Developer API Guide

PayLINK Developer API Guide

Version 2.1.253.3 Build Z
Released October 06, 2014

Table of Contents

Changes and Modifications

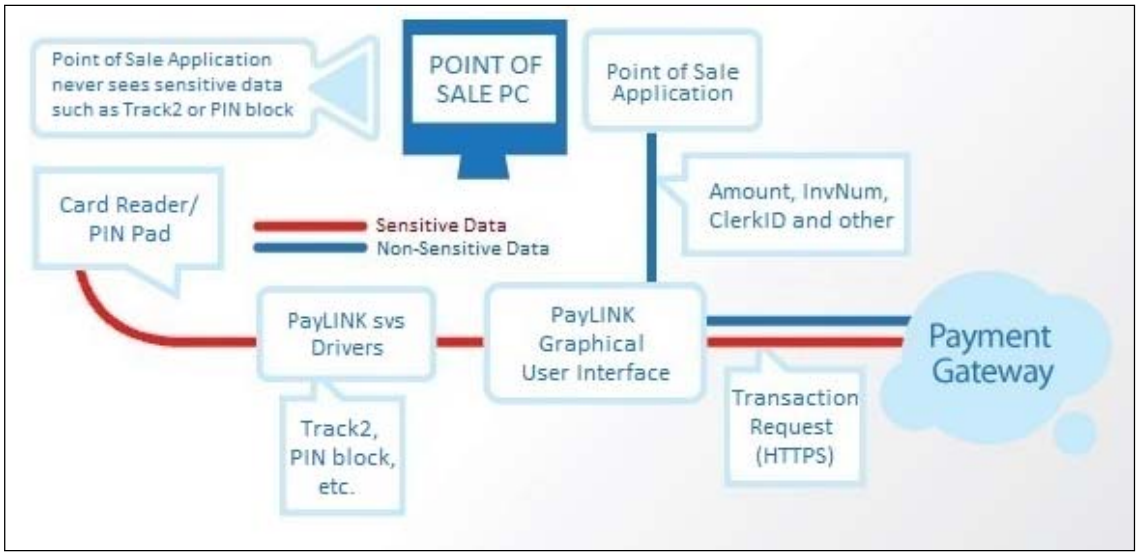
The table below lists changes made to the PayLINK Developer API Guide:

Version	Changes/Modifications	Pages
1.0	Document launch in new format.	All
2.1.253	Update to include Chapter 4 and re-organize Appendix	
2.1.253.1	Corrected SkipSignature element description.	p. 13
2.1.253.2	Corrected SkipSignature element description.	p. 13
2.1.253.2	Corrected Username in File Drop Method	p.10
2.1.253.2	Transaction Category Code aka billing code information added.	p.13
2.1.253.2	Fixed supported Browsers and OS.	p.7
2.1.253.2 Build Z	Service Fee Support added	p.15

Chapter 1. Overview

As card associations move towards stricter regulations for transaction security, the rising costs and complexity tied to standards compliance are a concern for both merchants and integrators. Any Point-of-Sales (POS) system that processes sensitive payment information requires PCI PA-DSS/PABP certification. The certification process is tedious and costly to maintain. If a POS system changes, it must go through the recertification process, increasing cost and time to market.

PayLINK is a client application that integrates into the transaction process by handling the collection and transmission of sensitive payment information, thereby offloading the certification responsibility from merchants and integrators. The POS system sends out order information to a locally installed, browser-based virtual terminal (VT) that collects sensitive payment data, drives POS hardware (e.g., card reader), submits the transaction to a server for processing, and returns the transaction results back to the POS system.



PayLINK comes with an executable installer file that integrators may incorporate into their POS system installation. To ease integration for POS developers, PayLINK provides a Windows object library to include in the POS application. The object library is accessible using Component Object Model (COM) or .NET technology.

When invoked, PayLINK displays a basic frame that encapsulates the browser, fetching and rendering the VT page. The object window is configurable, and integrators can brand the user interface using cascading style sheets. Except in the case of Silent Mode, the VT object window holds focus, floating on top of all other running applications.

Chapter 2. Integration Process

2.1. Requirements

Operating Systems

PayLINK has PA-DSS certification for the following operating systems:

- Windows Vista Business 32-bit
- Windows Vista Business 64-bit
- Windows 7 Professional 32-bit
- Windows 7 Professional 64-bit
- Windows 8 Professional 64-bit
- Server 2008
- Server 2012

> Failure to use a supported operating system results in noncompliance with PA-DSS certification.

Browsers

BridgePay only guarantees full support for PayLINK using the Internet Explorer browser version 9 or greater.

2.1.1.1. Additional Software Requirements

- .NET Framework 3.5 **and** .NET Framework 4.0 (or later)
- Microsoft POS for .NET 1.12
- Internet Explorer 9 or above
- All current Microsoft updates

Antivirus Software Interference

Before installing the PayLINK application, you **must** disable any antivirus scanning software on your machine. Antivirus scanning software may prevent the installation package's custom actions from properly executing. For example, during PayLINK installation, Norton AntiVirus Auto-Protect warns that a potentially dangerous script may execute during installation. The script must be able to run in order to complete installation successfully.

2.2. Getting Started

Before integrating, make sure you've properly installed, configured, and familiarized yourself with the PayLINK application.

We recommend performing the following steps before integration:

1. Install PayLINK using the **PayLINK User Guide**.
2. Along with the PayLINK User Guide, use the **PA-DSS Implementation Guide** to configure the PayLINK application.
3. Verify that PayLINK is functioning properly by putting it into **Demo Mode** and running a test credit sale transaction with one of the sample projects from the **Developer Center** at **http://www.bridgepaynetwork.com/developerCenter.html**.
4. If PayLINK returns a bad response for the test transaction, determine the source of the failure before continuing. If you receive a good response, proceed with integration.
5. Please pay special attention to any special notes in the **readme** file of the PayLINK root folder.

> ⓘ **Configuration**: The PayLINK settings application handles configuration.

> ⓘ **Registration**: The hosted page control requires the input of a serial number to validate against the registration server. The PayLINK application directly handles this serial number.

> ⓘ **Error Handling**: In addition to the payment page and supported hardware having timeouts, PayLINK records errors and debug information into log files in the "C:\Temp\" folder to assist in troubleshooting.

2.3. Integration Options

PayLINK currently offers three integration options: DLL, OCX, and a **File Drop Method**. Please note that the enumerations, properties and methods contained in this document are for integration with the DLL.

The OCX has identical classes, methods and properties as the DLL, except for the omission of the PaymentRequest.CustomFields. You must also postfix all classes within PayLINK.ocx with "UM" to distinguish them from the DLL classes. For example: **PaymentRequest** becomes **PaymentRequestUM**. See the **Developer Center** for a VB6 sample project of OCX integration.

> ⓘ Remember that the application integrating to PayLINK should **never** touch, collect, transmit, or store the account holder's actual card information.

> ⓘ The integrating application must always pass a unique identifier for each POS user or cashier.

2.4. Transaction Flow Summary

The following list summarizes the steps involved in processing a transaction:

1. The POS system collects order information and determines the customer will pay with one of the supported tender types.
2. The POS system invokes PayLINK's **ShowPage** method.
3. The PayLINK payment page loads and prompts user to enter customer and payment data.
4. After collecting all transaction information, PayLINK transmits the data to the server.
5. After receiving a response from the host, the payment page returns the processing results back to the POS system.
6. The POS system analyzes response data.

Chapter 3. PayLINK Integration

3.1. ShowPage Method

PayLINK has one method: **ShowPage**. The POS system populates any required and optional properties of the input object, **PaymentRequest**, before calling the ShowPage method. PayLINK parses data in the PaymentRequest object and displays a payment page with the appropriate form for gathering any needed payment information from the cashier and hardware.

PayLINK submits the information to the server. After processing completes, PayLINK returns the **ShowPageResult** and **PaymentResponse**. Only after validating the ShowPageResult object should you analyze the processing result information in the PaymentResponse.

3.1.1. Program Input Object

3.1.1.1. PaymentRequest

The following table contains descriptions for the properties of the ShowPage input object. Properties are optional unless otherwise indicated.

Property	Description
TenderType	Required. Valid values are: • **UNKNOWN** – 0* • **CREDIT** – 1 • **DEBIT** – 2 • **CHECK** – 3 • **EBT_FOODSTAMP** – 4 • **EBT_CASHBENEFIT** – 5 • **GIFT** – 6 • **LOYALTY** – 7* • **CASH** – 8*
TransType	Required. Valid values are: • **AUTH** – 1: Verifies/authorizes a payment amount on a check or credit card. • **SALE** – 2: Makes a purchase with a credit, debit, gift, EBT food stamp, EBT cash benefit, check, or unknown payment type. • **RETURN** – 3: Returns a credit, debit, gift EBT food stamp, or unknown payment. • **VOID** – 4**: Removes a credit or gift transaction from an unsettled batch.

Property	Description
	• **POSTAUTH** – 5: Completes a credit Auth transaction. • **FORCEAUTH** – 6: Forces a credit, EBT food stamp, or EBT cash benefit transaction into open batch. • **CAPTURE** – 7*: Reserved for future use. • **REPEATSALE** – 8: Performs a repeat sale using the RefNum from a previously processed credit card. • **CAPTUREALL** – 9**: Performs a settlement or batch close. • **ADJUST** – 10**: Adjusts the tip amount of a previously processed credit transaction. • **INQUIRE** – 11**: Performs a balance inquiry on a gift, EBT food stamp, or EBT cash benefit card. • **ACTIVATE** – 12: Activates a gift card. • **DEACTIVATE** – 13: Deactivates an active gift card account. • **RELOAD** – 14: Adds value to a gift card account. • **STATUS_CHECK –** 21*: Reserved for future use. • **SETUP** – 22*: Reserved for future use. • **INIT** – 23*: Reserved for future use. • **TOKENADD** – 24**: Generates a token for a credit or gift card. See **Card Vaulting** on page 31 for additional information. • **TOKENMOD** – 25**: Modifies a token for a credit or gift card. See **Card Vaulting** on page 31 for additional information. • **REVERSAL** – 26: Reverses a credit or debit transaction.
Amount	Required. Total transaction amount (includes subtotal, cash back, tax, and tip).
CashBackAmt	Total amount cash back.
ClerkID	Required. Employee/Clerk ID.
CssPath	Style sheet path. Allows developers to specify their own CSS file to customize the branding of the payment page.
CustomFields	Reserved for future use.
Username	User-name. Applies when supporting multiple server accounts.
Password	Token generated from the PayLINK application.
CustomerName	Customer name.cxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Zip	Customer billing ZIP/postal code.
TipAmt	Tip amount.
TaxAmt	Tax amount.

Property	Description
Street	Billing street address.
SurchargeAmt	Reserved for future use.
ServerID	Reserved for future use.
AutoSubmit	Enables/disables AutoSubmit feature that automatically sends the payment for processing as soon as PayLINK has all necessary information collected. Valid values are: • **True**: Enables AutoSubmit. • **False**: Disables AutoSubmit.
PONum	Purchase order number.
OrigRefNum	Original reference number. Used for follow-on transactions (e.g., Void, Reversal).
MiscAmt	Reserved for future use.
Misc2Amt	Reserved for future use.
Misc1Amt1	Reserved for future use.
MerchantKey	Reserved for future use.
InvNum	POS system invoice/tracking number.
AuthCode	Auth Code obtained via voice authorization from the payment host.
ExtData	Optional unless otherwise indicated. Extended data in XML format. Valid values are: • **<PLWindowSize>***WxH***</PLWindowSize>** Allows you to force the PayLINK transaction window size. For example: 1024x768, 0x0 (fullscreen). ⓘ Window size may range from 800x600 to fullscreen (minus the height of the Start bar). • **<PLReturnData>***ReturnData***</PLReturnData>** Forces what input fields are visible for credit returns. This tag focuses on the display of RefNum and credit card information on the payment page**.** Valid values are **refnum** and **ccinfo**. - If the value is **refnum**, only the RefNum field displays on the payment page. - If the value is **ccinfo**, the PayLINK ignores the RefNum, even if the system provides it. - If the tag is blank or not submitted, and the POS system does not provide a RefNum, both fields display on the payment page. The user must enter a RefNum or credit card information to process the transaction.

Property	Description
	- If this tag is blank or not submitted, and the POS system provides a RefNum, PayLINK uses the RefNum and it displays on the payment page. • **<CardVault>** Required for **TokenMod** transactions. See **Card Vaulting** on page 31 for more information. **<CustomerPaymentInfoKey>***X***</CustomerPaymentInfoKey>** **<Token>***1234567890***</Token>** **<ExpDate>***MMYY***</ExpDate>** **</CardVault>** • **<CardVault>** Required for **Sale**/**Auth**/**ForceAuth**/**Return** transactions with a tokenization flag. See **Card Vaulting** on page 31 for more information. **<CustomerPaymentInfoKey>***X***</CustomerPaymentInfoKey>** **<Transaction>**Read**</Transaction>** **<Token>***1234567890***</Token>** **<ExpDate>***MMYY***</ExpDate>** **</CardVault>** • **<Silent>***True***</Silent>** When you pass the enabled silent tag, the transaction will proceed in **Silent Mode**. Silent Mode: - Prevents payment page from attempting to take over the screen. - Displays payment page with an 800x600 resolution to allow for any Alt+Tab emergencies. - Minimizes payment page at start. - Forces AutoSubmit for the transaction. - Forces all input fields (e.g., last4, AVS, CVV) to be optional. - Disables pop-ups. - Sets transaction to fail from a bad expiration date (result code 60). - Sets transaction to fail from a bad MSR read (result code 60). - Sets transaction to fail from a bad PIN read (result code 60). - Disables signature capture. The following transactions always run in **Silent Mode**: - **Auth** with token passed. - **Sale** with token passed. - **Return** with token passed. - **ForceAuth** with token passed. Token transactions do not support **Demo Mode**.

Property	Description
	• **<SignatureText>***Text***</SignatureText>** Allows the integrator to submit custom text to be displayed on the top portion of the Signature capture screen. A new line can be forced by submitting '\n' in the text string. This only has an effect on the display of the Signature screen on the device, there are no functional effects. > Only supported on the Ingenico iSC350, i6780, and the Verifone MX8XX series. • **<SignatureYesNo>***True***</SignatureYesNo>** When combined with the SignatureText tag this allows the Integrator to ask the customer a Yes or No question and gather a signature. The answer will be returned in the SignatureResponse\AnswerData tag. • **<SkipSignature>***True*|*False***</SkipSignature>** This tag allows for finer control over when a signature is captured on Credit Card transactions. - If set to 'true' a signature will not be captured. - If set to 'false' a signature will be captured. This will override the skip signature amount configured in the Settings Application. This will also force a Signature to be captured when processing a card on file transaction. Normally a card on file transaction would not capture a signature (as they are card not present transactions). - Does not override the Silent tag - in the situation where both Silent and SkipSignature/False are submitted a signature will not be captured. > It is very important to note that if a retail merchant either skips signature capture or uses sig cap with card on file they will be responsible for verifying the signature is valid. During normal processing PayLink will display the signature to the cashier so it can be compared with the signature on the back of the card as required by PCI. In either of the cases specified here that will not occur so the POS must perform this step. • **<SignatureOnly>***True***</SignatureOnly>** When submitted to a Credit Card Sale, this will result in ONLY a Signature being collected and returned. The CC transaction will not occur, the signature will not be uploaded to the server. • **<TransCategory>***B/H/I/R***</TransCategory>** For My bridge pay integrations only. Passing the transcode in the ExtData tells the gateway what category transaction is being passed. The accepted values are B for billing, H for health, I for install and R for recurring.

Property	Description
	• For customers using the Bridge com gateway, PayLink has the ability to send service fees. A service fee sends two separate transactions to the bank under different merchant accounts. The **MerchantCode** is a string value with the merchant code of the merchant collecting the service fee. The **MerchantAccountCode** is a string value with the merchant account code of the merchant collecting the service fee. The fee amount is the amount in decimal format of the service fee. If the first transaction fails you get a decline on both transaction. If the second fails the first is reversed and you will receive a decline on both. You will receive a single response just like any other transaction. **<ServiceFee>** **<MerchantCode>** Bridge Com merchant Code**</MerchantCode>** **<MerchantAccountCode>** Bridge Com merchant account Code**</MerchantAccountCode>** **<FeeAmount>**1.00**</FeeAmount>** **</ServiceFee>** • For customers using the Bridge com gateway PayLink has the ability to send convenience fees. A convenience fee is added to the whole transaction amount on the gateway and is sent through as one transaction. The whole amount then goes to the merchant. The fee amount is the amount in decimal format of the convenience fee. **<ConvenienceFee>**1.00**</ConvenienceFee>**

* Not currently supported.

** Transactions with this TransType require the submission of an **Amount** of $0.00.

3.1.2. Program Output Object

3.1.2.1. ShowPageResult

The following table contains descriptions of the ShowPageResult output object properties.

Property	Description
ShowPageResultCode	**OK:** Payment page properly displayed. Proceed to transaction results.

Property	Description
	CANCEL: User clicked the cancel button on PayLINK before transaction was sent to processor. **ERROR:** Payment page did not properly display, so the transaction did not process.
Message	Accompanying string that describes the result of the transaction or nature of the error (e.g., "Invalid Account Selection"). For descriptions of the most common errors, see **ShowPage Method Errors** on page 31.

The following is an excerpt from a sample project demonstrating the ShowPageResult:

```
Select (resultOfThePage.Code)
    Case ShowPageResultCode.CANCEL
        MessageBox.Show("User Canceled")
    Case ShowPageResultCode.OK
        Dim transactionResult As PaymentResponse = payLinkPage.PaymentResponse
        MessageBox.Show("Result of Transaction: " + transactionResult.ResultTxt)
    Case ShowPageResultCode.ERROR
        MessageBox.Show(resultOfThePage.Msg, "Error Processing Payment", MessageBoxButtons.OK)
    Case Else
        MessageBox.Show("Unknown Result from page.")
End Select
```

3.1.2.2. PaymentResponse

The following table contains descriptions for the PaymentRequest output object properties.

> You should only look at the results in the PaymentResponse if the ShowPageResultCode has a value of **OK**.

Shaded rows indicate values not currently supported.

Property	Description
AuthCode	Transaction authorization code from the payment processor. This value is an approval code for approved transactions or an error code/message for declined transactions.
ApprovedAmount	The actual amount approved by host. This may differ from the requested amount.

Property	Description
AvsResponse	AVS response. See **AVS Response Code** on page 29 for a list of possible responses.
BogusAccountNum	Partially masked card number. The first 6 and last 4 digits of the card number are present with zeros in between. This card number will not pass the mod 10 check.
CardType	Displays the card type (determined by BIN range).
CvResponse	The CV response code. See **CVV Response Code** on page 30 for a list of possible responses.
HostCode	Payment processing host reference number.
HostResponse	Payment processing host response.
Message	Host or gateway message.
Message1	Reserved for future use.
Message2	Reserved for future use.
RefNum	Gateway reference/token number. Used for follow-on transactions (i.e., Void, Reversal).
RawResponse	Gateway raw response.
RemainingBalance	Remaining balance on gift or prepaid card.
RequestedAmount	Original requested amount of the transaction.
ResultCode	Result code of the transaction. See **Result Codes** on page 29 for more information.
ResultTxt	Details from the processor or payment gateway about the transaction result.
Timestamp	Time and date of the transaction.
ExtData	Extended data in XML format. Data may come from PayLINK and/or the host and may have nested values. Valid values are: • **<PLCardPresent>***CardPresence***</PLCardPresent>** Indicates a cardpresent transaction. • **<PLEntryMode>***EntryMode***</PLEntryMode>** Entry mode of cardholder data. • **<PLPONumber>***PONumber***</PLPONumber>** Purchase Order number. • **<PLCustCode>***CustomerCode***</PLCustCode>** Customer code. • **<PLNameOnCard>***Name***</PLNameOnCard>** Customer name. • **<PLStreet>***Street***</PLStreet>** The customer's street address. • **<PLZip>***ZIP***</PLZip>** Customer's ZIP code.

Property	Description
	• **<CardVaultResponse>** Returned after a successful card vault transaction. See **Card Vaulting** on page 31 for more information. **<Token>**1234567890123456**</Token>** **<CustomerPaymentInfoKey>**X**</CustomerPaymentInfoKey>** **<ExpDate>**MMYY**</ExpDate>** **</CardVaultResponse>** • **<SignatureResponse>** Describes the result of a signature upload and filename of the signature image. See • **Signature Capture** Response on page 34 for additional information. **<Result>***ResultCode***</Result>** Result Code of signature upload. **<RespMSG>***ResponseMessage***</RespMSG>** Details about signature upload result. **<SigFile>***SignatureImageFileName.bmp***<SigFile>** Filename of signature image. **</SignatureResponse>** • **Net_Count**=*X*,**Net_Amount**=*X*,**Settle_DT**=*YYYY-MM-DD HH:MM:SS* Additional information with details of the settled batch if you perform a batch close. - **Net_Count**: Total number of transactions in the closed batch. - **Net_Amount**: Net amount of transactions in the closed batch. - **Settle_DT**: The date and time of the requested batch close. This batch response information is not an XML string.

3.2. File Drop Method

Using the file drop integration method requires formatting the transaction data using the **PayLINK XML Data Format** described below. Create a new file with an extension of ".in1", insert the XML-formatted transaction data, and drop it into the PayLINK directory.

> ⓘ The actual file name has no impact on the process, and you may use whatever you want; only the file extension matters.

Once PayLINK picks up the file, the file extension changes to ".in2", indicating the file is processing. PayLINK creates another file of the same name with an extension of ".out" when processing completes. The completion file contains an XML response for the transaction.

The ".out" file contains an XML response for all processed transactions, even if there is an error in the processing.

If there are multiple payment files in the directory, PayLINK processes them one at a time in no particular order.

3.2.1. PayLINK XML Data Format

The DLL interface is the basis of the XML format, so you should be familiar with the properties and basic use of the DLL before continuing. Requests contain a PaymentRequest root tag and nested data elements containing DLL properties. The response XML has a similar framework containing all of the standard response data.

3.2.1.1. Examples

SimpleSaleTest.in1

```
<PaymentRequest>
    <Amount>3.00</Amount>
    <TenderType>CREDIT</TenderType>
    <TransType>SALE</TransType>
    <ClerkID>12313</ClerkID>
</PaymentRequest>
```

SimpleSaleTest.out

```
<ShowPageResponse>
    <ShowPageResult>
        <ShowPageResultCode>OK</ShowPageResultCode>
        <ShowPageResultMsg>OK</ShowPageResultMsg>
    </ShowPageResult>
    <PaymentResponse>
        <ResultCode>0</ResultCode>
        <ResultTxt>Approved</ResultTxt>
        <Message>APPROVAL DEMO-4</Message>
        <Message1/>
        <Message2/>
        <RefNum>20514</RefNum>
        <AuthCode>DEMO-4</AuthCode>
        <AvsResponse>S</AvsResponse>
        <CvResponse>U</CvResponse>
        <Timestamp>2012-06-27T14:05:16</Timestamp>
        <HostCode>020514020514</HostCode>
        <RequestedAmount>3.00</RequestedAmount>
        <ApprovedAmount>3.00</ApprovedAmount>
        <RemainingBalance/>
        <HostResponse>APPROVAL DEMO-4</HostResponse>
        <BogusAccountNum>4012000088881</BogusAccountNum>
```

```
        <CardType>Visa</CardType>
        <IsCommercialCard>False</IsCommercialCard>
        <ExtData>
            CardType=Visa
          <PLNameOnCard>Some Dude</PLNameOnCard>
          <PLEntryMode>MANUAL</PLEntryMode>
          <PLStreet>123 Any Street</PLStreet>
          <PLZip>31322</PLZip>
          <PLCardPresent>false</PLCardPresent>
          <PLRefNum/>
          <PLPONum/>
          <PLCustCode/>
          <PLTransTender>Credit</PLTransTender>
          <SubTotalAmt>3.00</SubTotalAmt>
           <TransCategory>b</TransCategory>
        </ExtData>
        <RawResponse>
          <![CDATA[<?xml version="1.0" encoding="utf-8"?><Response
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xmlns:xsd="http://www.w3.org/2001/XMLSchema"><Result>0</Result><RespMSG>Approved</RespMS
          G><Message>APPROVAL DEMO-4</Message><AuthCode>DEMO-
          4</AuthCode><PNRef>20514</PNRef><HostCode>020514020514</HostCode><GetAVSResult>S</Get
          AVSResult><GetAVSResultTXT>Service Not
          Supported</GetAVSResultTXT><GetStreetMatchTXT>Service Not
          available</GetStreetMatchTXT><GetZipMatchTXT>Service Not
          available</GetZipMatchTXT><GetCVResult>U</GetCVResult><GetCVResultTXT>Service Not
          Available</GetCVResultTXT><ExtData>CardType=Visa</ExtData></Response>]]>
        </RawResponse>
    </PaymentResponse>
</ShowPageResponse>
```

BadTenderTest.in1

```
<PaymentRequest>
    <Amount>3.00</Amount>
    <TenderType>CauseAnError</TenderType>
    <TransType>SALE</TransType>
    <ClerkID>12313</ClerkID>
</PaymentRequest>
```

BadTenderTest.out

```
<ShowPageResponse>
    <ShowPageResult>
    <ShowPageResultCode>ERROR</ShowPageResultCode>
    <ShowPageResultMsg>ERROR - Invalid TenderType or TransType</ShowPageResultMsg>
    </ShowPageResult>
</ShowPageResponse>
```

CancelTest.in1

```
<PaymentRequest>
    <Amount>3.00</Amount>
    <TenderType>CREDIT</TenderType>
    <TransType>SALE</TransType>
    <ClerkID>12313</ClerkID>
</PaymentRequest>
```

CancelTest.out

```
<ShowPageResponse>
    <ShowPageResult>
        <ShowPageResultCode>CANCEL</ShowPageResultCode>
        <ShowPageResultMsg>CANCEL</ShowPageResultMsg>
    </ShowPageResult>
</ShowPageResponse>
```

3.2.2. PayLinkHW.dll Support

The File Drop system also supports all commands that are supported by the PayLinkHW.dll interface. Simply write the XML that would normally be submitted to the PayLinkHW.dll to the file instead of the Payment XML described above. The output file will contain the same response XML as if you had been processing directly to the dl.

Chapter 4. Hardware Integration

PayLINK gives the integrator the ability to check on the status of what the hardware is currently doing as well as interact with it on a limited basis. This section describes the different "direct" interactions the integrator can have with the hardware.

4.1. PayLinkHW.dll

PayLinkHW.dll is located in the PayLINK\Bin directory and automatically registered in the systems GAC for easy access. This dll is responsible for allowing merchants who need an extended and highly customized hardware interaction to indirectly communicate with PayLINK's hardware. This interface has a single class with a single method. The PayLINKHW.ProcessHWCommand accepts and XML formatted string and returns an XML formatted string.

All of the commands below will only be executed if the PayLINK is either waiting for an MSR swipe, or not doing anything at all. If PayLINK is busy and can't run the command an appropriate error message will be returned.

- For security reasons this dll does not directly access PayLINK's hardware. It interfaces with PayLINK's internal hardware system allowing us to selectively expose certain functionality.
- It is not possible to access PCI sensitive information from this dll.
- These commands are only supported on the Ingenico i6780 and the Verifone MX8XX series. The Ingenico iSC350 has support for only section 4.1.4, there are plans to expand this support in the future.

4.1.1. Request Format

PayLINKHWReq – Root tag, contains 1 command tag and any number of Input tags.

Command – This tag contains the descriptor of the command to execute.

Input – This tag contains any input that a command may require. The Name attribute's value should contain the descriptor for the type of data being passed.

Example of empty XML

```
<PayLINKHWReq>
    <Command/>
    <Input Name="" />
</PayLINKHWReq>
```

4.1.2. Response Format

PayLINKHWResp – Root tag, contains 1 ResultCode tag and optional Message, OutPut tags.

ResultCode – Contains a number indicating Success/failure. 0 indicates success, any other value is an error

Message – This optional tag will contain any important text. An error message, or button selection for example.

OutPut – Some requests for input have more than one return and require a more advanced output. The Source attribute will indicate where the output is from. It will contain a variety of sub tags on a per command basis.

Example of empty XML

```
<PayLinkHWXMLResp>
    <ResultCode/>
    <Message/>
    <OutPut Source=""/>
</PayLinkHWXMLResp>
```

4.1.3. Input Name Values

Text – Contains a string value to be use by the command.

Multiples of this type are valid for some commands.

EndScreen – For commands that request information from the user this indicates what screen to display on the device after the information has been entered. Current supported values are 'Wait' and 'None'. Wait will cause the screen to display 'Please Wait'. None will cause the screen to not change after entry. This is intended to reduce processing time and screen flicker when chaining multiple commands together.

Only one input of this type is allowed.

SigType – Indicates the type of signature screen to use. Detailed in the Signature capture section below

Only one input of this type is allowed.

4.1.4. Line Display Commands

The Line Display allows the integrator to display a scrolling list of items along with two lines of header text. The LD is called while waiting for MSR and the customer swipes their card the LD screen will remain in place and display a message in the header indicating that the swipe was accepted. After 5 Seconds the original header text will resume. If new header text was submitted during this time the new text will be displayed.

Clearing the Item data will only clear data displayed via the 'AddItemToLD' command. To clear the header or column text submit their respective commands with an empty string. Submitting a Transaction Complete (cash) transaction to the PayLINK Payment interface will clear all LD data and reset the device back to the 'Please Wait' screen. If CST is enabled the device will cycle back to 'Please Swipe'. As items are added they will be added to the end of the list, when the bottom of the screen is reached items will be either be removed from the top of the list or scrolled with a scroll bar. This is controlled by the devices support for a scroll bar.

UpdateLDHeader – Updates the Header value on the LD screen to show the specified text. Triggers the LD Screen to be displayed if it's not currently on the screen.

Input: Text – Single entry

UpdateLDColumn – Updates the Column header value on the LD screen to show the specified text. Triggers the LD Screen to be displayed if it's not currently on the screen.

Input: Text – Single entry

AddItemToLD – Add an item to the bottom of the LD. Does not trigger the LD Screen to be displayed.

Input: Text – Single or multiple entry

ClearItemData – Clears all Item data currently displayed. Does not trigger the LD Screen to be displayed.

4.1.5. Signature Capture Commands

PayLINK support 3 different signature capture screens with 2 of them being customizable. These screens can be displayed and used to gather a signature. This signature will be returned to the POS and PayLINK will not maintain any record of it.

GetSignature – Prompt the customer for a signature using the specified screen and optional data.

Input: EndScreen, SigType, Text – Single entry

Possible SigTypes:

Basic – There will be no custom prompting, this will use the standard PayLINK signature capture screen.

Text - This will display the Text submitted on the screen.

QuestionYesNo - This will display the text submitted on the screen as well as Yes/No buttons. The option selected will be returned to the integrator. If the customer signs and accepts without making a selection they will be prompted with the RxYesNo command to explicitly request an answer.

Output: On approval the OutPut tag will contain a Source attribute of 'Signature', a 'Filename' tag and optionally an 'AnswerData' tag.

Filename contains the file name of the signature collected. This file is located in the folder <PayLinkInstall>wwwroot\Svt2\img\Receipts. This is the same behavior as if the signature was collected via the payments interface.

AnswerData tag will be present if the SigType was set to 'QuestionYesNo'. It will contain text indicating which option was selected, one of the following.

Yes | No | Cancel

4.1.6. Form Display Commands

The main advantage of having a multi-lane device is using it to interact with the customer to give and collect information. The commands here contain some highly specialized input requests from the customer as well as some informational display.

RxRelation – Prompt the customer for their relationship to the patient.

Input: EndScreen

Output: On success the Message tag will contain one of the following.

Patient | Spouse | Child | Power of Attorney | Other | Parent | Care Giver

RxYesNo – Prompt the customer with the text provided, and allow the customer to select Yes or No.

Input: EndScreen, Text – Single entry

Output: On success the Message tag will contain one of the follow.

Yes | No | Cancel

RxCounsel - The following steps will occur for each text input item:

- A text will be displayed at the top of the screen, and the Customer will be presented with 5 check box options: How to use, Common uses, Warnings and precautions, Side effects, Speak to pharmacist. The customer will have 3 buttons options, Back, Next, Skip Counsel.

- The customer can select as many checkboxes as they need.

- Pressing Next will move the screen to the next Text item. Pressing Next on the last screen will end the session.

- Pressing Back will move the screen to the previous text item.

- Pressing Skip will end the session.

- Once the session has ended the last buttons pressed and the status of all check boxes will be returned.

> When the custom presses Skip, box status is still returned.

Input: EndScreen, Text – Single or Multiple entry

Output: On success the Message tag will contain one of the following.

Next | Skip

The OutPut tags will contain a Source attribute of 'Counsel', a 'Text' tag and a 'Selected' tag.

Text will contain the text of the input this OutPut is for.

Selected will contain a listing of the checkboxes check for this text item. Possible Selected items are listed here.

0 | HowTo | CommonUse | Warnings | SideEffects | Pharmacist

> Note that 0 is special, and will only appear when no selection has been made. Others can appear in any combination. See example in section 4.1.7

SetCustomScreen - Display a specified screen on the device. No events from the screen will be propagated back to the integrator. They will only receive a success/fail indicating if the screen was displayed.

It is the integrators responsibility to ensure that the custom form they want to display is already on the device. To help with this PayLink will now attempt to download any file in that devices 'custom' forms folder when downloading its own files. For example, if you wanted to download 'text.icg' to the 6780 you would place the file in this location.

<PayLinkInstallDir>\Bin\Forms\Ing6780\text.icg

Then go into the settings application and download the forms per normal operations. Be sure to include any files required to display this screen. Do not replicate screen or file names with PayLink as this can cause undesired behavior.

4.1.7. Selected Examples

Request 1:

```
<PayLINKHWReq>
    <Command>UpdateLDHeader</Command>
    <Input Name="Text">Some header text·</Input>
</PayLINKHWReq>
```

Response 1:

```
<PayLINKHWXMLResp>
     <ResultCode>0</ResultCode>
</PayLINKHWXMLResp>
```

Request 2:

```
<PayLINKHWReq>
    <Command>RxCounsel</Command>
```

```
    <Input Name="EndScreen">End Screen</Input>
    <Input Name="Text">Test Data</Input>
    <Input Name="Text">Looooooong test data. Wow look how long this is! Why would you ever use
something this long? This is probably to looooooong.</Input>
    <Input Name="Text">Another Test data, with Cookies this time.</Input>
</PayLINKHWReq>
```

Response 2:

```
<PayLINKHWXMLResp>
    <ResultCode>0</ResultCode>
    <Message>Next</Message>
    <OutPut Source="Counsel">
        <Text>Test Data</Text>
        <Selected>HowTo, CommonUse, SideEffects </Selected>
    </OutPut>
    <OutPut Source="Counsel">
        <Text>Looooooong test data. Wow look how long this is! Why would you ever use something this
long? This is probably to looooooong.</Text>
        <Selected>Pharmacist</Selected>
    </OutPut>
    <OutPut Source="Counsel">
        <Text>Other Test data, with Cookies this time.</Text>
        <Selected>0</Selected>
    </OutPut>
</PayLINKHWXMLResp>
```

4.1.8. Common Result Codes

0 – Success – No errors were reported by the hardware system

1 – Error occurred parsing the request

2 – Error processing the request

3 – Error communicating with the service

4 – Error Reported by the service, Message will contain text from service.

5 – Service is not running

12 – Service Declined the request, Message will contain the reason.

4.2. Hardware Event Notification

PayLINK has an internal service that reports important events in the hardware process. This service populates the event information of any properly functioning hardware into the registry. The registry will update at a maximum of one event per 100 milliseconds. Normally, however, consecutive events occur at a much lower frequency.

- **Regkey**: HKLM\Software\PayLINK\Client
- **Value**: LastHWEvent
- **Data**: <Device>|<event>|<text> (e.g., “MSR|Start|”)

4.2.1. Device

This parameter indicates the type of device for the current event.

Device	Description
MSR	Magnetic Stripe Reader (keyboard wedge is not supported).
PIN	PIN pad.
CHKIMG*	Check image reader.
SIGCAP*	Signature capture.
MANUALCARD*	Encrypted manual card entry.
MICR*	MICR reader.
DEVICEDISPLAY*	The display on a multilane device.

* This device is not currently supported.

4.2.2. Event

This parameter indicates the type of event.

Event	Description
Start	The device is waiting for input from the user.
Stop	The device is no longer waiting for input from user.
GoodData	The device received valid data from the user.
BadData	The device received invalid data from the user.
Canceled	The user canceled input.
TimeOut	Data entry has timed out.
Error	An error has occurred. Check the message text.

A. Appendix

A.1. Response Values

A.1.1. Result Codes

The following table contains descriptions of the result codes returned in the **ResultCode** response field from PayLINK. Please note that when programmatically validating the result of a transaction, you should use this value instead of any other response message describing the result.

Value	Description
0	Approved.
12	Declined. Review **ResultTxt** field in **PaymentResponse** to determine why the transaction was not processed.
All Others	Not Processed. Review **ResultTxt** field in **PaymentResponse** to determine why the transaction was not processed.

A.1.2. AVS Response Code

The following table contains the possible response values returned for address verification (AVS).

Value	Description
X	Exact: Address and 9-digit ZIP match.
Y	Yes: Address and 5-digit ZIP match.
A	Address: Address matches, but ZIP does not.
Z	5-digit Zip: 5-digit ZIP code matches, but address does not.
W	Whole Zip: 9-digit ZIP matches, but address does not.
N	No: Neither address nor ZIP matches.
U	Unavailable: Address information is not available.
G	Unavailable: Address information not available for international transaction.
R	Retry: System unavailable or timeout.
E	Error: Transaction unintelligible for AVS, or edit error in the message prevents address verification.
S	Not Supported: Issuer doesn't support AVS service.

Value	Description
B	Street Match: Street address matches for international transaction, but the postal code does not. Visa-specific value.
C	Street Address: Street address and postal code are not verified for international transaction. Visa-specific value.
D	Match: Both street address and postal code match for international transaction. Visa-specific value.
I	Not Verified: Address information not verified for international transaction. Visa-specific value.
M	Match: Street address and postal code matches for international transaction. Visa-specific value.
P	Postal Match: Postal code matches for international transaction, but street address doesn't. Visa-specific value.
0	No response sent. The gateway returns this value.
5	Invalid AVS response.

A.1.3. CVV Response Code

The following table contains the possible response values returned for a CVV2/CVC2/CID check.

Value	Description
M	CVV2/CVC2/CID Match.
N	CVV2/CVC2/CID No match.
P	Not processed.
S	Issuer indicates that the CV data should be present on the card, but the merchant has indicated that the CV data is not present on the card.
U	Unknown: Issuer has not certified for CV, or issuer has not provided Visa/MasterCard with the CV encryption keys.
X	Server provider did not respond.

A.2. Errors

A.2.1. ShowPage Method Errors

The following table contains the most common errors for **ShowPageResult**.

Value	Description
Unable to activate license	License is not yet activated.
PaymentRequest is required	Occurs if no payment request sent.
Invalid Account Selection	Occurs if a valid account is not set. This may indicate the UserID/Token is invalid or the default account is not set in PayLINK.
Missing TenderType	TenderType not passed into PayLINK.
Missing TransType	TransType not passed into PayLINK.
ClerkID Required	ClerkID not passed into PayLINK.
Amount Must be $0.00	Amount must be set to 0.00.
Invalid Amount	Invalid amount passed into PayLINK.
Invalid TenderType or TransType	Invalid TenderType or TransType not passed into PayLINK.
Exception.Message	Occurs if there is an unusual/unknown exception.
Invalid Result returned	Unrecognized transaction result returned.
TX Result Missing	Missing transaction result.

> ⓘ The **Message** property of the **Exception** object contains any other unknown or unusual exception messages.

A.3. Additional Feature Information

A.3.1. Card Vaulting

PayLINK supports SecureLINK's card vaulting functionality for credit and gift cards.

A.3.2. Generating a Token

Run a credit or gift transaction using the **TokenAdd** TransType with an **Amount** of $0.00. PayLINK prompts the user to enter information necessary to create a token. Process the transaction response like you would a normal credit sale. See **PaymentResponse Card Vault Data** below for information on the values returned for a successful transaction.

A.3.3. Modifying a Token

Run a credit or gift transaction using the **TokenMod** TransType with an **Amount** of $0.00. PayLINK prompts the user to enter the original **Token**, **CustomerPaymentInfoKey**, and **ExpDate.** Process the transaction response like you would a normal credit sale transaction. See **PaymentResponse Card Vault Data** below for information on the values returned for a successful transaction.

If you pass **all** required fields for processing a TokenMod transaction within ExtData of the request, the data populates into static fields on the PayLINK payment page.

For example:

```
<CardVault>
    <CustomerPaymentInfoKey>72</CustomerPaymentInfoKey>
    <Token>1234567890</Token>
    <ExpDate>1216</ExpDate>
</CardVault>
```

A.3.4. Processing a Sale, Auth, ForceAuth, Return Transaction

Run the credit or gift transaction with the following card vaulting information in the ExtData:

```
<CardVault>
    <CustomerPaymentInfoKey>X</CustomerPaymentInfoKey>
    <Transaction>Read</Transaction>
    <Token>1234567890</Token>
    <ExpDate>MMYY</ExpDate>
</CardVault>
```

See **PaymentResponse Card Vault Data** below for information on the values returned for a successful transaction.

A.3.5. PaymentResponse Card Vault Data

PayLINK returns the following ExtData in the PaymentResponse for any successful card vaulting transaction:

```
<CardVaultResponse>
    <CustomerPaymentInfoKey>X</CustomerPaymentInfoKey>
    <Transaction>Read</Transaction>
    <ExpDate>MMYY</ExpDate>
```

```
</CardVaultResponse>
```

A.3.6. File Drop Delay Notification

For integrators using the File Drop integration, there is a feature that alerts the user of any delayed starts of the PayLINK service. Once File Drop launches, a timer watches for the PayLINK service to run. If PayLINK isn't running after one minute, a prompt notifies the user of the delay.

A.3.7. Customer Selected Tender (CST)

This feature allows the integrator to allow the customer to select what tender should be used for a transaction. This feature must be turned on in the PayLINK setting application before it can be accessed from the API. Once activated the MSR will remain on the "Please Swipe Card" screen between transactions. Once a card has been swiped the device will display "Please Wait" until a transaction is submitted to PayLINK. The transaction and swipe can occur in any order, PayLINK will wait for both before continuing. Once PayLINK has a swipe and a transaction the following steps will occur.

- Look up the Bin on the card to determine if it is Debit enabled.
- If the card is Debit enabled, and the transaction amount is over the Prefer Debit Amount (configured in the Settings application) the customer will be prompted to enter their PIN.
- Otherwise they will be prompted to confirm the transaction amount for Credit card processing.
- In either case if the customer cancels entry they will be given a screen allowing them select what tender they want to use (Credit, Debit, Gift, or EBT Cash/Food).
- If the selection made requires PIN entry they will be prompted for PIN.
- Once a Tender is finalized the PayLINK screen will update and process the transaction normally using the selection made by the customer.
- After the transaction has completed the customer's selection will be returned to the integrator in the PLTransTender tag of the ExtData.

> PayLink will cache swiped card data until a transaction is submitted. It is very important that the POS clear the cache every time it completes a transaction without PayLINK. Failure to do so could result in processing on a previous customer's card. The cache can be cleared by submitting a $0.00 Cash Sale to PL.

If CST is active and the POS submits a tender type this will override any selection that the customer may have made. They will not be re-prompted to enter any information.

A.3.8. Signature Capture Response

If you use signature processing, PayLINK returns additional elements within the ExtData of the PaymentResponse that describe the result of a signature upload and the filename of the signature image:

```
<SignatureResponse>
    <Result>ResultCode</Result> Result Code of signature upload.
    <RespMSG>ResponseMessage</RespMSG> Details about signature upload result.
    <SigFile>SignatureImageFileName.bmp<SigFile> Filename of signature image.
</SignatureResponse>
```

The PayLINK application stores uploaded signature images as bitmap files within the PayLINK directory at "<install dir>\wwwroot\Svt2\img\receipts\". For example, a signature file named "0046f56d-1db2-4670-a3fb-9091ccc69fac.bmp" is located at "C:\Program Files (x86)\PayLINK\wwwroot\Svt2\img\receipts\0046f56d-1db2-4670-a3fb-9091ccc69fac.bmp".

PayLINK also stores an OPOS-formatted, text representation of the signature file within the same directory. This file shares the same name as the bitmap image but has an .opos file extension. For example, "C:\Program Files (x86)\PayLINK\wwwroot\Svt2\img\receipts\0046f56d-1db2-4670-a3fb-9091ccc69fac.opos".

> PayLINK purges stored image and OPOS files after every 10 transactions.

A.4. Terms of Use Agreement

1. **ACKNOWLEDGMENT AND ACCEPTANCE OF AGREEMENT**

   The Terms of Use Agreement "TOU" is provided by BridgePay to you as an end user "USER" of the information obtained from BridgePay, any amendments thereto, and any operating rules or policies that may be published from time to time by BridgePay, all of which are hereby incorporated by reference. The TOU comprises the entire agreement between USER and BridgePay and supersedes any prior agreements pertaining to the subject matter contained herein.

2. **DESCRIPTION OF SPECIFICATIONS AND INFORMATION**

   BridgePay is providing USER with the information concerning the technical requirements for allowing point of sale software to send and receive electronic transaction data to the BridgePay network for authorization and/or settlement. To utilize the Specifications, USER must: (i) provide for USER's own access to the World Wide Web and pay any fees associated with such access, and (ii) provide all equipment necessary for USER to make such connection to the World Wide Web, including a computer, modem and Web browser.

3. **USER'S REGISTRATION OBLIGATIONS**

In consideration of use of the Specifications, USER agrees to: (i) provide true, accurate, current, and complete information about USER as requested on the Registration Form, and (ii) to maintain and update this information to keep it true, accurate, current and complete. This information about a USER shall be referred to as "Registration Data". If any information provided by USER is untrue, inaccurate, not current, or incomplete, BridgePay has the right to terminate USER's access to the Specifications and refuse any and all current or future use of the Specifications.

4. **MODIFICATIONS TO AGREEMENT**

BridgePay may change the TOU from time to time at its sole discretion. Changes to the TOU will be announced and publicly available to all USERs.

5. **MODIFICATIONS TO SPECIFICATIONS**

BridgePay reserves the right to modify or discontinue, temporarily or permanently, the use of any of the Specifications with or without notice to USER. USER agrees that BridgePay shall not be liable to USER or any third party for any modification or discontinuance of a Specification.

6. **USER ACCOUNT, PASSWORD AND SECURITY**

USER will receive a password when registering their company (account) to become a Partner. Upon approval, that password will allow USER access into the Partner Portal. USER is responsible for maintaining the confidentiality of the password and account, and is fully responsible for all activities that occur under USER's password or account. USER agrees to immediately notify BridgePay of any unauthorized use of USER's password or account or any other breach of security.

7. **LICENSE GRANT**

a. Subject to the terms and conditions of this Agreement, BridgePay hereby grants to USER a personal, limited, perpetual, non-exclusive, non-transferable, annual subscription license to make and use the SOFTWARE accompanying this TOU to be installed on CPUs residing on Licensee's premises, solely for Licensee's internal use. BridgePay and its suppliers shall retain title and all ownership rights to the product and this Agreement shall not be construed in any manner as transferring any rights of ownership or license to the SOFTWARE or to the features or information therein, except as specifically stated herein. use and reproduce the following solely to develop, manufacture, test and support the products: (i) in object code form (except as may be agreed by the parties in writing or as otherwise set forth in this Agreement), (ii) the applicable software in object code form, except as may be agreed by the parties in writing or as otherwise set forth in this Agreement, (iii) BridgePay materials which shall include all documentation.

b. Upon the annual anniversary date of the activation of the license, the USER will be responsible the renewal of the subscription of the license either through their RESELLER or directly to BridgePay.

8. **TRADEMARKS**

USER acknowledges that BridgePay owns exclusive rights in the BridgePay trademarks. USER will not use BridgePay as part of any of its product, service, domain or company names and will not take nor authorize any action inconsistent with BridgePay's' exclusive trademark rights during the term of this Agreement or thereafter. Nothing in this Agreement grants USER ownership or any rights in or to use the BridgePay trademarks except in accordance with this license. USER will use a legend on its website and, where commercially feasible, on all printed materials and products bearing the BridgePay trademarks similar to the following: "(USER name) uses the BridgePay™ mark under express license from BridgePay, LLC."

9. **USER OBLIGATIONS**

a. USER shall utilize its BridgePay assigned developer ID in each application utilizing the BridgePay specification

b. USER shall not reverse-engineer, reverse-compile or disassemble any BridgePay software or otherwise attempt to derive the source code to any BridgePay software.

c. USER shall have no right to (i) disclose any BridgePay source code or BridgePay source code documentation to any third party, (ii) use or reproduce any BridgePay source code or BridgePay source code documentation other than as permitted or contemplated by this Agreement. No licenses are granted by BridgePay to USER by implication or estoppels to the BridgePay source code or BridgePay source code documentation.

d. USER shall comply with all applicable card association regulations, applicable federal, state and local statutes and BridgePay required procedures and identified best practices. USER agrees (i) not to use the Specifications for illegal purposes; and (ii) to comply with all applicable laws regarding the transmission of technical data exported from the United States.

10. **DISCLAIMER OF WARRANTIES**

USER expressly agrees that use of the Specifications is at USER's sole risk. The

Specifications are provided on an "as is" basis.

a. BRIDGEPAY EXPRESSLY DISCLAIMS ALL WARRANTIES OF ANY KIND, WHETHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT

b. BRIDGEPAY MAKES NO WARRANTY THAT THE SPECIFICATION WILL MEET USER'S REQUIREMENTS, NOR DOES BRIDGEPAY MAKE ANY WARRANTY AS TO THE RESULTS THAT MAY BE OBTAINED FROM THE USE OF THE SPECIFICATIONS OR AS TO THE ACCURACY OR RELIABILITY OF ANY INFORMATION OBTAINED THROUGH USE OF THE SPECIFICATIONS. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OF CERTAIN WARRANTIES, SO SOME OF THE ABOVE EXCLUSIONS MAY NOT APPLY TO YOU.

11. **TERMINATION BY BridgePay**

USER agrees that BridgePay may terminate USER's password, account or use of the Specifications:

a. If BridgePay determines in its sole discretion that USER has violated or acted inconsistently with the letter or spirit of the TOU;

b. If the USER has violated the rights of BridgePay, or that USER's continued use of the Specifications poses a material threat to the security, stability or ongoing operation of the System or Specifications.

c. BridgePay may terminate this Agreement for cause at any time upon providing not less than ten (10) business day's prior written notice to USER. USER acknowledges and agrees that any termination of access privileges to the Specifications under any provision of the Agreement may be effected without prior notice.

d. BridgePay shall in the event that a license has not been renewed and BridgePay has not received and validated payment from either the USER or the RESELLER within 30 days of the anniversary date of the original activation the deactivate or terminate the usage of the license.

12. **LIMITATION OF LIABILITY**

In no event shall BridgePay be liable to USER for any incidental, consequential or punitive damages related to this Agreement or the use of BridgePay software or specifications. The liability of BridgePay hereunder shall be limited to the fees paid to BridgePay pursuant to this Agreement. USER agrees that BridgePay shall not be liable for any direct, indirect, incidental, special, or consequential damages, resulting from the use or the inability to use the Specifications, including but not limited to, damages for loss of profits, use, data or other intangibles, even if BridgePay has been advised of the possibility of such damages. Some jurisdictions do not allow the limitation or exclusion of liability for incidental or consequential damages so some of the above limitations may not apply to you.

NOTICE: Any notice to USER or to BridgePay shall be made via either e-mail or regular mail. BridgePay may also provide notices of changes to the TOU or other matters by displaying notices to USERs, generally on the Specifications.

13. **SPECIAL DAMAGES**

In no event will BridgePay be liable to the USER, consequential or punitive damages, including but not limited to, lost profits, even if such party knew of the possibility of such damages.

14. **INDEMNIFICATION**

a. USER shall be liable to and shall indemnify and hold BridgePay, its employees, representatives, successors and permitted assigns harmless from and against any and all claims, demands by third parties, losses, liability, cost, damage and expense, including litigation expenses and reasonable attorneys' fees and allocated costs for in-

house legal services, to which BridgePay, its employees, representatives, successors and permitted assigns may be subjected or which it may incur in connection with any claims which arise from or out of or as the result of (i) USER's breach of this Agreement, (ii) the performance by USER of its duties and obligations under this Agreement or (iii) the negligent or willful misconduct of USER, its officers, employees, agents and affiliates in the performance of their duties and obligations under this Agreement.

15. **PROTECTION OF CONFIDENTIAL INFORMATION**

All information of a business nature relating to the BridgePay specification, software, application interfaces, services, processes, merchant and cardholder data, product or programming techniques of either party shall be deemed confidential ("Confidential Information"). This shall not prohibit each party from disclosing such Confidential Information to persons required to have access thereto for the performance of this Agreement; provided, however, that such persons shall be required to keep such Confidential Information confidential to the same standard that the disclosing party is obligated to keep the Confidential Information confidential.

16. **FORCE MAJEURE**

In no event shall BridgePay be liable with respect to the failure of its duties and obligations under this Agreement (other than an obligation to pay money) which is attributable to acts of God, war, terrorism, conditions or events of nature, civil disturbances, work stoppages, equipment failures, power failures, fire or other similar events beyond its control.

17. **GENERAL**

a. The Specifications Agreement and the relationship between USER and BridgePay shall be governed by the laws of the State of Illinois without regard to its conflict of law provisions.

b. The failure of BridgePay to exercise or enforce any right or provision of the TOU shall not constitute a waiver of such right or provision. If any provision of the TOU is found by a court of competent jurisdiction to be invalid, the parties nevertheless agree that the court should endeavor to give effect to the parties' intentions as reflected in the provision, and the other provisions of the TOU remain in full force and effect.

c. USER agrees that regardless of any statute or law to the contrary, any claim or cause of action arising out of or related to use of the Specifications or the Specifications Agreement must be filed within ninety (90) days after such claim or cause of action arose or be forever barred.

18. **SECTION TITLES**

The section titles in the TOU are for convenience only and have no legal or contractual effect.