each unrelated operating system family. Refer to the *SET Testing Policies and Procedures* for a complete explanation.

batch. (1) In the credit card world , a batch is a collection of fund transfer requests that are all done at the same time (that is, *in a batch*). Individual fund transfers are not performed for each individual payment, but, rather, a group of transfers is processed so that both the merchant and the financial institution can agree on the funds that are to be transferred. Before a batch is *closed* (that is, the funds are exchanged) there is usually some type of reconciliation process so that both sides agree on the amounts. A group of records or data processing jobs brought together for processing or transmission. (2) Within IBM e-commerce, one of the fundamental WebSphere Commerce Payments objects is the Batch. A Batch is an object with which Payment and Credit objects are associated. Transfer of funds is to occur when the batch is closed. (3) A group of records or data processing jobs brought together for processing or transmission.

batch close date. One of two numeric search parameters that defines the chronological start of your search. Specify a date that precedes the batch close date for the batch you want to search.

batch number. The number that identifies the batch. WebSphere Commerce Payments assigns a number to the batch when the payment is deposited.

batch open date. One of two numeric search parameters that defines the chronological start of your search. Specify a date that precedes the batch open date for the batch you want to search.

batch number. The number that identifies the batch. The number WebSphere Commerce Payments assigns to the batch when the payment is deposited.

batch search. Search for a single batch or group of batches, based on a defined list of characteristics.

BCD. See binary-coded decimal notation.

big endian. A format for storage or transmission of binary data in which the most significant bit (or byte) is placed first. Contrast with little endian.

binary-coded decimal (BCD) notation. A binary-coded notation in which each of the decimal digits is represented by a binary numeral; for example, in binary-coded decimal notation that uses the weights 8, 4, 2, 1, the number "twenty-three" is represented by 0010 0011 (compare its representation 10111 in the pure binary numeration system). (I) (A)

bitmapped message. A variable-length transaction in which each bit in an array of bits indicates the presence or absence of a data field within the transaction.

brand. Within IBM e-commerce, the Cassette object for all of the WebSphere Commerce Payments cassettes (for example, Cassette for SET and Cassette for CyberCash). Each financial transaction for a WebSphere Commerce Payments cassette is associated with a particular brand (for example, MasterCard or VISA). Each account with a financial institution can be configured to support one or more brands.

browser. See Web browser.

browser plug-in. See Web browser plug-in.

C

CA. See certificate authority.

capture. (1) In SETCo., a transaction sent after the merchant has shipped the goods. This transaction will trigger the movement of funds from the Issuer to the Acquirer and then to the merchant account. (2) In the credit card world, payment is actually made when the funds are *captured*. All payments must be authorized *and* captured (although this work can be done at the same time). Note that all payments are associated with a batch (see "void payment" on page 79) and that the actual capture occurs when the associated batch is closed.

capture reversal. In SETCo., a transaction sent when the information in a previous capture message was incorrect or should never have been sent (such as when the goods were not actually shipped). If the capture reversal is the result of incorrect information, it will be followed by a new capture message with the correct information.

cardholder. In e-commerce, a person who has a valid payment card account and uses software that supports e-commerce.

cardholder application. In SETCo., a cardholder application, sometimes called a wallet, that is run by an online consumer that enables secure payment card transactions over a network. SET Cardholder applications must generate SET protocol messages that can be accepted by SET Merchant, Payment Gateway, and Certificate Authority components.

cascading. In high-availability cluster multiprocessing (HACMP), pertaining to a cluster configuration in which the cluster node with the highest priority for a particular resource acquires the resource if the primary node fails but relinquishes the resource to the primary node upon reintegration of the primary node into the cluster. Contrast with concurrent and rotating.

cassette. (1) In e-commerce, a software component consisting of a collection of Java classes and interfaces that can be easily installed into other software components involved in e-commerce to extend the function of these components. (2) In IBM e-commerce, a WebSphere Commerce Payments concept. The WebSphere Commerce Payments provides a framework that can support many different forms of payment. WebSphere Commerce Payments cassettes are written by IBM or third-party vendors to support different payment protocols (such as, SET and CyberCash) within the WebSphere Commerce Payments Framework. Thus, WebSphere Commerce Payments is an extensible product that can support additional protocols.

cast. In programming languages, an operator that converts the value of its operand to a specified type.

CERN. Conseil Européen pour la Recherche Nucléaire (European Laboratory for Particle Physics). Located in Geneva, Switzerland, CERN initiated the World Wide Web and was the first organization to create a Web server. The CERN Web server is the basis for many commercially available servers.

certificate. (1) In e-commerce, a digital document that binds a public key to the identity of the certificate owner, thereby enabling the certificate owner to be authenticated. A certificate is issued by a certificate authority (CA). (2) In SETCo., a certificate that has been digitally signed by a trusted authority (usually the cardholder financial institution) to identify the user of the public key. SET defines the following certificate types:

- signature
- key encipherment
- certificate signature
- CRL signature

certificate authority (CA). (1) In e-commerce, an organization that issues certificates. The CA

authenticates the certificate owner's identity and the services that the owner is authorized to use, issues new certificates, renews existing certificates, and revokes certificates belonging to users who are no longer authorized to use them. See issuer. (2) In SETCo., certificate authority refers to both the component and to the entity issuing and verifying digital certificates. The component is a product run by a Certificate Authority entity that is authorized to issue and verify digital certificates as requested by Cardholder Wallet components, Merchant Server components, and/or Payment Gateway components over public and private networks.

certificate chain. (1) In SETCo., a hierarchy of digital certificates. The certificate at the top of the hierarchy is called the ″root certificate.″ (2) In SETCo., an ordered grouping of digital certificates, including the root certificate, that are used to validate a specific certificate.

certificate renewal. In SETCo., the process by which a new certificate is created for an existing public key.

certificate revocation. In SETCo., the process of revoking an otherwise valid certificate by the entity that issued the certificate.

certificate revocation list. In SETCo., a list of certificate serial numbers previously issued by a certificate authority that indicate the certificates that are invalid prior to normal expiration due to compromise, disaffiliation, or some other unusual circumstance.

certification. In SETCo., the process of ascertaining that a set of requirements or criteria has been fulfilled and attesting to that fact to others, usually with some written instrument. A system that has been inspected and evaluated as fully compliant with the SET protocol by duly authorized parties and process would be said to have been certified.

certification authority. See certificate authority.

certified. In SETCo., the process of ascertaining that a set of requirements or criteria has been fulfilled and attesting to that fact to others, usually with some written instrument. A system that has been inspected and evaluated as fully compliant with the SET protocol by duly authorized parties and process would be said to have been certified.

CGI. See Common Gateway Interface.

CGI program. A computer program that runs on a Web server and uses the Common Gateway Interface (CGI) to perform tasks that are not usually done by a Web server (for example, database access and form processing). A CGI script is a CGI program that is written in a scripting language such as Perl.

CGI script. See CGI program.

clerk. (1) In IBM e-commerce, this is a WebSphere Commerce Payments concept. The WebSphere Commerce Payments has four different access rights. A clerk is defined on a per-merchant basis and has the lowest level of access. (2) A clerk is a low-level employee.

client. A computer system or process that requests a service of another computer system or process that is typically referred to as a server. Multiple clients may share access to a common server.

closed. An order moves into closed state when its associated payment, or payments, moves from deposited state into closed state (that is, when the batch associated with the payment closes). When an order is in closed state, the financial transaction is complete; monies are deposited, and the order cannot be modified. No commands are permitted for orders in this state.

cluster. In high-availability cluster multiprocessing (HACMP), a set of independent systems (called nodes) that are organized into a network for the purpose of sharing resources and communicating with each other.

cluster node. In high-availability cluster multiprocessing (HACMP), an RS/6000 system that participates in a cluster.

commerce service provider (CSP). An Internet service provider that hosts merchant shopping sites and processes payments for the merchants.

Common Gateway Interface (CGI). A standard for the exchange of information between a Web server and computer programs that are external to it. The external programs can be written in any programming language that is supported by the operating system on which the Web server is running. See CGI program.

concurrent. In high-availability cluster multiprocessing (HACMP), pertaining to a cluster configuration in which all cluster nodes use a resource simultaneously. A cluster node can fail and reintegrate into the cluster without affecting other cluster nodes or the resource. Contrast with cascading and rotating.

confidentiality. In SETCo., the protection of sensitive and personal information from unintentional and intentional attacks and disclosure.

constructor. In programming languages, a method that has the same name as a class and is used to create and initialize objects of that class.

constructor method. See constructor.

conversation. A logical connection between two transaction programs using an LU 6.2 session. Conversations are delimited by brackets to gain exclusive use of a session.

credit. In SETCo., a transaction sent when the merchant needs to return money to the cardholder (via the Acquirer and the Issuer) following a valid capture message, such as when goods have been returned or were defective.

credit reversal. In SETCo., a transaction sent when the information in a previous credit transaction was incorrect or should have never been sent.

cryptographic key. In SETCo., a value which is used to control a cryptographic process, such as encryption or authentication. Knowledge of an appropriate key allows correct decryption or validation of a message.

cryptography. (1) In SETCo., a mathematical process used for encryption or authentication information. (2) The discipline which embodies principles, means, and methods for the transformation of data in order to hide its information content, prevent its undetected modification and unauthorized use, or a combination thereof. (3) The transformation of data to conceal its contents and to prevent one person from forging or modifying another person's messages.

CSP. See commerce service provider.

CyberCash CashRegister. An electronic payment processing service that is provided by CyberCash, Inc. The CyberCash CashRegister enables merchants to accept and process various types of electronic payments for goods or services that are purchased over the Internet.

CyberCash cassette. A payment cassette that provides support for the CyberCash CashRegister.

D

daily batch totals. The Daily Batch Totals report computes the totals for all batches closed on the date specified on the Search window. The totals include all payments and credits made for all payment types.

decryption. In computer security, the process of transforming encoded text or ciphertext into plain text.

derived products. In SETCo., derived products are components that are created from a product that has received a SET Mark license. Derived products must be created from a product that has received the SET Mark, regardless of operating system family. Please refer to the *SET Testing Policies and Procedures* for a complete explanation.

deposit all . Selects all of the order payments displayed for deposit.

deposited amount . The amount deposited for a Payment. The deposited amount can be different than the approved amount.

deposit selected . Selects the order payments that you want to deposit.

digital envelope. (1) In SETCo., a cryptographic technique to encrypt data and send the encryption key along with the data. Generally, a symmetric algorithm is used to encrypt the data and an asymmetric algorithm is used to encrypt the encryption key. (2) In e-commerce, a package of encrypted data and the encryption key.

digital signature. (1) In SETCo., information encrypted with an entity private key, which is appended to a message to assure the recipient of the authenticity and integrity of the message. The digital signature proves that the message was signed by the entity owning, or having access to, the private key. (2) In e-commerce, data that is appended to, or is a cryptographic transformation of, a data unit and that enables the recipient of the data unit to verify the source and integrity of the unit and to recognize potential forgery.

distinguished name. In SET programs, information that uniquely identifies the owner of a certificate.

document type definition (DTD). The rules that specify the structure for a particular class of SGML or XML documents. The DTD defines the structure with elements, attributes, and notations, and it establishes constraints for how each element, attribute, and notation may be used within the particular class of documents. A DTD is analogous to a database schema in that the DTD completely describes the structure for a particular markup language.

DTD. See document type definition.

dual signature. In SETCo., a digital signature that covers two or more data structures by including secure hashes or each data structure in a single encrypted block. Dual signing is done for efficiency, that is, to reduce the number of public key encryption operations.

E

EAR file. An Enterprise Archive file represents a J2EE application that can be deployed in a WebSphere application server. EAR files are standard Java archive files and have the file extension .ear.

e-business. Either (a) the transaction of business over an electronic medium such as the Internet or (b) any organization (for example, commercial, industrial, nonprofit, educational, or governmental) that transacts its business over an electronic medium such as the Internet. An e-business combines the resources of traditional information systems with the vast reach of an electronic medium such as the Internet (including the World Wide Web, intranets, and extranets); it connects critical business systems directly to critical business constituencies--customers, employees, and suppliers. The key to becoming an e-business is building a

transaction-based Web site in which all core business processes (especially all processes that require a dynamic and interactive flow of information) are put online to improve service, cut costs, and sell products.

ECML. See Electronic Commerce Modeling Language.

e-commerce. (1) The exchange of goods and services for payment between the cardholder and merchant when some or all of the transaction is performed via electronic communication. (2) The subset of e-business that involves the exchange of money for goods or services purchased over an electronic medium such as the Internet.

electronic commerce. See e-commerce.

Electronic Commerce Modeling Language (ECML). In e-commerce, a universal format for wallets that streamlines the collection of electronic data for shipping, billing, and payment on a merchant's Web site and thereby enhances the online shopping experience for consumers and merchants. IBM is one of many companies that are collaborating to develop ECML.

encryption. (1) In SETCo., the process of converting information in order to render it into a form unintelligible to all except holders of a specific cryptographic key. Use of encryption protects information between the encryption process and the decryption process (that is, the inverse of encryption), against unauthorized disclosure. (2) In computer security, the process of transforming data into an unintelligible form in such a way that the original data either cannot be obtained or can be obtained only by using a decryption process.

event. In the Tivoli environment, any significant change in the state of a system resource, network resource, or network application. An event can be generated for a problem, for the resolution of a problem, or for the successful completion of a task. Examples of events are: the normal starting and stopping of a process, the abnormal termination of a process, and the malfunctioning of a server.

event listener. In IBM e-commerce, a computer program that waits to be informed of events of interest and acts upon them.

expiry. (1) The certificate expiration date assigned when the certificate was obtained. Certificates are specific to payment types (for example, SET or CyberCash.) (2) Specifies the card expiration date. An expiry value is required for SET protocol. The value is specified as a string and is used on the payment initiation message. For example, 199911 is an expiry value.

Extensible Markup Language (XML). A standard metalanguage for defining markup languages that was derived from and is a subset of SGML. XML omits the more complex and less-used parts of SGML and makes it much easier to (a) write applications to handle document types, (b) author and manage structured information, and (c) transmit and share structured information across diverse computing systems. The use of XML does not require the robust applications and processing that is necessary for SGML. XML is being developed under the auspices of the World Wide Web Consortium (W3C).

F

failover. See fallover.

fallover. In high-availability cluster multiprocessing (HACMP), an active node's acquisition of resources that were previously owned by another cluster node in order to maintain the availability of those resources.

financial institution. (1) In SETCo., an establishment responsible for facilitating customer-initiated transactions or transmissions of funds for the extension of credit or the custody, loan, exchange, or issuance of money, such as a bank or its designate. (2) Within IBM e-commerce, banks, building societies, and credit unions are examples of financial institutions. An institution that provides financial services.

financial network. Within IBM e-commerce, the aggregate of card processors, acquirers, card issuers, and other institutions through which payment card transaction processing is traditionally performed.

firewall. In communication, a functional unit that protects and controls the connection of one network to other networks. The firewall (a) prevents unwanted or unauthorized communication traffic from entering the protected network and (b) allows only selected communication traffic to leave the protected network.

force. Within IBM e-commerce, a WebSphere Commerce Payments verb. An attempt to settle a batch (see 78). If the reconciliation step fails, the batch is still not closed on the WebSphere Commerce Payments (although it may be out of balance or not closed at the financial institution).

FQDN. See fully qualified domain name.

fully qualified domain name (FQDN). In the Internet suite of protocols, the name of a host system that includes all of the subnames of the domain name. An example of a fully qualified domain name is `mycomputer.city.company.com`. See host name.

H

HACMP. See high-availability cluster multiprocessing.

handle. In the AIX operating system, a data structure that is a temporary local identifier for an object. Allocating a handle creates it. Binding a handle makes it identify an object at a specific location.

hardware token. In SETCo., a portable device (for example, smart card, PCMCIA cards) specifically designed to store cryptographic information and possibly perform cryptographic functions in a secure manner.

has been certified. A system that has been inspected and evaluated as fully compliant with the SET protocol by duly authorized parties and process would be said to have been certified.

hash. See root key hash.

heartbeat. In software products, a signal that one entity sends to another to convey that it is still active.

high-availability cluster multiprocessing (HACMP). An application service that enables up to eight RS/6000 servers to access the same data in parallel. This optimizes application execution and scalability and protects against unplanned outages and server downtime.

home page. The initial Web page that is returned by a Web site when a user specifies the uniform resource locator (URL) for the Web site. For example, if a user specifies the URL for the IBM Web site, which is `http://www.ibm.com` , the Web page that is returned is the IBM home page. Essentially, the home page is the entry point for accessing the contents of the Web site. The home page may sometimes be called the ″welcome page″ or the ″front page.″

host. To provide the software and services for managing a Web site.

HostCapture/PostAuth. Within IBM e-commerce, this is a CyberCash concept. One of the three processing models supported by the CyberCash CashRegister service. In particular, the AcquirerProfile field of an account may be set to `HostCapture/PostAuth = 2`, which indicates that the acquirer controls batch processing and a separate deposit request is required to capture the funds after a payment is authorized.

host byte order. The byte order that a central processing unit (CPU) uses to store and process data. This byte order can be big endian or little endian, depending on the particular CPU. Contrast with network byte order.

host name. In the Internet suite of protocols, the name given to a computer. Sometimes, “host name” is used to mean fully qualified domain name; other times, it is used to mean the most specific subname of a fully qualified domain name. For example, if `mycomputer.city.company.com` is the fully qualified domain name, either of the following may be considered the host name:

- `mycomputer.city.company.com`
- `mycomputer`

HTML. See Hypertext Markup Language.

HTTP. See Hypertext Transfer Protocol.

Hypertext Markup Language (HTML). A markup language that conforms to the SGML standard and was designed primarily to support the online display of textual and graphical information that includes hypertext links.

Hypertext Transfer Protocol (HTTP). In the Internet suite of protocols, the protocol that is used to transfer and display hypertext documents.

I

idempotency. (1) A property of a mathematical operation whereby repeating the operation produces no change in the final result. For example, the operation of deducting $25.00 from an account balance is not idempotent, but the operation of setting an account balance to $500.00 is idempotent. (2) In SET Secure Electronic Transaction, a property that enables the sender of a request to repeat the request with a guarantee that the outcome will be the same regardless of whether the request is lost, the response is lost, or the request or response is delayed due to network problems. Idempotency is necessary because the SET protocol works in environments where message delivery is not guaranteed, and when the sender does not receive a response, it cannot determine the cause of the delay. If a SET application does not receive a response in a reasonable amount of time, it resends the message; when the receiving SET application determines that it has already processed that message, it retrieves the previous response and sends that response again.

instrumentation. In application or system software, either (a) monitoring functions that provide performance and other information to a management system or (b) the use of monitoring functions to provide performance and other information to a management system.

identify. To establish the identity.

installment payments. In SETCo., a type of payment transaction negotiated between the merchant and the cardholder which permits the merchant to process multiple authorizations. Cardholder specifies a maximum number of permitted Authorization for paying through installment payments.

integrity. In SETCo., the quality of information or a process that is free from error, whether induced accidentally or intentionally.

interactive. In SETCo., a generic class for a network transport mechanism that is dependent on a logical session being maintained during the message exchange (for example, World Wide Web sessions).

internet. (1) In SETCo., the largest collection of networks in the world, interconnected in such a way as

to allow them to function as a single virtual network. (2) A collection of interconnected networks that use the Internet suite of protocols. The internet that allows universal access is referred to as the Internet (with a capital "I"). An internet that provides restricted access (for example, to a particular enterprise or organization) is frequently called an intranet, whether or not it also connects to the public Internet.

Internet. The worldwide collection of interconnected networks that use the Internet suite of protocols and permit public access.

interoperability. In SETCo., the ability to exchange messages and keys, both manually and in an automated environment, with any other party implementing this standard, provided that both implementations use compatible options of this standard and compatible communications facilities.

interprocess communication (IPC). The process by which programs communicate data to each other and synchronize their activities. Semaphores, signals, and internal message queues are common methods of interprocess communication.

intranet. A private network that integrates Internet standards and applications (such as Web browsers) with an organization's existing computer networking infrastructure.

IP address. The unique 32-bit address that specifies the location of each device or workstation on the Internet. For example, 9.67.97.103 is an IP address.

IPC. See interprocess communication.

issuer. (1) In SETCo., the financial institution or its agent that issues the unique primary account number (PAN) to the cardholder for the payment card brand. (2) In e-commerce, a financial institution that issues payment cards to individuals. An issuer can act as its own certificate authority (CA) or can contract with a third party for the service.

J

J2EE. (Java 2 Enterprise Edition) J2EE is designed to support applications that implement enterprise services for customers, employees, suppliers, partners, and others who make demands on or contributions to the enterprise. This can be a single module or a group of modules packaged into an .ear file with a J2EE application deployment descriptor. J2EE applications are typically engineered to be distributed across multiple computing tiers.

Java. An object-oriented programming language for portable interpretive code that supports interaction among remote objects. Java was developed and specified by Sun Microsystems, Incorporated.

Java Database Connectivity (JDBC). An application programming interface (API) that has the same characteristics as Open Database Connectivity (ODBC) but is specifically designed for use by Java database applications. Also, for databases that do not have a JDBC driver, JDBC includes a JDBC to ODBC bridge, which is a mechanism for converting JDBC to ODBC; it presents the JDBC API to Java database applications and converts this to ODBC. JDBC was developed by Sun Microsystems, Inc. and various partners and vendors.

Java Development Kit (JDK). A software package that can be used to write, compile, debug, and run Java applets and applications.

Java Runtime Environment (JRE). A subset of the Java Development Kit (JDK) that contains the core executables and files that constitute the standard Java platform. The JRE includes the Java Virtual Machine, core classes, and supporting files.

Java Virtual Machine (JVM). A software implementation of a central processing unit (CPU) that runs compiled Java code (applets and applications).

JDBC. See Java Database Connectivity.

JDK. See Java Development Kit.

JRE. See Java Runtime Environment.

JVM. See Java Virtual Machine.

K

keepalive message. In Internet communications, a message sent among nodes when no data traffic has been detected for a given period of time. This communication ensures the vitality of the session by keeping the link "alive."

key. In computer security, a sequence of symbols that is used with a cryptographic algorithm for encrypting or decrypting data. See private key and public key.

key pair. In computer security, a public key and a private key. When the key pair is used for encryption, the sender uses the public key to encrypt the message, and the recipient uses the private key to decrypt the message. When the key pair is used for signing, the signer uses the private key to encrypt a representation of the message, and the recipient uses the public key to decrypt the representation of the message for signature verification. See asymmetric and digital signature.

key ring. In computer security, a file that contains public keys, private keys, trusted roots, and certificates.

L

little endian. A format for storage or transmission of binary data in which the least significant bit (or byte) is placed first. Contrast with big endian.

M

memory leak. A condition in which a computer program allocates memory and does not free (or properly free) this memory. If the program continues to run and is not terminated, it uses large amounts of real memory and eventually runs out of memory.

merchant. In SETCo., a seller of goods, services, and/or other information who accepts payment for these items electronically. The merchant may also provide electronic selling services and/or electronic delivery of items for sale.

merchant chargeback. Within IBM e-commerce, when fraud occurs and a merchant is liable for funds not obtained, a financial institution may issue a merchant chargeback, reclaiming funds previously credited to a merchant's account.

merchant server. (1) In SETCo., a Merchant Server component is a product run by an online merchant to process payment card transactions and authorizations. It communicates with the Cardholder Wallet, Payment Gateway, and Certificate Authority components. (2) In e-commerce, a Web server that offers cataloged shopping.

merchant settings. The settings that a merchant has made for a cassette. In the WebSphere Commerce Payments user interface, the Payment System object displays as Merchant Settings.

MIME. See Multipurpose Internet Mail Extensions.

mirroring. In the AIX operating system, the maintenance of more than one copy of stored data to prevent the loss of data.

Multipurpose Internet Mail Extensions (MIME). An Internet standard for identifying the type of object being transferred across the Internet. MIME types include several variants of audio, graphics, and video.

mutex. A mutual exclusion lock. See mutual exclusion mechanism.

mutual exclusion mechanism. In software, a method for preventing two separately executing pieces of code from interfering with each other's use of a particular data object. For example, if one thread is executing a function that modifies a shared data structure, the application may need to prevent other threads from simultaneously attempting to read the data before the modifications are complete.

N

network. In SETCo., a collection of communication and information processing systems that may be shared among several users.

network byte order. The byte order that a network uses to transmit data. In the Internet, this byte order is always big endian. Contrast with host byte order.

node. See cluster node.

normal mode. In the IBM Payment Gateway, the processing scheme in which a batched SET message is presented in its entirety to the customized exits of the Payment Gateway Application. Contrast with supervisor mode.

number of credits. In SETCo., a credit is a transaction sent when the merchant needs to return money to the cardholder (via the Acquirer and the Issuer) following a valid capture message, such as when goods have been returned or were defective. Credits can be for up to the total amount of all payments associated with an Order. There can be zero or more Credits per Order.

number of payments. In SETCo., a payment is a request by the merchant to the financial institution to approve all or part of an order. In many cases, all the money authorized for collection by the order will be collected in a single payment. Some payment systems may allow the money authorized in one order (that is, one set of payment instructions) to be collected in multiple payments, depending on the business model. There can be zero or more payments per order.

O

ODBC. See Open Database Connectivity.

ODBC bridge. See Java Database Connectivity.

Open Database Connectivity (ODBC). A standard application programming interface (API) for accessing data in both relational and nonrelational database management systems. Using this API, database applications can access data stored in database management systems on a variety of computers even if each database management system uses a different data storage format and programming interface. ODBC is based on the call level interface (CLI) specification of the X/Open SQL Access Group and was developed by Digital Equipment Corporation (DEC), Lotus, Microsoft, and Sybase. Contrast with Java Database Connectivity.

online catalog. In SETCo., shopping on the Internet is simple with online catalogs. Online catalogs are Web pages that display items for sale by an online merchant.

order. An order represents all the instructions and information needed from the consumer (payer) in order for the merchant (payee) to collect money.

order amount. The amount of the order.

order fulfillment. Within IBM e-commerce, merchant systems responsible for shipping or distributing orders for which payment has been received. It is believed that an order fulfillment system would query the WebSphere Commerce Payments to determine what goods are to be shipped.

order search. Search for a single order or group of orders, based on a defined set of characteristics.

out of balance. An unsuccessful attempt was made to balance a batch. All totals do not agree.

P

password. For computer or network security, a specific string of characters entered by a user and authenticated by the system in determining the user's privileges, if any, to access and manipulate the data and operations of the system.

payment. In SETCo., a payment is a request by the merchant to the financial institution to approve all or part of an order. In many cases, all the money authorized for collection by the order will be collected in a single payment. Some payment systems may allow the money authorized in one order (that is, one set of payment instructions) to be collected in multiple payments, depending on the business model.

payment amount. The total payment amount deposited by the merchant for this order.

payment card. (1) In SETCo., a term used to collectively refer to credit cards, debit cards, charge cards, and bank cards issued by a financial institution and which reflects a relationship between the cardholder and the financial institution. (2) In e-commerce, a credit card, debit card, or charge card (a) that is issued by a financial institution and shows a relationship between the cardholder and the financial institution and (b) for which a certificate can be issued from an authenticated certificate authority.

payment cassette. A cassette that implements an electronic payment protocol.

payment gateway. (1) In SETCo., a payment gateway component is a product run by an acquirer or a designated third party that processes merchant authorization and payment messages (including payment instructions from cardholders) and interfaces with private financial networks. (2) In e-commerce, the entity that handles transactions between a merchant and an acquirer.

Payment Gateway Application. In the Payment Gateway Transaction Manager (PGTM), the component that processes SET transactions.

Payment Gateway Transaction Manager (PGTM). In the IBM Payment Gateway, the component that is the application-level routing switch. It provides the protocol-level conversion for managing incoming and outgoing communication, and it provides base services for the intelligent routing of transactions to applications. It manages the communication with merchants and routes transactions among merchants, the Payment Gateway Application, and the acquirer's private financial networks.

payment number. (1) Numeric token. (2) A unique identifier for a particular payment within an order.

payment server. In e-commerce, the electronic equivalent of a cash register that organizes and accepts payment for the goods and services selected for purchase. A payment server uses other components, such as a payment gateway and a payment management system, to complete the financial transactions.

Payment Suite. The brand name for IBM's family of payment products for e-commerce.

PGTM. See Payment Gateway Transaction Manager.

port. In the Internet suite of protocols, a specific logical connector between the Transmission Control Protocol (TCP) or the User Datagram Protocol (UDP) and a higher-level protocol or application. See well-known port.

port number. In the Internet suite of protocols, the identifier for a logical connector between an application entity and the transport service.

primary account number (PAN). In SETCo., the assigned number that identifies the card issuer and cardholder. This account number is composed of an issuer identification number, an individual account number identification, and an accompanying check digit, as defined by ISO 7812–1985.

private key. (1) In SETCo., a cryptographic key used with a public key cryptographic algorithm, uniquely associated with an entity and not made public. This key is used to create digital signatures or to decrypt messages or files. (2) In computer security, a key that is known only to its owner. Contrast with public key. See public key cryptography.

protocol. The meanings of, and the sequencing rules for, requests and responses used for managing a network, transferring data, and synchronizing the states of network components.

public key. (1) In SETCo., a cryptographic key used with a public key cryptographic algorithm, uniquely

associated with an entity publicly available. It is used to verify signatures that were created with he matched private key. Public keys are also used to encrypt messages or files that can only be decrypted using the matched private key. (2) In computer security, a key that is made available to everyone. Contrast with private key. See public key cryptography.

public key cryptography. In computer security, cryptography in which public keys and private keys are used for encryption and decryption.

purge. Within IBM e-commerce, a WebSphere Commerce Payments verb. To remove all associated Payments and Credits from a Batch object, treating it as if it has just been created.

R

random. In SETCo., a value in a set that has equal probability of being selected from the total population of possibilities and is, hence, unpredictable.

realm. In the WebSphere family of products, a database of users, groups, and access control lists. A user must be defined in a realm to access any resource belonging to that realm.

recurring payments. In SETCo., a type of payment transaction initiated by the cardholder that permits the merchant to process multiple authorizations. There are two kinds of recurring payments:

1. multiple payments for a fixed amount
2. repeated billings

refund. Identifies the Credit amount in the smallest denomination of the particular currency used to place the Order.

registration authority. In SETCo., an independent third-party organization that processes payment card applications for multiple payment card brands and forwards applications to the appropriate financial institutions.

reintegration. In high-availability cluster multiprocessing (HACMP), the actions that occur within the cluster when a component that had previously detached from the cluster returns to the cluster. These actions are controlled by the event scripts and when necessary, by manual intervention.

root certificate. In SETCo., the certificate at the top of the certificate hierarchy. See certificate chain.

root key hash. In SET programs, a hexadecimal value that is used to verify the validity of a root certificate. The hash value is published for a consumer to use when the software does not recognize the root certificate.

rotating. In high-availability cluster multiprocessing (HACMP), pertaining to a cluster configuration in which the cluster node with the highest priority for a particular resource acquires the resource if the primary node fails and retains the resource even upon reintegration of the primary node into the cluster. Contrast with cascading and concurrent.

RS/6000. A family of workstations and servers based on IBM's POWER architecture. They are primarily designed for running multi-user numerical computing applications that use the AIX operating system.

S

sale. (1) In the credit card world, a sale occurs when a transaction is authorized and marked for capture all at once rather than using a two-step process. (2) Within IBM e-commerce, a WebSphere Commerce Payments user interface verb. It means a simultaneous Approve and Deposit.

sale all. Selects all orders displayed to approve and move the associated payment directly into deposited state. The sale function automatically performs an approve and a deposit on your payment.

sale selected. Selects the orders that you want to approve and move the associated payment directly into deposited state. The sale function automatically performs an approve and a deposit on your payment.

sales transaction. In SETCo., a payment authorization transaction that allows a merchant to authorize a transaction and request payment in a single message to the acquirer.

Secure Electronic Transaction. See SET Secure Electronic Transaction.

Secure Sockets Layer (SSL). (1) In SETCo., Secure Socket Layer (SSL) (developed by Netscape Communications Company) is a standard that encrypts data between a Web browser and a Web server. SSL does not specify what data is sent or encrypted. In an SSL session, all data sent is encrypted. (2) A security protocol that provides communication privacy. SSL enables client/server applications to communicate in a way that is designed to prevent eavesdropping, tampering, and message forgery. SSL was developed by Netscape Communications Corp. and RSA Data Security, Inc.

server. In SETCo., a functional unit that provides services to one or more clients over a network. Examples include a file server, a print server, and a mail server.

servlet. An application program, written in the Java programming language, that is executed on a Web server. A reference to a servlet appears in the markup for a Web page, in the same way that a reference to a graphics file appears. The Web server executes the

servlet and sends the results of the execution (if there are any) to the Web browser. Contrast with applet.

SET. See SET Secure Electronic Transaction.

SET logo. In SETCo., the SET logo or SET Mark is your assurance that the merchant is using software that has successfully completed the SET Software Certification test.

SET cassette. A payment cassette that provides support for the SET protocol.

SET Secure Electronic Transaction. (1) In SETCo., the SET Secure Electronic Transaction™ protocol is an open industry standard developed for the secure transmission of payment information over the Internet and other electronic networks. (2) A specification for securing payment card transactions over open networks such as the Internet. SET was developed by Visa, MasterCard, IBM, and other technology companies.

settle. Within IBM e-commerce, a WebSphere Commerce Payments verb. An attempt to close a Batch object and transfer funds. As part of the settling procedure, there may be some reconciliation or balancing steps (depending on the cassette and financial institution policy) to ensure that the merchant and financial institution agree on the funds being transferred. If the reconciliation step fails, the batch may remain in an open state.

settle all. Settles all batches displayed. The batches can then submit payments and refunds for processing by a payment processor.

settle batches. Settle batches is used to submit batches (payments and refunds) for processing by a payment processor. You can choose to settle one Batch, or multiple Batches.

settle selected. Settles the batches you selected. The selected batches can then submit payments and refunds for processing by a payment processor.

sibling. In SETCo., sibling products are components which, by virtue of being within the same operating system family, are closely related to baseline products. Siblings must be of the same operating system family as the baseline product from which they were created, with identical functionality. Refer to the *SET Testing Policies and Procedures* for a complete explanation.

SMIT. See System Management Interface Tool.

socket. An endpoint provided by the transport service of a network for communication between processes or application programs.

socks-enabled. Pertaining to TCP/IP software, or to a specific TCP/IP application, that understands the socks protocol. “Socksified” is a slang term for socks-enabled.

socksified. See socks-enabled.

socks protocol. A protocol that enables an application in a secure network to communicate through a firewall via a socks server.

socks port. The port on which the Socks server is listening.

socks server. A circuit-level gateway that provides a secure one-way connection through a firewall to server applications in a nonsecure network.

SSL. See Secure Sockets Layer.

stack. A slang term for the set of protocols that comprise TCP/IP. The preferred term is TCP/IP.

supervisor. Can perform all payment processing functions for the merchant.

supervisor mode. In the IBM Payment Gateway, the processing scheme in which a batched SET message is presented as a series of individual requests to the customized exits of the Payment Gateway Application. Contrast with normal mode.

System Management Interface Tool (SMIT). An interface tool of the AIX operating system for installing, maintaining, configuring, and diagnosing tasks.

T

TEC. See Tivoli Enterprise Console.

terminal capture. Within IBM e-commerce, a CyberCash concept. One of the three processing models supported by the CyberCash CashRegister service. In particular, the AcquirerProfile field of an account may be set to `Terminal Capture = 3` , which indicates that the merchant controls batch processing.

thread. A stream of computer instructions that is in control of a process. A multi-threaded process begins with one stream of instructions (one thread) and may later create other instruction streams to perform tasks.

thread pool. The threads that are being used by or are available to a computer program.

time approved. The date and time that this Payment entry was created.

time opened. The time that the batch was created.

time ordered. The time that the order entry was created.

Tivoli Enterprise Console (TEC). A Tivoli product that collects, processes, and automatically initiates corrective actions for system, application, network, and database events; it is the central control point for events from all sources. The Tivoli Enterprise Console provides a

centralized, global view of the network computing environment; it uses distributed event monitors to collect information, a central event server to process information, and distributed event consoles to present information to system administrators.

Tivoli GEM. See Tivoli Global Enterprise Manager.

Tivoli Global Enterprise Manager (Tivoli GEM). A Tivoli product that allows system administrators to graphically monitor, control, and configure applications residing in distributed and host (S/390) environments and to use the concept of business systems management to organize related components, thereby providing a business perspective for management decisions. Tivoli Global Enterprise Manager gives information technology staff a logical view of the computing environment; this view shows, at a glance, the status of the multiple applications that comprise the enterprise's business system, including application components, the relationships among and between components, and the flow of data between the applications. By providing this view from a business perspective, Tivoli Global Enterprise Manager enables system administrators to quickly make determinations about the business impact of any component failure. Addressing technology problems from the business perspective greatly improves the effectiveness of system administrators and provides a higher level of service to users.

Tivoli Inventory. A Tivoli product that enables system administrators to gather hardware and software information for a network computing environment. It scans the managed resources and stores inventory information in the configuration repository.

Tivoli management software. The overall descriptor for software from Tivoli Systems Inc., which includes Tivoli Enterprise software (for systems management in a large organization), Tivoli IT Director (for systems management in a small or medium organization), and Tivoli Cross-Site (for the management of e-commerce systems). Tivoli management software enables organizations to centrally manage their computing resources (including the critical applications that drive business performance and profits) in a simple and straightforward manner.

Tivoli Ready. Pertaining to a product that has passed rigorous product certification testing by Tivoli Systems Inc. to ensure that the product delivers turnkey (or ″out-of-the-box″) integration with Tivoli management software. A product that has passed this certification testing carries the Tivoli Ready logo.

transaction. In SETCo., a sequence of one or more related messages.

trust chain. In SETCo., a synonym for certificate chain. See 70.

trusted root. In the Secure Sockets Layer (SSL), the public key and associated distinguished name of a certificate authority (CA).

U

uniform resource locator (URL). (1) A sequence of characters that represent information resources on a computer or in a network such as the Internet. This sequence of characters includes (a) the abbreviated name of the protocol used to access the information resource and (b) the information used by the protocol to locate the information resource. For example, in the context of the Internet, these are abbreviated names of some protocols used to access various information resources: `http` , `ftp` , `gopher` , `telnet` , and `news` ; and this is the URL for the IBM home page: `http://www.ibm.com` . (2) The address of an item on the World Wide Web. It includes the protocol followed by the fully qualified domain name (sometimes called the host name) and the request. The Web server typically maps the request portion of the URL to a path and file name. For example, if the URL is `http://www.networking.ibm.com/nsg/nsgmain.htm` , the protocol is `http` ; the fully qualified domain name is `www.networking.ibm.com`; and the request is `/nsg/nsgmain.htm`.

URL. See uniform resource locator.

user exit routine. A user-written routine that receives control at predefined user exit points. User exit routines can be written in assembler or a high-level language.

V

virtual sales slip. In SETCo., detailed information on a financial transaction which is generated by the merchant's online store and downloaded to your digital wallet. Typical items contained in the virtual sales slip are confirmation of your order, shipping details, tax (if applicable), and total amount of sale.

virtual store. An interactive simulation of a store on the World Wide Web.

void payment. Within IBM e-commerce, a verb meaning to nullify or cancel a payment operation (that is, to make it as if it never happened).

W

wallet. In the IBM Payment Suite, software that enables a user to make approved payments to authenticated merchants over public networks and to manage payment card accounts and purchases.

WAR file. A Web Archive (WAR) file is a Java archive file used to store one or more of the following: servlets; JavaServer Pages (JSP) files; utility classes; static

documents (such as HTML files, images and sound); client-side applets, beans and classes; descriptive meta-information. Its standard file extension is .war. WAR files are used to package Web modules.

Web. See World Wide Web.

Web browser. (1) Within IBM e-commerce, software running on the cardholder processing system that provides interface to public data networks. (2) A client program that initiates requests to a Web server and displays the information that the server returns.

Web browser plug-in. In SETCo., software installed on the cardholder's computer used to add functions to the Web browser.

webmaster. The person who is ultimately responsible for managing and maintaining a particular Web site.

Web page. Any document that can be accessed by a uniform resource locator (URL) on the World Wide Web. Contrast with home page.

Web server. A server that is connected to the Internet and is dedicated to serving Web pages.

Web site. A Web server that is managed by a single entity (an organization or an individual) and contains information in hypertext for its users, often including hypertext links to other Web sites. Each Web site has a home page. In a uniform resource locator (URL), the Web site is indicated by the fully qualified domain name. For example, in the URL `http://www.networking.ibm.com/nsg/nsgmain.htm` , the Web site is indicated by `www.networking.ibm.com` , which is the fully qualified domain name.

WebSphere. Pertaining to a family of IBM software products that provide a development and deployment environment for basic Web publishing and for transaction-intensive, enterprise-scale e-business applications.

well-known port. In the Internet suite of protocols, one of a set of preassigned protocol port numbers that address specific functions used by transport-level protocols such as the Transmission Control Protocol (TCP) and the User Datagram Protocol (UDP). The File Transfer Protocol (FTP) and the Simple Mail Transfer Protocol (SMTP), for example, use well-known port numbers.

World Wide Web (WWW). A network of servers that contain programs and files. Many of the files contain hypertext links to other documents available through the network.

WWW. See World Wide Web.

X

XML. See Extensible Markup Language.

Numerics

2KP transaction. A SET transaction in which the cardholder messages are unsigned and two key pairs (one for the merchant and one for the payment gateway) are used for encryption.

3KP transaction. A SET transaction in which the cardholder messages are unsigned and three key pairs (one for the merchant, one for the payment gateway, and one for the cardholder) are used for encryption.

Index

Special Characters

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Readers' Comments — We'd Like to Hear from You

IBM WebSphere Commerce Payments for Multiplatforms
Cassette for CyberCash Supplement
Version 3.1

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	☐	☐	☐	☐	☐

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	☐	☐	☐	☐	☐
Complete	☐	☐	☐	☐	☐
Easy to find	☐	☐	☐	☐	☐
Easy to understand	☐	☐	☐	☐	☐
Well organized	☐	☐	☐	☐	☐
Applicable to your tasks	☐	☐	☐	☐	☐

Please tell us how we can improve this book:

Thank you for your responses. May we contact you? ☐ Yes ☐ No

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

Name _____ Address _____

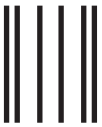
Company or Organization _____

Phone No. _____

Readers' Comments — We'd Like to Hear from You

Cut or Fold
Along Line

Fold and Tape **Please do not staple** Fold and Tape



NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

IBM Corporation
Software Reengineering
Department G7IA / Bldg 503
P.O. Box 12195
Research Triangle Park, NC
27709-9990

Fold and Tape **Please do not staple** Fold and Tape

Cut or Fold
Along Line

Printed in U.S.A.