heidelpay Payment Platform
the complete solution for e-commerce

Whitelabel Integration Basics

Version: 1.0
Date: 2014-07-31
English

Heidelberger Payment GmbH

Vangerowstraße 18
69115 Heidelberg
Germany

info@heidelpay.de

Phone: +49 (0) 6221 65170 - 20
Fax: +49 (0) 6221 65170 - 12

The information in this document is subject to change without notice. Heidelberger Payment GmbH reserves the right to change its products and services at any time to incorporate the latest technological developments. Although this document has been prepared with every precaution to ensure accuracy, Heidelberger Payment GmbH assumes no liability for any errors or omissions, nor for any damages resulting from the application or use of this information.

Heidelberger Payment GmbH welcomes customer input on corrections and suggestions for improvement.

Table of Contents

1 About This Manual

This document describes the **heidelpay Payment Platform**, the **heidelpay CheckOut** and **heidelpay Intelligence Platform**. It provides an overview of their main features and functions. This document is aimed at technical personnel involved in integrating merchants' systems

First time users should read:

- Chapter 2: heidelpay Payment Platform
  This chapter describes the **heidelpay Payment Platform**.
- Chapter 3: heidelpay Payment Platform Interface
  This chapter describes the **heidelpay Payment Platform Interface.**
  The HTTP-POST-Parameter Interface an all input parameter groups are described.
- Chapter 4: Implementing Payment Transactions
  This chapter describes the **hPP** whitelabel payment transaction Interface.
  The simple synchronous workflow is introduced.
- Chapter 5: asynchronous workflow
  This chapter describes the asynchronous workflow by example.

1.1 Conventions

The following conventions are used throughout this manual:

> **!**
>
> This is a safety notice. Safety notices contain important information, typically regarding an unexpected transfer of funds.

> This is a note. Notes contain helpful information.

Brands of the Heidelberger Payment GmbH are shown in bold grey type. For example: **heidelpay CheckOut**.

Parameter names are shown in monospace italic type. For example: *`Merchant Mode`*.

Parameter values are shown in monospace all caps type. For example: `INTEGRATOR_TEST`.

Code examples are shown in monospace type within a grey frame. For example:

```
//If PROCESSING_RESULT is ACK return success URL string
if ($processingResult === 'ACK') {
  echo 'http://www.theexampleshop.tld/success.php';
}
```

1.2 Further Assistance

For technical support, please contact:

- Email: technik@heidelpay.de
- Phone: +49 (0) 6221 65170 - 10

For sales support, please contact:

- Email: sales@heidelpay.de
- Phone:+49 (0) 6221 65170 - 20

For general information, please contact:

- Email: info@heidelpay.de
- Phone: +49 (0) 6221 65170 – 20

2 heidelpay Payment Platform

Heidelberger Payment GmbH, **heidelpay** for short, is a certified e-commerce payment service provider. **heidelpay** offers an all-in-one payment solution for online shops, the **heidelpay Payment Platform**.

The **heidelpay Payment Platform** offers easy, flexible integration of all popular payment methods into your online shop. The platform supports various ways of integration, like ready-made shop modules, an XML interface, and the **heidelpay CheckOut**.

The **heidelpay Payment Platform** supports all popular e-commerce payment methods, like credit card, online transfer and, direct debit.

2.1 Test and Live Platform

heidelpay provides two separate installations of the **hPP**, a test installation and a live installation. Both installations of the **hPP** offer the same functionality. The difference between the two platforms is the handling of transfer of funds.

Transactions on the test installation will never cause transfer of funds. No transfer either between you and the customer or you and **heidelpay** will take place.

Transactions on the live installations however will cause transfer of funds.

2.2 Merchant and Transaction Modes

The **hPP** recognises three different modes for merchants and for transactions.

- *Merchant Mode*
    - INTEGRATOR_TEST
    - CONNECTOR_TEST
    - LIVE
- *Transaction Mode*
    - INTEGRATOR_TEST
    - CONNECTOR_TEST
    - LIVE

The merchant mode is assigned to you by **heidelpay**. The transaction mode is assigned to each individual transaction by your integration. The following sections list and explain the three different modes.

2.2.1 Integrator_Test

Transactions are sent to the integrator but not to the validator (risk management) or connector modules. This mode is used to test compliance against the integrator module

External financial partners are never contacted and there will not be any transfer of funds between you and the customer.

2.2.2 Connector_Test

Transactions enter the integrator module, access the validator modules (risk management) and then go to the connector. The connector operates in test mode.

2.2.3 Live

Transactions enter the integrator module, access the validator modules (risk management) and then go to the connector. The connector operates in live mode.

2.2.4 Combinations of Merchant and Transaction Modes

Table 1, Table 2 and Table 3 show which combinations of *`Merchant Mode`* and *`Transaction Mode`* will cause a transfer of funds. There are two different types of transfer of funds: The first type being payments between you and the customer, the second type being resulting transaction fees for you.

Table 1: Transfer of funds in Merchant Mode: `INTEGRATOR_TEST`

Transaction Mode	INTEGRATOR_TEST		CONNECTOR_TEST		LIVE	
Platform	Test	Live	Test	Live	Test	Live
Payments	No	No			Error	Error
Transaction fees	No	**Yes**	No	**Yes**	Error	Error

Table 2: Transfer of funds in Merchant Mode: `CONNECTOR_TEST`

Transaction Mode	INTEGRATOR_TEST		CONNECTOR_TEST		LIVE	
Platform	Test	Live	Test	Live	Test	Live
Payments	No	No	No	No	No	Error
Transaction fees	No	**Yes**	No	**Yes**	No	Error

Table 3: Transfer of funds in Merchant Mode: `LIVE`

Transaction Mode	INTEGRATOR_TEST		CONNECTOR_TEST		LIVE	
Platform	Test	Live	Test	Live	Test	Live
Payments	No	No	No	No	No	**Yes**
Transaction fees	No	**Yes**	No	**Yes**	No	**Yes**

2.3 Overview Payment Methods

The **heidelpay Payment Platform** supports various payment methods. The platform identifies each payment method by a two letter code.

Table 4: List of supported payment methods

Payment Method	Two Letter Identifier
Credit Card	`CC`
Debit Card	`DC`
Direct Debit	`DD`
Invoice	`IV`
Online Transfer	`OT`
Payment Card	`PC`
Prepayment	`PP`
Virtual Account	`VA`
Mobile Payment	`MP`

The list of payment methods supported by the **heidelpay Payment Platform** is continually growing. Please contact our sales department for an up to date list.

The following sections list all supported payment brands and provides test data for each payment method.

2.3.1 Credit Card

Table 5: List of supported credit card brands

Brand	Identifier	Availability
American Express	AMEX	worldwide
Diners	DINERS	worldwide
Discovery	DISCOVERY	worldwide
MasterCard	MASTER	worldwide
Visa	VISA	worldwide

Table 6: Test data for credit card

Brand	Card Number	Valid Through	CVV
American Express	375000000000007	any future date	any number
Diners			
Discovery	6011587918359498	any future date	any number
MasterCard	5105105105105100	any future date	any number

Brand	Card Number	Valid Through	CVV
Visa	4012888888881881	any future date	any number

2.3.2 Debit Card

Table 7: List of supported debit card brands

Brand	Identifier	Availability
4B	FOURB	
Carte Bleue	CARTEBLEUE	
Dankort	DANKORT	
EURO6000	EURO6000	
Maestro	MAESTRO	
Postepay	POSTEPAY	
Servired	SERVIRED	
Solo	SOLO	
VISA Electron	VISAELECTRON	

Table 8: Test data for debit card

Brand	Card Number	Valid Through	CVV
4B			
Carte Bleue	4111111111111111	any future date	any number
Dankort			
EURO6000			
Maestro	6799851000000032	any future date	any number
Postepay			
Servired			
Solo	6334580500000000	any future date	any number
VISA Electron	4012888888881881	any future date	any number

2.3.3 Direct Debit

Table 9: List of supported countries for direct debit

Country	Identifier
Austria	AT
Belgium	BE
Czech Republic	CZ
Denmark	DK
Finland	FI
France	FR
Germany	DE

Country	Identifier
Hungary	HU
Italy	IT
Netherlands	NL
Norway	NO
Spain	ES
Sweden	SE
Switzerland	CH
United Kingdom	GB

Table 10: Test data for direct debit

Country	Country ISO Code	Bank Code	Account
Austria	AT	20151	938044617
Austria	AT	20111	28161647502
Austria	AT	1232111	65785423
Belgium	BE	539	0075470-34
Czech Republic	CZ	0800	19-2000145399
Denmark	DK	0040	0440116243
Finland	FI	123456	785
France	FR	20041	010050500013M02606
Germany	DE	37040044	5320130
Germany	DE	38050000	46581
Germany	DE	10000000	1234567890
United Kingdom	GB	601613	31926819
Hungary	HU	10012349	12345678-91234567
Italy	IT		X0542811101000000123456**
Italy	IT		B0501812100000000115000**
Netherlands	NL		0417164300
Netherlands	NL		0000012112
Netherlands	NL		0123456789
Norway	NO		60033321115
Norway	NO		60031234568
Spain	ES		21000418450200051332**
Spain	ES		20382739996000057498**
Sweden	SE	5491	0000003
Switzerland	CH	100	123456-1-123-11
Switzerland	CH	4003	999999-99-999

Table 11: Test data for SEPA direct debit

Country	Country ISO Code	IBAN	BIC

2.3.4 Invoice

Table 12: List of supported invoice brands

Brand	Identifier	Availability
BillSAFE	BILLSAFE	Germany

Table 13: Test data for invoice

Brand	First/Given Name	Last/Family Name	Postal Code	City
BillSAFE	Paul	Positiv	49084	Osnabrück

2.3.5 Online Transfer

Table 14: List of supported online transfer brands

Brand	Identifier	Availability
SOFORTÜberweisung	DE	Germany
SOFORTÜberweisung	AT	Austria
iDEAL	NL	Netherlands
Giropay	DE	Germany

Table 15: Test data for online transfer

Brand	Bank Code	Account Number	PIN	TAN
SOFORTÜberweisung	88888888	123456		
Giropay	12345679	0000000300		

2.3.6 Payment Card

Table 16: List of supported payment card brands

Brand	Identifier	Availability
MangirKart	MANGIRKART	Turkey

Table 17: Test data for payment card

Brand

2.3.7 Prepayment

Table 18: List of supported prepayment brands

Brand	Identifier	Availability
BarPay		Germany

2.3.8 Virtual Account

Table 19: List of supported virtual account brands

Brand	Identifier	Availability
PayPal	PAYPAL	worldwide

Table 20: Test data for virtual account

Brand
PayPal

2.4 Transaction Types

The **heidelpay Payment Platform** supports various transaction types. The platform identifies each transaction type by a two letter code. While a payment method describes the method the customer uses to pay, the transaction type describes the activity you place on the payment method.

Table 21: List of all supported transaction types

Transaction Type	Three Letter Identifier	Two Letter Identifier	Description
Debit	`DEB`	`DB`	Debits a customer account.
Redebit	`RED`	`RB`	Re-executes a prior `DEBIT` using the same data.
Reservation	`RES`	`PA`	`RESERVATION` is used for a transaction in which the merchant needs authorization of a charge, but does not wish to actually make the charge at this point in time.
Capture	`CAP`	`CP`	`CAPTURE` causes a prior `RESERVATION` charge to be incurred against the customer account.
Receipt	`REC`	`RC`	`RECEIPT` is created when the merchant account is credited from a prior transaction (typically via payment methods `INVOICE`, `PREPAYMENT` or `ONLINE TRANSFER`).
Reversal	`REV`	`RV`	Reverses a prior `DEBIT`. A `REVERSAL` is only possible as long as no transfer of funds has occurred. If a transfer of funds has occurred a `REFUND` will be executed instead.
Refund	`REF`	`RF`	
Credit	`CRE`	`CD`	Credits a customer account.
Registration	`REG`	`RG`	Registers a payment instrument of a customer for future use.
Reregistration	`RRE`	`RR`	Changes data of an existing `REGISTRATION`.
Confirm Registration	`CFR`	`CF`	Confirms a `REGISTRATION` as still valid.
Deregistration	`DRE`	`DR`	Marks a `REGISTRATION` as no longer valid.
Schedule	`SCH`	`SD`	Defines periodically occurring transactions.
Change Schedule	`CSH`	`RS`	Changes an existing `SCHEDULE`.
End Schedule	`ESH`	`DS`	Ends the periodic execution of an existing `SCHEDULE`.
Chargeback	`CHB`	`CB`	Is generated when debiting the customer fails in a later stage of processing. For example the customer revokes the transaction at the issuer.

Transaction Type	Three Letter Identifier	Two Letter Identifier	Description
Chargeback Notification	`CBN`		Indicates an incoming or unprocessed chargeback.
Chargeback Reversal	`CBR`		Revocation of a prior chargeback.
Reconciliation	`RCL`	`RL`	Generated by the system when balancing accounts.
Finalise	`FIN`	`FI`	Finalises a transaction.

2.5 Payment Codes

The combination of the preceding described Payment Methods and Payment Types is called Payment Code. The Payment Code is one of the most central elements of a request, because it determines the behaviour and the mandatory elements of a request.

In order to build or parse a specific method or type code, there is an easy scheme for all occurrences. All type codes come from the same general set of codes and are totally polymorphic. That means e.g. that the payment code for a direct debit (DD) transaction of type refund (RF):

```
PAYMENT.CODE=DD.RF
```

is the same for a credit card (CC) transaction of type refund (RF), except the preceding method code:

```
PAYMENT.CODE=CC.RF
```

3 heidelpay Payment Plattform Interface

There are two ways to communicate with the **hPP**. Both methods are based on Secure Hypertext Transfer Protocol (HTTPS). The first approach simply uses the HTTP-POST-Parameters for passing information to **hPP**.The second way is to use the XML Interface. Here the information is formatted within a XML message. This XML message is passed to the **hPP** as HTTP-Message-Body.

Because of simplicity and clarity, we will use the POST interface for further considerations and example.
If you are interested in the XML-Interface please do not hesitate to contact us.

3.1 POST-Interface Sample

For the POST-Interface a possible sample Transaction-Request can look like this.

```
REQUEST.VERSION=1.0
SECURITY.SENDER=123a456b789c123d456e789f012g345
USER.LOGIN=12345678123456781234567 8abcdabcd
USER.PWD=geheim
TRANSACTION.MODE= INTEGRATOR_TEST
TRANSACTION.RESPONSE=SYNC
TRANSACTION.CHANNEL=546a456b789c123d456e789f012g821
IDENTIFICATION.TRANSACTIONID=MerchantAssignedID
IDENTIFICATION.SHOPPERID=customerid12345
IDENTIFICATION.INVOICEID=20140800012
PAYMENT.CODE=DD.DB
PRESENTATION.AMOUNT=1.00
PRESENTATION.CURRENCY=EUR
PRESENTATION.USAGE=Order Number 1234
ACCOUNT.HOLDER=Joe Doe
ACCOUNT.IBAN=DE64700700240618495000
ACCOUNT.COUNTRY=DE
NAME.GIVEN=Joe
NAME.FAMILY=Doe
ADDRESS.STREET=Leopoldstr. 1
ADDRESS.ZIP=80798
ADDRESS.CITY=München
ADDRESS.STATE=BY
ADDRESS.COUNTRY=DE
CONTACT.EMAIL=info@provider.com
CONTACT.IP=123.123.123.123
```

The HTTPS Post response repeats some information of the request and adds additional parameters to it. The most important additions are the TRANSACTION.UNIQUEID, TRANSACTION.SHORTID and the Processing group parameters.

```
RESPONSE.VERSION=1.0
SECURITY.SENDER=123a456b789c123d456e789f012g345
TRANSACTION.MODE= INTEGRATOR_TEST
TRANSACTION.RESPONSE=SYNC
TRANSACTION.CHANNEL=546a456b789c123d456e789f012g821
IDENTIFICATION.TRANSACTIONID=MerchantAssignedID
IDENTIFICATION.UNIQUEID=h987i654j321k098l765m432n210o987
IDENTIFICATION.SHORTID=1234.5678.9876
IDENTIFICATION.SHOPPERID=customerid12345
IDENTIFICATION.INVOICEID=20090100012
PROCESSING.CODE=DD.DB.90.00
PROCESSING.TIMESTAMP=2003-02-12 14:58:07
PROCESSING.RESULT=ACK
PROCESSING.STATUS.CODE=90
PROCESSING.STATUS=NEW
PROCESSING.REASON.CODE=00
PROCESSING.REASON=Successful Processing
PROCESSING.RETURN.CODE=000.000.000
PROCESSING.RETURN=Transaction succeeded
PAYMENT.CODE=DD.DB
PRESENTATION.AMOUNT=1.00
PRESENTATION.CURRENCY=EUR
PRESENTATION.USAGE=Order Number 1234
CLEARING.AMOUNT=1.00
CLEARING.CURRENCY=EUR
CLEARING.DESCRIPTOR=1234.1234.1234 - Order Number 1234
RISKMANAGEMENT.PROCESS=AUTO
AGGREGATION.RM.AG=true
AGGREGATION.RM.AG.STATUS.CODE=90
AGGREGATION.RM.AG.STATUS=SUCCESS
AGGREGATION.RM.AG.REASON.CODE=90
AGGREGATION.RM.AG.REASON=Risk Management Score is 90
VALIDATOR.RM.AC=true
VALIDATOR.RM.AC.STATUS.CODE=00
VALIDATOR.RM.AC.STATUS=SUCCESS
VALIDATOR.RM.AC.REASON.CODE=00
VALIDATOR.RM.AC.REASON=Account Validation Successful
VALIDATOR.RM.BC=true
VALIDATOR.RM.BC.STATUS.CODE=00
VALIDATOR.RM.BC.STATUS=SUCCESS
VALIDATOR.RM.BC.REASON.CODE=00
VALIDATOR.RM.BC.REASON=Bank Code Validation Successful
```

3.2 Parameter Groups

3.2.1 Header

The header group of the HTTPS Post parameters holds transmission and security related information.

Request:

```
REQUEST.VERSION=1.0
SECURITY.SENDER=123a456b789c123d456e789f012g345
```

Response:

```
RESPONSE.VERSION=1.0
SECURITY.SENDER=123a456b789c123d456e789f012g345
```

Parameter	Data type	Length	Mandatory	Description
REQUEST.VERSION	Alphanumeric	3	Mand.	The VERSION parameter indicates a major release. The HTTPS Post parameter scheme changes in such

				way that the preceding one is not compatible anymore with the new one. Therefore the VERSION parameters in the request and response message have to be increased.
SECURITY.SENDER	Alphanumeric	32	Mand.	Each Server which sends requests to the system has an own sender unique ID. The sender UID is no logical business orientated subdivision like the channel ID, but refers to physical installations of software. Please provide here the value you have received from the customer support department.

3.2.2 Transaction

The Transaction group contains all information required to process a transaction.

```
TRANSACTION.MODE=LIVE
TRANSACTION.RESPONSE=SYNC
TRANSACTION.CHANNEL=546a456b789c123d456e789f012g821
```

The Transaction group has three parameters which determine the processing of the transaction.

Value for TRANSACTION.MODE	Description
INTEGRATOR_TEST	Transaction is just send to the Integrator and not to the Validator (Risk Management) or Connector modules. Used to test compliance against the Integrator module.
CONNECTOR_TEST	Transaction enters the Integrator module, accesses the Validator modules (Risk Management) and then goes to the Connector. The Connector operates in test mode.
LIVE	Transaction enters the Integrator module, accesses the Validator modules (Risk Management) and then goes to the Connector. The Connector operates in live mode.
SYNC	Transaction is processed in the synchronous mode, which means the client will get in the direct response the result of the processing.
ASYNC	Transaction is processed in the asynchronous mode (also known as batch mode). The direct response will just aknowledge the receipt of the transaction. The result of the processing is delivered back as indirect response over another asynchronous communication channel like a SFTP repository or a POST call back method.
Format: 32 alphanumeric digits	The channel ID is a unique key for the identification of the unit which sends transactions into the system. Every merchant can have multiple channels for different purposes. Possible division criteria are for example different shops, organizational units,

	customer groups or countries. The channel ID doesn‟t refer to physical installations of the software like the sender ID but is a logical business orientated subdivision. Different channels help to analyze the entirety of transactions and to provide different system configurations for a nonuniform transaction base. The channel IDs are assigned by the account management.

3.2.3 User

The User group contains the information about the user sending the request.

```
USER.LOGIN=12343214123432141234321412343214
USER.PWD=geheim
```

USER Parameter	Data type	Length	Mandatory	Description
USER.LOGIN	Alphanumeric	32	Mand.	User Id of the sending user. This user must be configured with SEND rights.
USER.PWD	Alphanumeric	5…16	Mand.	Password of the sending user

3.2.4 Identification Group

The identification group contains all IDs which are used for the identification of the transaction:

```
IDENTIFICATION.TRANSACTIONID
IDENTIFICATION.UNIQUEID
IDENTIFICATION.SHORTID
IDENTIFICATION.REFERENCEID
IDENTIFICATION.SHOPPERID
IDENTIFICATION.INVOICEID
```

In the request the merchant can provide a Transaction ID for own matching purposes. For transaction types which require a reference to a former transaction (Capture, Reversal, Refund) a Reference ID has to be provided. The Shopper ID is used to group transactions of a certain shopper.

Request:

```
IDENTIFICATION.TRANSACTIONID=MerchantAssignedID
IDENTIFICATION.REFERENCEID=r123i654j321k098l765m432n21e456
IDENTIFICATION.SHOPPERID=shopper000321
```

While processing the system generates a universal unique ID. This Unique ID must be used for all automated matching and search purposes. The Reference ID is the Unique ID of the referenced transaction. To provide an ID which is shorter and easy to enter manually, the Short ID is provided. The Short ID is used for the descriptor of the transaction and to search manually for transactions in the management platform. The Short ID doesn’t guarantee universal uniqueness, however the probability for non-uniqueness is very low.

Response:

```
IDENTIFICATION.TRANSACTIONID=MerchantAssignedID
IDENTIFICATION.UNIQUEID=h987i654j321k0981765m432n210o987
IDENTIFICATION.SHORTID=1234.5678.9876
IDENTIFICATION.REFERENCEID=r123i654j321k0981765m432n21e456
IDENTIFICATION.SHOPPERID=shopper000321
```

Parameter IDENDIFICATION	Data type	Length	Mandatory	Description
IDENTIFICATION.TRANSACTIONID	Alphanumeric	0..256	Optional	Id assigned by the merchant. Uniqueness in the system or even just within the channel is not checked.
IDENTIFICATION.UNIQUEID	Alphanumeric	32	Mand.	Only ID where the uniqueness within the system is absolutely guaranteed. Has to be used for all automated matching and reference purposes.
IDENTIFICATION.SHORTID	Numeric / Dots	14	Mand.	ID which is used for manual entry and search purposes. The likelihood for uniqueness is very high, but not guaranteed.
IDENTIFICATION.INVOICEID	Alphanumeric	0..256	Optional	Id assigned by the merchant to assign it to a certain invoice. Typically this invoice id is the id the merchant also communicates to the shopper for a certain invoice
IDENTIFICATION.SHOPPERID	Alphanumeric	0..256	Optional	Id assigned by the merchant to assign it to a certain shopper. Typically the user id or customer id of the shopper within the merchant‟s shop is sent in here. It can be used to search for all transactions of one shopper in the analysis backend tool.
IDENTIFICATION.REFERENCEID	Alphanumeric	32	Cond. Mand.	References to the Unique ID of another transaction. Only needed for the submission of following transaction types: Capture, Reversal and Refund. Chargeback and Deposit transactions contain the Reference ID in the response

				message.

3.2.5 Payment Group

The Payment group determines which payment method and type to use and provides all monetary payment details of the transaction. Furthermore it contains the description of the transaction by means of the USAGE and DESCRIPTOR parameters.

```
PAYMENT.CODE=DD.DB
```

If you want to use certain payment methods, please ensure that the applied Channel ID has been activated for these methods by the account management. Depending on the chosen payment method, there are specific types available:

	Bank transfer				**Card payment**		**Online**		**Other**	
	DD	CT	PP	IV	CC	DC	MP	OT	VA	RM
Payment types										
Preauthorization (PA)	x	x	x	x	x	x	x	x	x	x
Debit (DB)	x				x	x	x	x	x	
Credit (CD)		x					x		x	
Reversal (RV)	x	x			x	x			x	
Refund (RF)	x	x	x	x	x	x	x	x	x	
Rebill (RB)	x				(x)	(x)				
Chargeback (CB)	x				x			x		
Receipt (RC)	x	x	x	x			x	x		
Registration Types										
Registration (RG)	x	x	x	x	x	x		x	x	
Confirmation (CF)	x									
Reregistration (RR)	x	x	x	x	x	x		x	x	
Deregistration (DR)	x	x	x	x	x	x		x	x	

A detailed description of the payment types can be found in the respective chapters. The following table gives a basic overview:

TYPE Name	**Description**
Debit	Debits the credit card or bank account of the end customer. (CF +)
Credit	Credits the bank account of the end customer. (CF -)
Preauthorisation	Reserves a certain amount on the credit card of the end customer. This amount can be captured later. Certain Acquirers call this transaction type also reservation. (CF neutral)
Reversal	Reverses the transaction before the clearing cut off time of the processor or bank. A reversed transaction does not appear on the statement of the end

	customer. (CF -)
Refund	After the cut off time has expired a customer can only be refunded by executing a Refund transaction. The amount of the Refund transaction can also be less than the original amount. (CF -)
Chargeback	If a debit transaction of a credit card or direct debit account is returned for any reason, a Chargeback is loaded into the system. (CF -)
Receipt	If a credit transfer transaction is returned by the bank or if the end customer makes a payment to the clearing account, a Receipt is created. (CF +)

In the request the Payment group has to include the Presentation tag, while the response gives back the Presentation tag as well as the Clearing tag. The Clearing tag contains all data as the end customer obtains it on his credit card or bank statement.

Request:

```
PAYMENT.CODE=DD.DB
PRESENTATION.AMOUNT=1.00
PRESENTATION.CURRENCY=EUR
PRESENTATION.USAGE=Order Number 1234
```

Response:

```
PAYMENT.CODE=DD.DB
PRESENTATION.AMOUNT=1.00
PRESENTATION.CURRENCY=EUR
PRESENTATION.USAGE=Order Number 1234
CLEARING.AMOUNT=1.00
CLEARING.CURRENCY=EUR
CLEARING.DESCRIPTOR=1234.1234.1234 - Order Number 1234
CLEARING.BANKNAME=Royal Bank of Scottland
CLEARING.FXRATE=0.500000
CLEARING.FXSOURCE=FTX
CLEARING.FXDATE=2003-02-12
```

All parameters within the Presentation group are mandatory:

Parameter	Data type	Length	Mandatory	Description
PRESENTATION				
PRESENTATION.AMOUNT	#0.00	1..10,2	Mand.	Presentation Amount in currency of the Currency parameter. The dot is used as decimal separator.
PRESENTATION.CURRENCY	Alpha	3	Mand.	Currency code according to the ISO 4217 specification plus the currency "PTS" for Points (e.g. used for loyalty programs).
PRESENTATION.USAGE	Alphanumeric	0..128	Mand.	Provides the dynamic part of the descriptor, which appears on the end customer‟s statement.

	Enables the end customer to associate the transaction on the statement to the online transaction.

The BANKNAME parameter just appears for direct debit or credit transfer transactions in countries where the name of the clearing bank can be retrieved. The FX ... parameters within the CLEARING group are just given if a currency conversion had to take place.

Parameter **CLEARING**	**Data type**	**Length**	**Mandatory**	**Description**
CLEARING.AMOUNT	#0.00	1..10,2	Mand.	Settlement amount on the end customer‟s account in currency of the CURRENCY parameter. The dot is used as decimal separator.
CLEARING.CURRENCY	Alpha	3	Mand.	Currency code according to the ISO 4217 specification plus the currency “PTS” for Points (e.g. used for loyalty programs).
CLEARING.DESCRIPTOR	Alphanumeric	14..256	Mand.	Appears on the statement of the end customer. The Descriptor is a concatenation of the Short ID, system defined text and the USAGE parameter. Enables the end customer to associate the transaction on the statement with the online transaction.
CLEARING.BANKNAME	Alpha	0..64	Cond. Mand.	Name of the end customer‟s bank for direct debit and credit transfer transactions. The BANKNAME parameter just appears in countries where this name can be retrieved.

CLEARING.FXRATE	#0.000000	1..6,6	Cond. Mand.	If the currency provided in PRESENTATION parameters is not equal to the currency of the end customer‟s account, Direct Debit and Credit Transfer transactions have to be converted. The applied Foreign Exchange Rate is shown with a precision of 6 decimal places after the dot.
CLEARING.FXSOURCE	Alphanumeric	0..256	Cond. Mand.	The Source of the applied Foreign Exchange Rate.
CLEARING.FXDATE	Date	10	Cond. Mand.	The applied rates are fixed rates of the applied Foreign Exchange Source for a certain date.
CLEARING.BALANCE	#0.00	1..10,2	Optional	Returns the remaining balance on the account. This field is only set for accounts (e.g. loyalty cards) that support this feature.

Please note that in case of a chargeback the presentation amount is the amount given by the bank and not by the merchant.

3.2.6 Account Group

The Account group holds all information regarding a credit card or bank account. Many parameters depend on the chosen payment method. The credit card account parameters look like this:

```
ACCOUNT.HOLDER=Joe Doe
ACCOUNT.NUMBER=1234 1234 1234 1234
ACCOUNT.BRAND=VISA
ACCOUNT.EXPIRY_MONTH=09
ACCOUNT.EXPIRY_YEAR=2005
ACCOUNT.VERIFICATION=1233
```

The direct debit account parameters are shown below:

```
ACCOUNT.HOLDER=Joe Doe
ACCOUNT.NUMBER=618495000
ACCOUNT.BANK=70070024
ACCOUNT.COUNTRY=DE
```

For a complete list of all possible values and more specific length restrictions see the later chapters about the different payment methods:

Parameter **ACCOUNT**	**Data type**	**Length**	**Mandatory**	**Description**
ACCOUNT parameters	Data Type	Length	Mand. / Optional	Description
ACCOUNT.HOLDER	Alpha	4..128	Mand.	Holder of the credit card or bank account. If it can be assumed that end customer is the holder of the account this field can be concatenated from the given and family name of the NAME parameters.
ACCOUNT.NUMBER	Alpha numeric	3..64	Cond. Mand.	Numer of the credit card or domestic bank account. Includes also possible check digits.
ACCOUNT.BRAND	Alpha	3..10	Cond. Mand.	Name of the credit card brand.
ACCOUNT.EXPIRY_MONTH	Numeric	2	Cond. Mand.	Expiration month of the credit card.
ACCOUNT.EXPIRY_YEAR	Numeric	4	Cond. Mand.	Expiration year of the credit card.
ACCOUNT.VERIFICATION	Numeric	3..4	Cond. Mand.	The verification number of the credit card (CVV2, CVC2, FDBC).
ACCOUNT.BANKNAME	Alpha numeric	0..255	Cond. Mand.	Especially of interest for Online Transfer methods to determine which bank was chosen.
ACCOUNT.BANK	Alpha numeric	0..12	Cond. Mand.	The domestic code of the bank which holds the direct debit or credit transfer account.
ACCOUNT.IBAN	Alpha numeric	15..28	Cond. Mand.	International Bank Account Number. Can be provided instead of the domestic NUMBER parameter if available.
ACCOUNT.BIC	Alpha	8 or 11	● Cond. Mand.	● Bank Identifier Code (SWIFT).

	numeric			Can be provided instead of the domestic BANK parameter if available.
ACCOUNT.COUNTRY	Alpha	2	Cond. Mand.	Country code according to the ISO 3166-1 specification.
ACCOUNT.LIMIT	#0.00	1..10,2	Cond. Mand.	Maximum preauthorization (PA) or debit (DB) amount for a single transaction on a specific account. Generally this tag is optional, yet certain direct debit schemes require the declaration of a limit (e.g. Norway).
ACCOUNT.IDENTIFICATION	Alpha numeric	1..16	Cond. Mand.	Certain direct debit processes require a custom identification number. Currently used for Norway, Belgium and Italy.
ACCOUNT.REGISTRATION_URL	Alpha numeric	10..256	Cond. Mand.	Especially used for paper based direct debit schemes in order to give the end customer a URL where he can to download the corresponding paper mandate, which he can fill in and sign.

3.2.7 Customer Group

The customer group contains all customer specific information like name, address and contact details. The NAME and ADDRESS group are used basically for risk management purposes while the CONTACT group is important for collection, call back validations and transmission of the mandate templates for direct debit countries which require a written confirmation.

The IDENTIFICATON group contains very specific customer information, which is necessary for some advanced risk management checks or dedicated direct debit schemes (e.g. domiciliation bancaria in Spain).

```
NAME.SALUTATION=Mr
NAME.TITLE=Dr
NAME.GIVEN=Joe
NAME.FAMILY=Doe
NAME.COMPANY=SampleCompany
ADDRESS.STREET=Leopoldstr. 1
ADDRESS.ZIP=80798
ADDRESS.CITY=München
ADDRESS.STATE=BY
ADDRESS.COUNTRY=DE
CONTACT.PHONE=+49-89-1234566
CONTACT.MOBILE=+49-172-1234566
CONTACT.EMAIL=info@provider.com
CONTACT.IP=123.123.123.123
CUSTOMER.IDENTIFICATION.PAPER=PASSPORT
CUSTOMER.IDENTIFICATION.VALUE=D00143434
```

Parameter **NAME**	**Data type**	**Length**	**Mandatory**	**Description**
NAME.SALUTATION	Alpha	1..20	Optional	
NAME.TITLE	Alphanumeric	1..20	Optional	Title of the end customer.
NAME.GIVEN	Alpha	2..40	Mandatory	Given name of the end customer.
NAME.FAMILY	Alpha	2..40	Mandatory	Family name of the end customer.
NAME.SEX	Alpha	1	Optional	Sex of the shopper, „M“ for male or „F“ for female
NAME.BIRTHDATE	Alphanumeric	10	Optional	Date in the format yyyy-MM-dd, e.g. 1970-09-12
NAME.COMPANY	Alphanumeric	2..40	Optional	Company name of the end customer.
Parameter **ADDRESS**	**Data type**	**Length**	**Mandatory**	**Description**
ADDRESS.STREET	Alphanumeric	5..50	Mandatory	Street and house number of the end customer.
ADDRESS.ZIP	Alphanumeric	1..10	Mandatory	ZIP code of the city of the end customer.
ADDRESS.CITY	Alpha	2..30	Mandatory	City where the end customer lives.
ADDRESS.STATE	Alpha	2..10	Optional	State of the city of the end customer. Not required for many countries.
ADDRESS.COUNTRY	Alpha	2	Mandatory	Country code according to the ISO 3166-1 specification.
Parameter	**Data type**	**Length**	**Mandatory**	**Description**

CONTACT				
CONTACT.PHONE	Alphanumeric	8..25	Optional	Used for risk management and collection. Has to start with a digit or a '+', at least 7 and max 25 chars long
CONTACT.MOBILE	Alphanumeric	10..25	Optional	Used for risk management and collection. Has to start with a digit or a '+', at least 7 and max 25 chars long
CONTACT.EMAIL	Alphanumeric	6..128	Mandatory	Used for risk management, collection and transmission of direct debit mandates.
CONTACT.IP	000.000.000.000	15	Optional	IP number of end customer. Used for statistics and collection.
Parameter **IDENTIFICATION**	**Data type**	**Length**	**Mandatory**	**Description**
CUSTOMER.IDENTIFICATION.PAPER	IDCARD PASSPORT TAXSTATEMENT	5..12	Cond. Mandatory	Type of identity paper
CUSTOMER.IDENTIFICATION.VALUE	Alphanumeric	8..64	Cond. Mandatory	Number of identity paper

3.2.8 Recurrence Group (Manual Recurrence)

Recurring transactions are flagged with the RECURRENCE parameter.

For initial transaction (containing CVV code) send the following parameter:

```
RECURRENCE.MODE="INITIAL"
```

All recurring transactions (without a CVV code) are sent with this parameter:

```
RECURRENCE.MODE="REPEATED"
```

Parameter **RECURRENCE**	**Data type**	**Length**	**Mandatory**	**Description**

RECURRENCE.MODE	Alphanumeric	0..64	Mandatory	One of "INITIAL" or "REPEATED"

This tag will only take affect if the used Merchant Account is supporting recurring transaction and is configured for recurring transactions in the hIP. Please check with your account manager if the used acquirer is supporting recurring transactions!

3.2.9 Criterion Group

The CRITERION group gives the freedom to add additional customized attributes to the transaction, which are not available by the standard set of HTTPS Post parameters. With the help of these freely definable „Criteria‟ a customised statistics and analysis can be performed on the base of the transactions which have these parameters included.

Possible examples are the turnover of different affiliates, age groups or any other imaginable value or subdivision which should be available in the analysis front end later.

```
CRITERION.Affiliate=ExternalShopXY
CRITERION.Age=30-40
CRITERION.known=yes
CRITERION.CustomerID=1234567
```

The Criterion group has a purely statistical and informative value and should not be confused with the Channel ID. The Channel ID is the central reference for all kinds of configuration information of the payment process as well as for the pricing and billing process.

Also the Channel ID can exclusively be assigned by the customer support department, while the different Criteria can be defined dynamically by the merchant, without any consultation of the payment service provider.

Parameter and name **CRITERION**	**Data type**	**Length**	**Mandatory**	**Description**
CRITERION.<AN32>	Alphanumeric	0..1024	Optional	Freely definable value for a specific criterion. The value can be entered in the analysis platform to retrieve all transactions with this criterion value. If several differently named criteria have the same values, also the name of the criteria must be provided in the web front end in order to retrieve the right transactions (see below).
Value for <AN32>	**Description**			
AN32	Freely definable name for a specific criterion. The same name must be entered in the analysis platform to retrieve all transactions which have this criterion.			

3.2.10 Authentication Group

This group allows the merchant to send in any kind of Authentication information that authenticates the transaction itself like 3-D-Secure, SMS-identification and the like.

Currently this group is used to send in authentication information gathered from a Verified by Visa (VbV) or Mastercard Secure Code (MSC) request performed by a merchant himself.

See the document "Asynchronous Workflow" for more details about 3-D-Secure processes.

```
AUTHENTICATION.TYPE=3DSecure
AUTHENTICATION.RESULT_INDICATOR=05
AUTHENTICATION.PARAMETER.VERIFICATION_ID=AAACAgSRBklmQCFgMpEGAAAAAAA
AUTHENTICATION.PARAMETER.VERIFICATION_ TYPE=2
AUTHENTICATION.PARAMETER.XID=CAACCVVUlwCXUyhQNlSXAAAAAAA
```

The Authentication group has one value to determine the type of Authentication.

<table>
<tr><td><b>Values for
AUTHENTICATION.TYPE</b></td><td colspan="4"><b>Description</b></td></tr>
<tr><td>3DSecure</td><td colspan="4">Use this value for the type if the authentication was a 3DSecure process.</td></tr>
<tr><td>Parameter
AUTHENTICATION</td><td>Data type</td><td>Length</td><td>Mandatory</td><td>Description</td></tr>
<tr><td><b>AUTHENTICATION.PARAMETER.<AN32&gt;</b></td><td><b>Alpha-numeric</b></td><td><b>1..32</b></td><td><b>Cond.</b></td><td><b>Value of Parameter of the specified Authentication type.</b></td></tr>
<tr><td>AUTHENTICATION.RESULT_INDICATOR</td><td>Alpha-numeric</td><td>1..128</td><td>Optional</td><td>Contains the result of the Authentication process. For 3D-Secure this must be one of the following (ECI-Value):
01 = MASTER_3D_ATTEMPT
02 = MASTER_3D_SUCCESS
05 = VISA_3D_SUCCESS
06 = VISA_3D_ATTEMPT
07 = DEFAULT_E_COMMERCE</td></tr>
<tr><td><b>Values for
&lt;AN32&gt;</b></td><td colspan="4"><b>Description</b></td></tr>
<tr><td>Alphanumeric 1..32</td><td colspan="4">Freely definable name for a specific parameter.</td></tr>
</table>

3.2.11 Processing Group

The processing group contains a summary of the result of the complete processing. The structure of the status and reason codes is hierarchical while the return code is an independent, internal value which is used for very specific return messages. Any merchant side matching should be performed on the processing code or status and reason codes.

```
PROCESSING.CODE=DD.DB.90.00
PROCESSING.TIMESTAMP=2003-02-12 14:58:07
PROCESSING.RESULT=ACK
PROCESSING.STATUS.CODE=90
PROCESSING.STATUS=NEW
PROCESSING.CONFIRMATON STATUS=CONFIRMED
PROCESSING.REASON.CODE=00
PROCESSING.REASON=Successful Processing
PROCESSING.RETURN.CODE=000.000.000
PROCESSING.RETURN=Transaction succeeded
PROCESSING.RISK_SCORE=-20
```

The result derives from the status of the transaction, which means that a transaction which has the status REJECTED or FAILED has the result NOK, while all other statuses result in an ACK. For each status there are one or several reasons.

Parameter **PROCESSING**	**Data type**	**Length**	**Mandatory**	**Description**
PROCESSING.CODE	AA.AA.00.00	11	Mandatory	The processing code of a transaction. It‟s a simple concatenation of method, type, status and reason code. It provides the context within a status or reason has appeared and contains together with the return code all information about the processing of a transaction.
PROCESSING.TIMESTAMP	Timestamp	19	Mandatory	Date and time when the transaction was processed.
PROCESSING.RESULT	ACK, NOK	3	Mandatory	In the case of the status REJECTED or FAILED the result is NOK (Not OK). In all other cases the result is ACK (Acknowledge).
PROCESSING.STATUS.CODE	00	2	Mandatory	The status code of a transaction. See the appendices for a complete list.
PROCESSING.STATUS	Alphanumeric	0..16	Mandatory	Status message which belongs to the Status code (e.g. NEW, REJECTED …). Please use the code and not the message for matching purposes.

PROCESSING. CONFIRMATON_STATUS	CONFIRMED, PENDING	0..20	Optional	In case of a response to a Registration (RG) or Reregistration (RR) request, this status message tells if the registration was auto-confirmed immediately (CONFIRMED) or is waiting (PENDING) for a confirmation. A debit request (DB) can only be sent in, if the Registration was confirmed.
PROCESSING.REASON.CODE	00	2	Mandatory	The reason code of a transaction. Every status has one or several reasons..
PROCESSING.REASON	Alphanumeric	0..64	Mandatory	Reason message which belongs to the Reason code. (e.g. Successful Processing, Account Validation, Bank Code Validation …). Please use the code and not the message for matching purposes.
PROCESSING.RETURN.CODE	0000.0000.0000	11	Mandatory	The return id of a transaction.
PROCESSING.RETURN	Alphanumeric	0..256	Mandatory	Return message which belongs to the Return code (e.g. Validation Algorithm DE102 failed, …). Please do not match on return messages.
PROCESSING.RISK_SCORE	Numeric		Optional	Contains the risk score for the executed transaction (for a payment transaction the score of the corresponding RM.RI transaction). This value is only returned if risk operations (e.g. blacklist, velocity checks …) were executed.

3.2.12 Asynchronous Response Processing Group

ATTENTION: Please be aware that not all asynchronous processes can fully be covered with the POST Transaction integration. Most of them are only possible with XML Integration. Please contact your account manager for more details!

For asynchronous response messages additional paramers are part of the response message that contain the redirect information for the merchant.

```
PROCESSING.REDIRECT.URL= https://www.mybank.com/3D_validation
PROCESSING.REDIRECT.PARAM.TermUrl=https://terminalurl.org/payment/3D_response
PROCESSING.REDIRECT.PARAM.PAReq=m123n456o789p876q543r22323145346576
PROCESSING.REDIRECT.PARAM.MD=24358432975908324758904327589434
```

Depending on the type of the asynchronous process (i.e. Online Bank Transfer, Verified By VISA, Mastercard Securecode, …) the Redirect group contains a number of different parameters.

Typically the merchant redirects the end user‟s browser to the PROCESSING.REDIRECT_URL and passes the other parameters in this subgroup as parameters to the PROCESSING.REDIRECT_URL.

Please refer to the document “Asynchronous Workflow” for an explanation how these parameters must be processed by the merchant to fulfill the workflow requirements of the asynchronous process.

Parameter REDIRECT	Data type	Length	Mandatory	Description
PROCESSING. **REDIRECT_URL**	Alphanumeric, URL	2048	Mandatory	URL that must be called by the merchant in order to proceed. The merchant redirects the browser to this URL.
PROCESSING. **REDIRECT.PARAM.<AN32>**	Alphanumeric	4096	Optional	Any kind of parameter needed for the workflow. Each Parameter needs to be posted to the URL.
Values for **<AN32>**	**Description**			
Alphanumeric 1..32	Freely definable name for a specific parameter.			

3.2.13 Connector Group

Within the Connector group information about the connector which was selected for processing of the transaction is given back. This is especially used in conjunction with Prepayment (PP) or Invoice (IV) transactions, where the end customer needs to know to which bank account he has to direct his payment. Furthermore it is necessary to identify the receiving bank account in the context of mandate registrations (DD.RG).

```
CONNECTOR.ACCOUNT.HOLDER= Internal Account Holder
CONNECTOR.ACCOUNT.NUMBER=618495000
CONNECTOR.ACCOUNT.BANK=70070024
CONNECTOR.ACCOUNT.IBAN= DE82700202700666898869
CONNECTOR.ACCOUNT.BIC= HYVEDEMM
CONNECTOR.ACCOUNT.COUNTRY=DE
```

Parameter	Data type	Length	Mandatory	Description
CONNECTOR				
CONNECTOR.ACCOUNT.HOLDER	Alpha	4..128	Mand.	Holder of the bank account. This is generally a company name.
CONNECTOR.ACCOUNT.NUMBER	Alpha-numeric	3..64	Cond. Mand.	Account number of the processing bank account. Includes also possible check digits.
CONNECTOR.ACCOUNT.BANK	Alpha-numeric	0..12	Cond. Mand.	The domestic code of the bank which holds the direct debit or credit transfer account.
CONNECTOR.ACCOUNT.IBAN	Alpha-numeric	15..28	Cond. Mand.	International Bank Account Number. This number can be used by foreign customers to make cross-border credit transfers.
CONNECTOR.ACCOUNT.BIC	Alpha-numeric	8 or 11	Cond. Mand.	Bank Identifier Code (SWIFT). This bank code can be used by foreign customers to make cross-border credit transfers.
CONNECTOR.ACCOUNT.COUNTRY	Alpha	2	Cond. Mand.	Country code according to the ISO 3166-1 specification.

4 Implementing Payment Transactions

Chapter 3 dealt with data formats and possible configuration settings which can be processed by the **hPP**. After knowing the configuration possibilities we want describe how real payment transactions can be implemented into your web application respectively your workflows. This is what the following chapter describes.

Therefore the **hPP** offers several ways:

- use one of plenty heidelpays out-of-the-box working webshop-modules
- use the iframe based **heidelpay checkout** form named **hco**
- use heidelpays whitelabel interface

The whitelabel approach is what this document dealt with.

4.1 Whitelabel Interface

The **hPP** offers a whitelabel payment Interface. This allows you to guide your customer through the payment process, without letting him know that he has left your web application. Styling, layout and validation of the payment form is done by your web application.

4.1.1 Generic Workflow

The first steps of a successful whitelabel payment processing are always the same:

1) Sending general information about the payment and the customer to the hPP whitelabel interface.
2) Retrieve the response parameter FRONTEND.REDIRECT_URL from hPP.
3) Let your web application view your customer a payment form.
   The action URL of the form must be the FRONTEND.REDIRECT_URL.
   The form fields names must follow the hPP filed name conventions.
4) E.g. do some validation on the input from your customer and submit the form
5) Depending on the payment method the customer is redirected to a third party company or not
6) Retrieve the response from hPP about the payment to FRONTEND.RESPONSE_URL.
7) Further steps are depends on the chosen payment method

For a synchronous case like e.g. for direct debit with a SEPA account the sequence of events is shown in Figure 1.

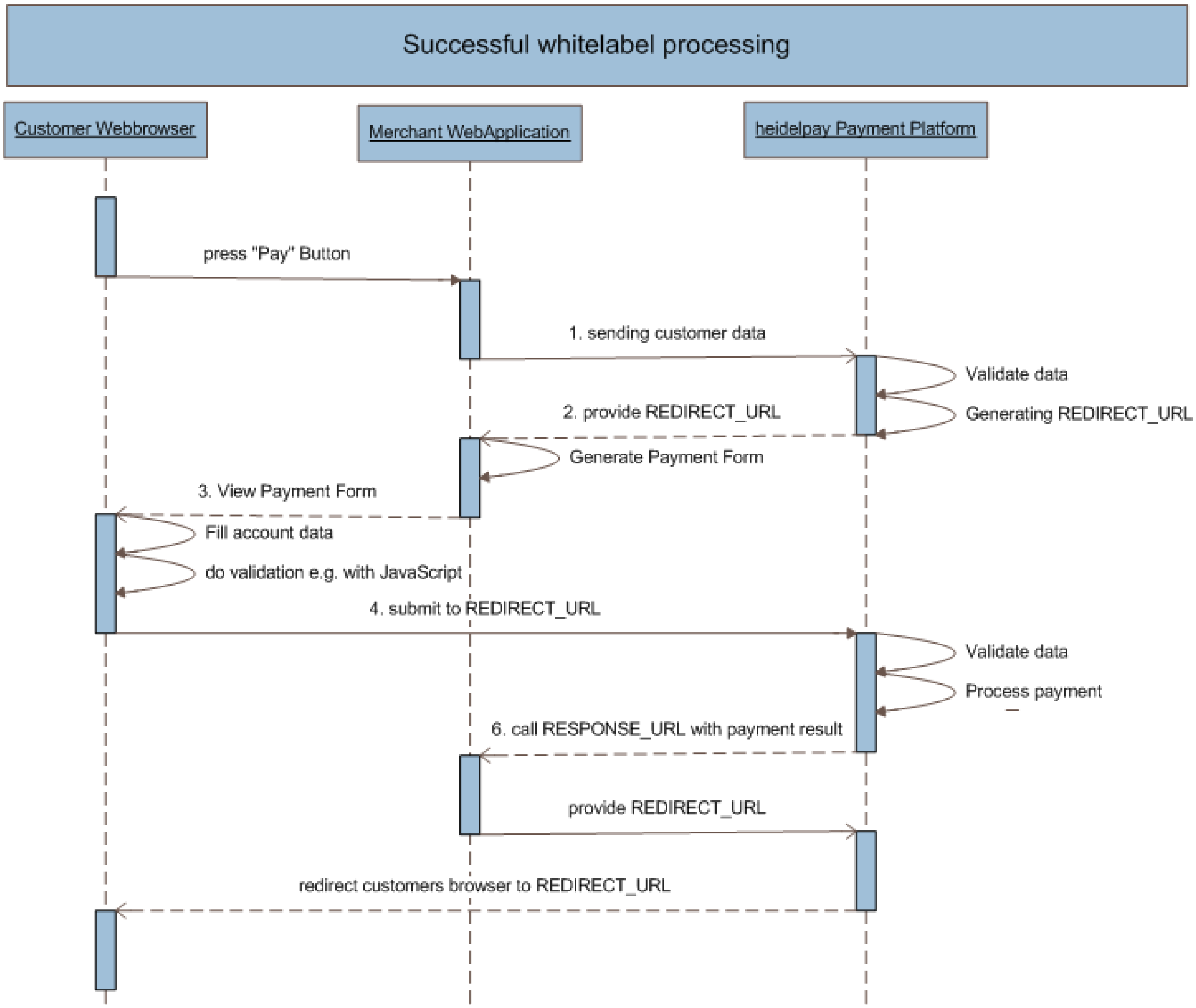

Figure 1 Processing of successful SEPA Debit processing

As you can see in Figure 1 the synchronous direct debit scenario step 5 doesn’t occur due to no third party interaction is needed.

4.1.2 First Request

The first step is sent a initial request to the post interface:

- For test system : https://test-heidelpay.hpcgw.net/ngw/post
- For live system : https://heidelpay.hpcgw.net/ngw/post

Your web application can send all customer data that is already known (such as the address). The following additional parameters need to be sent in the first request.

Parameter FRONTEND	Data type	Length	Mandatory	Description
FRONTEND.ENABLED	true	5	yes	Switches the whitelabel on or off.
FRONTEND.MODE	WHITELABEL	16	yes	User has to enter payment data.
FRONTEND.RESPONSE_URL	URL	256	yes	The URL where the shop wants to receive the async response
FRONTEND.SUCCESS_URL	URL	256	no	The URL where the customer should be sent if the transaction is successful AND there is an error when sending the response
FRONTEND.FAILURE_URL	URL	256	no	The URL where the customer should be sent if the transaction is not successful AND there is an error when sending the response

Especially the mandatory fields from the following POST-Parameter groups should be transmitted within the first request:

- Header
- Transaction
- User
- Identification Group
- Payment Group
- Customer Group

4.1.3 The First Response

If there is no validation error for the given parameters from the first request, the response will contain FRONTEND.REDIRECT_URL. Your web application needs to generate and display a HTTP- form with POST action method to the customer, with the returned FRONTEND.REDIRECT_URL as action URL and input fields for the configured payment method. Especially the members of the parameter group ACCOUNT, described in chapter 3.2.6 must be part of the payment form to complete the entire payment data after submitting the form to the FRONTEND.REDIRECT_URL. But which fields are required strongly depends on the chosen payment method and the workflow associated with it.

If for any reason your request could not be processed e.g. due to wrong currency code the response doesn’t contain the FRONTEND.REDIRECT_URL. The parameters of the processing group provide information about the occurred error.

4.1.4 Workflows

In the last chapter 4.1.3 there was described how your web application has to submit all necessary data to the **hPP** for payment transaction processing. The further proceeding depends on the payment method.

There must be distinguishing between two workflows:

- synchronous Workflows:
- asynchronous Workflows

4.1.5 synchronous Workflow SEPA example

Characteristic for the synchronous Workflow is, that the **hPP** can process (accept/decline) the payment process with the given data and completely without any more user interaction. This is the case for the payment methods Direct Debit,Credit Card (without 3dSecure), invoice and pre-payment (and subsequent debits in an recurring setup e.g. via paypal or mbe4).

An example for the processing of an SEPA Direct Debit Transaction follows.

First request to the post interface https://test-heidelpay.hpcgw.net/ngw/post

```
RESPONSE.VERSION=1.0
IDENTIFICATION.TRANSACTIONID=MerchantAssignedID

SECURITY.SENDER=31HA07BC8142C5A171745D00AD63D182
USER.LOGIN=31ha07bc8142c5a171744e5aef11ffd3
USER.PWD=93167DE7

TRANSACTION.MODE=CONNECTOR_TEST
TRANSACTION.RESPONSE=SYNC
TRANSACTION.CHANNEL=31HA07BC81714C94B2603CDF37C660F1

PAYMENT.CODE=DD.DB
PRESENTATION.AMOUNT=1.00 €
PRESENTATION.CURRENCY=EUR
IDENTIFICATION.TRANSACTIONID=4711MyId4711
NAME.GIVEN=Maximilian
NAME.FAMILY=Mustermann
ADDRESS.STREET=Vangerowstrasse 18
ADDRESS.ZIP=69115
ADDRESS.CITY=Heidelberg
ADDRESS.STATE=BY
ADDRESS.COUNTRY=DE

FRONTEND.ENABLED=true
FRONTEND.MODE=WHITELABEL
FRONTEND.RESPONSE_URL=http://myShop.com/payment/responseProcessor.jsp?id=4711MyId4711&action=response
FRONTEND.SUCCESS_URL=http://myShop.com/payment/responseProcessor.jsp?id=4711MyId4711&action=success
FRONTEND.FAILURE_URL=http://myShop.com/payment/responseProcessor.jsp?id=4711MyId4711&action=fail
```

In the case of valid request we should get a repose back which contains the following parameter:

```
POST.VALIDATION=ACK
PROCESSING.RESULT=ACK
FRONTEND.REDIRECT_URL= https://test-heidelpay.hpcgw.net/ngw/whitelabel?state=81684b51ffffffffbdbaa451
```

With this Information from the first response your web application can build a form which maybe looks like this one.

```
<form method="POST" action=" https://test-heidelpay.hpcgw.net/ngw/whitelabel">
     Account holder:<input type="text" name="ACCOUNT.HOLDER" value = "holder"/><br/>
     Iban: <input type="text" name="ACCOUNT.IBAN" value = ""/><br/>
     Bic: <input type="text" name="ACCOUNT.BIC" value = ""/><br/>
     Account country:<input type="text" name="ACCOUNT.COUNTRY" value = "DE"/><br/>
     E-Mail:<input name="CONTACT.EMAIL"/><br/>
     <input type="hidden" name="state" value="81684b51ffffffffbdbaa451"/>
--------------------------------------------------------------------------------------
```

```
  <br><input type="submit" name="pay" value="Pay now" /></form>
```

As you can see in the example form above, the naming of the input fields equals the post-parameter names described in chapter 3.2.

5 asynchronous Workflows

In comparison to standard transaction processing this process, especially due to the shopper authentication or other interaction with third party providers, is asynchronous and requires the merchant to provide an additional interface to accept the transaction result. The sequence diagram in Figure 2 shows an example of an asynchronous credit card payment with 3DSecure.

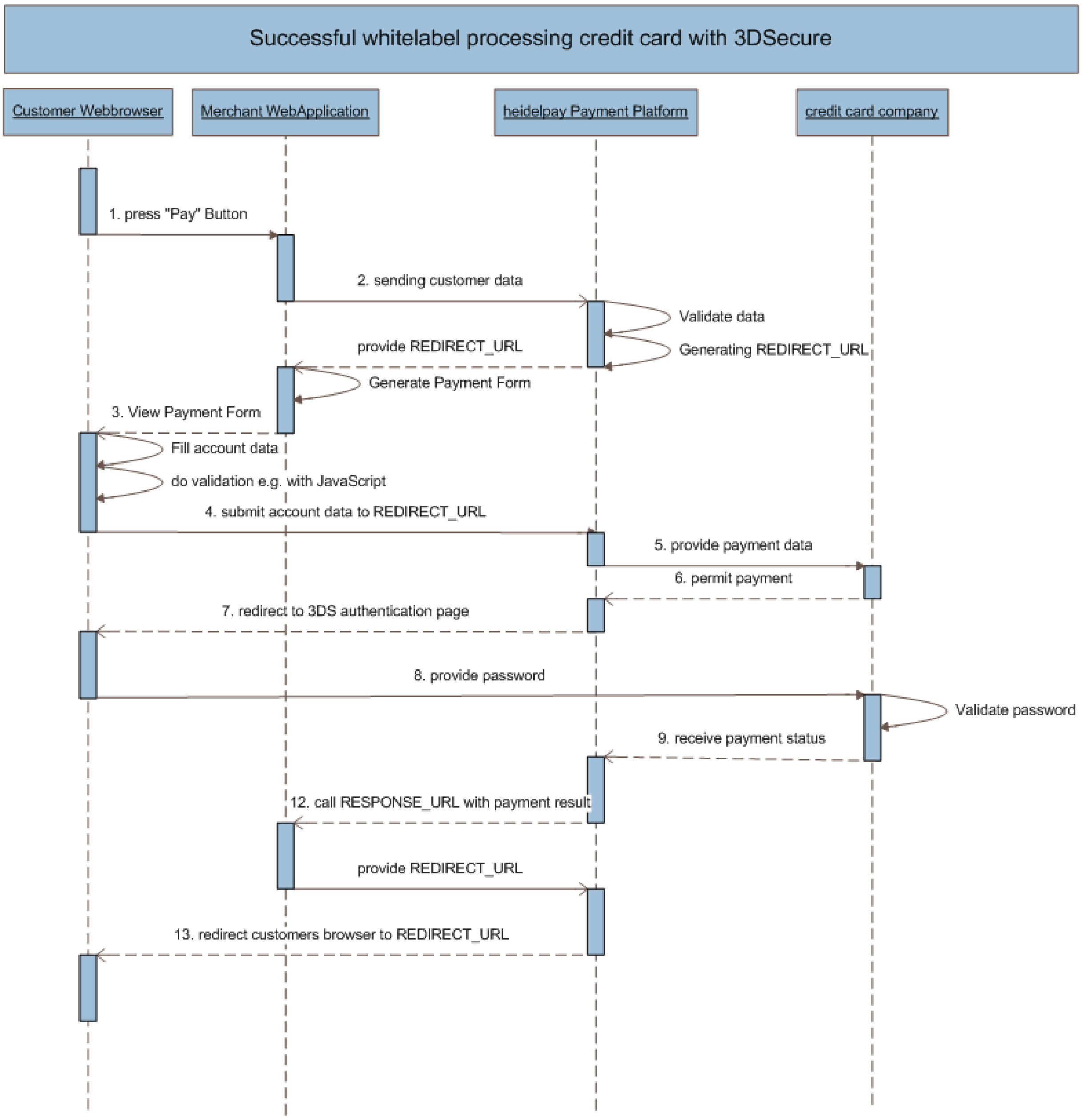

Figure 2 3DSecure credit card transaction

Step	Description
1	The customer browses through the merchant’s site, adds items to his shopping cart and finally purchases the goods.
2	The merchant sends a payment request to **hPP**.
3	The Merchant shows his customer a payment form with the former received redirect url as action url.
4	The customer is submit his account data directly to the **hPP**
5	If the CHANNEL is configured for an asynchronous workflow and the payment data is sufficient for the workflow, the payment transaction is switched into asynchronous mode. The payment system sends a request to the ACS (Access control server of the credit card company) and enquires whether the authentication process can be executed for this transaction.
6	The ACS’s response is returned to the payment system. If neither authentication nor proof of authentication attempt is available, asynchronous processing ends, and a synchronous authorization request, if appropriate, is executed. If the authentication attempt was successful, an URL and some transaction specific data are returned to the payment system.
7	The ACS receives the “Payer Authentication Request”. The customer’s browser is redirected automatically to the ACS web frontend.
8	The ACS authenticates the customer using processes applicable to PAN (password, TAN, PIN, etc.). Alternatively, the ACS may produce a proof of authentication attempt.
9	A Payer Authentication Response is sent to the **hPP** via a redirect from ACS.
10	In case the Authentication was successful, the original payment transaction is sent to the authorization system typically bank or acquirer(not shown in sequence diagram above)
11	Acquirer responds based on Authentication information and account data (not shown in sequence diagram above)
12	The payment system sends the asynchronous response message to the ResponseUrl the merchant provided in the initial payment request and redirects the end user’s browser to it.
13	The Merchant presents to the customer the information about the outcome of the authorization

5.1 Credit Card Transactions with 3DSecure

The most popular 3-D Secure Methods are:

- Verified by VISA(VbV)” and
- “Mastercard Secure Code (MSC)”

5.1.1 Recommendations and Requirements to 3DSecure enabled Merchants

Merchants must train their customer support staff on 3DSecure so that they can respond effectively to customer inquiries.
Good starting points are:

- Appendix A in Merchant Customer Service in http://www.mastercard.com/us/merchant/security/what_can_do/SecureCode/faq.html
- Online resources in
  http://www.mastercard.com/us/merchant/security/what_can_do/SecureCode/faq.html
  http://usa.visa.com/personal/security/visa_security_program/vbv/faq.html
- For using Mastercard SecureCode and Verified by VISA Logos and Program Identifier check:
  - Appendix in http://www.mastercard.com/us/merchant/security/what_can_do/SecureCode/getting_started.html and http://www.mastercardmerchant.com/securecode/artwork.html
- Chapter 6.8 in www.visa.com/verifiedmerchants

Merchants should be aware of some Guidelines and Best Practices set by Mastercard and VISA concerning their 3D Secure Implementation towards the end users.
Those guidelines include:

- display authentication not as a popups since browser nowadays usually block popups - see chapter 6.6 and 6.7 in www.visa.com/verifiedmerchants
- how to respond to Failed Authentication Processing (code 100.380.401 or 100.390.103) - see chapter 6.10 in www.visa.com/verifiedmerchants
- Mastercard requests merchants to notify customers about 3D Secure attempts processing (e.g. when the user tries to enroll during the transaction workflow)

To integrate and test on the test system, you may find test-data in “heidelpay-Integration Basics_en”

The following credit card numbers are configured as enrolled for VbV and MSC:

Brand	Number	Expiry	Verification
VISA	4711 1000 0000 0000	10 / 2008	123
MASTER	5453 0100 0005 9543	10 / 2010	345

5.1.2 asynchronous Workflow Credit Card with 3DS example

For example with the processing of a credit card transaction with 3DSecure the step no 5 from the generic workflow description in chapter 4.1.1 occurs. As you can see in Figure 2 the customer browser is redirected to the authentication page of the credit card company.

A complete example for an asynchronous credit card transaction with 3DS follows.

First request to the post interface https://test-heidelpay.hpcgw.net/ngw/post

```
IDENTIFICATION.TRANSACTIONID=MerchantAssignedID
SECURITY.SENDER=31HA07BC8142C5A171745D00AD63D182
USER.LOGIN=31ha07bc8142c5a171744e5aef11ffd3
USER.PWD=93167DE7
TRANSACTION.MODE=CONNECTOR_TEST
TRANSACTION.RESPONSE=SYNC
TRANSACTION.CHANNEL=31HA07BC8142C5A171749A60D979B6E4

IDENTIFICATION.SHOPPERID=customerid12345
IDENTIFICATION.INVOICEID=20090100012
PAYMENT.CODE=CC.PA
PRESENTATION.AMOUNT=1.00
PRESENTATION.CURRENCY=EUR
PRESENTATION.USAGE=Order Number 1234
NAME.GIVEN=Maximilian
NAME.FAMILY=Mustermann
ADDRESS.STREET=Vangerowstrasse 18
ADDRESS.ZIP=69115
ADDRESS.STATE=DE1
ADDRESS.COUNTRY=DE
ADDRESS.CITY=Heidelberg
FRONTEND.LANGUAGE=de

FRONTEND.ENABLED=true
FRONTEND.MODE=WHITELABEL
FRONTEND.RESPONSE_URL= http://myShop.com/payment/responseProcessor.jsp?id=4711MyId4711&action=response
FRONTEND.SUCCESS_URL=http://myShop.com/payment/responseProcessor.jsp?id=4711MyId4711&action=success
FRONTEND.FAILURE_URL=http://myShop.com/payment/responseProcessor.jsp?id=4711MyId4711&action=fail
```

This initial POST corresponds to the step no 1 mentioned in Figure 2.

In the case of valid request we should get a repose back which contains the following parameter:

```
POST.VALIDATION=ACK
PROCESSING.RESULT=ACK
FRONTEND.REDIRECT_URL= https://test-heidelpay.hpcgw.net/ngw/whitelabel?state=e2c9533cffffffffae0c837c
```

With this Information from the first response your web application can build a form which maybe looks like this one.

```
<form method="POST" action="https://test-heidelpay.hpcgw.net/ngw/whitelabel">
     Card number:
    <input type="text" name="ACCOUNT.NUMBER" value = ""/><br>
     Card brand:
    <select name="ACCOUNT.BRAND">
       <option value="VISA">Visa</option>
       <option value="MASTER">MasterCard</option>
```

```
      <option value="VISAELECTRON">Visa Electron</option>
    </select><br>
     Card holder: <input type="text" name="ACCOUNT.HOLDER" value ="" /><br>
     Expiry date:
    <select name="ACCOUNT.EXPIRY_MONTH" /><option>01</option><option>02</option>
      <option>03</option><option>04</option><option>05</option><option>06</option>
      <option>07</option><option>08</option><option>09</option><option>10</option>
      <option>11</option><option>12</option></select>
    <select name="ACCOUNT.EXPIRY_YEAR" /><option>2014</option><option>2015</option></select><br>
   CVV:<input type="text" name="ACCOUNT.VERIFICATION" value=""/><br>
    E-Mail:>input name="CONTACT.EMAIL" value=""/><br>
    IP:<input name="CONTACT.IP" value="123.123.123.123"/><
    State ID<input type="hidden" name = "state" value="e2c9533cffffffffae0c837c" />
    <br><br>
    -------------------------------------------------------------------------------------
    <br><input type="submit" name="pay" value=">>>>     Pay     <<<<" />
  </form>
```

After submitting this form to **hPP** step no 5 from Figure 2 (the redirection to the 3DSecure page of the credit card company) happens.

5.2 Online Bank Transfer

Typical examples for Online Bank Transfer are sofortüberweisung (www.sofort.com) in the European-Union, Giropay in Germany (www.giropay.de) and IDEAL in the Netherlands (www.ideal.nl).

Although these and many others have totally different technical interfaces and slightly different business workflows, they can all be used via the standard **hPP** Whitelabel interface.

5.2.1 Architecture Overview

In general the architecture is the same as described before in chapter 5, only the final communication back to the merchant is slightly different.

Figure 3 shows the adopted workflow for Online Bank Transfer, the differences to prior chapters start with step 12.

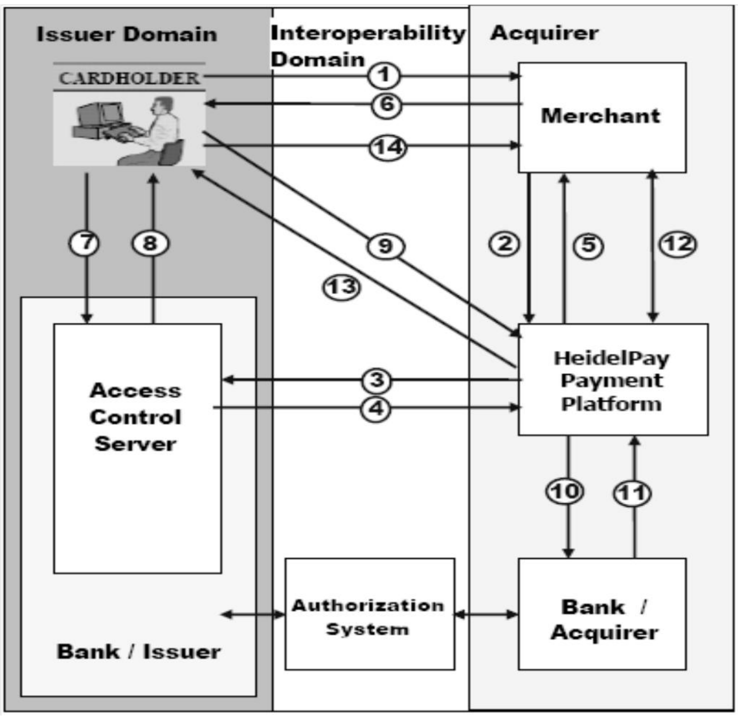

Figure 3 Adapted Architecture Overview for Online Bank Transfer

Step	Description
12	The payment system sends the asynchronous response message directly as XML to the ResponseUrl the merchant provided in the initial payment request. This message is URL encoded! Unlike described in 5.1 the browser is not directly redirected to this URL. Instead of that the merchant must reply to this payment response message with the URL the browser should be redirected to. This URL should be sent back as plain text without any formatting (just written to the output stream).
13	The payment system redirects the browser to the URL received at the end of step 12.
14	Merchant presents shopper information on authorization outcome

5.2.2 Online Transfer Workflow

The general workflow for an Online Transfer is very similar to a Prepayment process. This makes sense, as Online transfer is nothing else than Prepayment with the difference that the merchant receives immediate confirmation that the money was transferred to his account, whereas for Prepayment this usually takes one to three days.

The payment method used for Online Transfer is “OT”.

If the merchant wants to initiate an Online Transfer, he must send in a transaction with the payment type “PA” (similar to prepayment). Therefore the Payment code looks like:

```
PAYMENT.CODE=OT.PA
```

As soon as the end user has transferred the money, a referencing receipt with the payment code “OT.RC” is generated in the system and sent back to the merchant as a confirmation for the payment.

Be aware, that depending on the country of the online transfer process, different tags in the Account parameter group might be mandatory or not. Look at chapter 5.2.2.4, 5.2.2.5 and 5.2.2.6 for details on that.

5.2.2.1 Online Transfer Whitelabel Payment Request

If the provided channel is configured for Online Bank Transfer, the payment request must contain the Frontend subgroup. To test Online Bank Transfer you must use the mode CONNECTOR_TEST. If you use INTEGRATOR_TEST, the transaction will be handled synchronously without any redirection.

Like in the example in chapter 4.1.5 the instantiation of a payment for OT consists of two steps. The first step is to get a FRONTEND.REDIRECT_URL by providing already known customer data. Within this initial Request the frontend parameter group must provide the parameter FRONTEND.RESPONSE_URL.

The ResponseUrl is the URL that is called at the end of the asynchronous workflow to post the result of the payment to the merchant. As the main difference to the 3D-Secure process described in chapter 5.1 *the end user's browser is not redirected to this URL directly.* Instead, the merchant must reply to the post message sent to the response url with a URL as plain text. The **hPP** server will then redirect the end user's browser to the received URL.

The initial Request may contain the FRONTEND.SESSSIONID Parameter. The SessionID can be used by the merchant to identify the session of the end users's browser. It will be part of the asynchronous response message at the end to be able to reload the correct session for the end user. The SessionID will be part of the asynchronous message provided in the response parameter at the end of the payment process.

5.2.2.2 Online Transfer Whitelabel Payment Response

After submitting the Account data to the FRONTEND.REDIRECT_URL the validation of the supplied data takes place. If the transaction is validated successfully and the remote bank does support the online bank transfer for the specified bank and account, the payment transaction is switched to asynchronous mode. In this case the attribute "response" of the Transaction parameter in the response message is "ASYNC" and the response contains redirect information for the client browser. To log in at the third party system e.g. to the online banking web interface with username/password/PIN and commit the transaction with a TAN.

In case the transaction is not successful, the system replies with a direct payment response declining the OT.PA transaction.

NOTE: Although the result of this transaction is successful (ACK), it does not mean that the payment transaction will be successful. You have to wait for the asynchronous response message to get the final payment result.

In the successful case the response contains the following parameter.

```
PROCESSING.REDIRECT_URL
```

Your web application should use this URL for redirection of your customer's browser. Depending on which OT method you implement your customer finishes the transaction with login to the third party system and authorizing the payment. After your customer has entered the necessary data at the third party system, the **hPP** server is notified if the end user has successfully executed the transaction or not.

If the transaction could not be processed or nothing happens within a certain timeframe (depending on the online transfer process), the payment transaction is aborted.

In both cases the merchant is notified with an asynchronous response message about the result of the payment transaction. This message is posted to the URL initially specified in the PROCESSING.RESPONSE_URL parameter and is URL encoded.

The merchant must respond to this response message with a URL written plain to the output stream. The end user's browser is then redirected back to the merchant web application (to the URL finally received from the merchant).

Note:

Depending on the Online Transfer country this response optionally also contains the information which bank details were really used by the end user. This is especially helpful if you need the account for credits or refunds in the future!

5.2.2.3 Handling Timeouts

Depending on the Online Transfer payment method there are several scenarios when the final result of the online transfer cannot be determined right away and a timeout with the Online Transfer payment provider occurs.

- the shopper is redirected to the Online Transfer payment page and simply does not continue or closes the browser
- the shopper is redirected to the Online Transfer payment page, goes through the payment process, finishes the payment transactions and does not click the "Back to Shop" button.
- the Online Transfer payment provider experiences a down time in the middle of the payment process of a shopper.

In all those scenarios the shop will not be notified right away by the payment system. The payment platform will try to determine the state of the payment transaction in the background and asynchronously inform the merchant of the state of the payment. Depending on the problem that occurred in the background (e.g. a downtime of the provider) this might take hours until the final result is communicated back to the merchant.

Alternatively the merchant may implement the XML Query API (see separated documentation on that) to query the state of a certain transaction or to query all Receipts of the last (e.g. 30) minutes and do a reconciliation with the number of expected Receipts.

At anytime – of course – all Receipts (successful or rejected) can be found in the **HiP online platform** as well and can be exported by CSV.

5.2.2.4 EPS-Austria

TBD

5.2.2.5 IDEAL- Netherlands

TBD

5.2.2.6 Sofortüberweisung and paycode

TBD

5.2.2.7 Przelewy24

If the merchant wants to initiate a payment via Przelewy24 he has to send the payment code OT.PA like mentioned in chapter 5.2.2. This initial request must contain at last the NAME, ADRESS and ACCOUNT group in addition to the mandatory groups. In any case the frontend parameter group must provide the parameter FRONTEND.RESPONSE_URL.

The response contains a parameter PROCESSING_REDIRECT_URL and some PROCESSING_REDIRECT_PARAMETERS. It's up to the merchant to redirect the customer's web browser to the redirect URL with passing the received redirect parameters as HTTP-POST parameters. Przelewy24 presents the customer a selection of predefined payment methods (per merchant can be defined statically which methods are allowed).

Once the customer is successfully through the payment process the already in chapter 5.2.2.2 described online transfer workflow takes place.

For the "direct money transfer scenario" the receipt (OT.RC) is present a few seconds after the customer is through the payment process. The OT.RC-Response is what is passed to the URL received from the request to FRONTEND.RESPONSE_URL.

For the OT.RC in the "late money transfer scenario" an email or HTTP-POST notification can be set up. More about that can be found in the"Heidelpay-Push-Notification" document.

All Przelewy24 Transactions must be done with currency PLN.

To perform a refund the payment code is OT.RF. The IDENTIFICATION.REFERENCEID must point to the unique id of the receipt. Partial refunds are possible.

5.3 Virtual Account (VA) Payment

A typical examples for Virtual Accounts (Wallets or providers that let you choose from different payment methods) is PayPal.

Although these and many others have totally different technical interfaces and slightly different business workflows, they can all be used via the standard HeidelPay whitelabel interface.

5.3.1 Architecture Overview

In general the architecture is the very similar to what is described in prior chapters Figure 3 shows the adapted workflow for Virtual Account Payment.

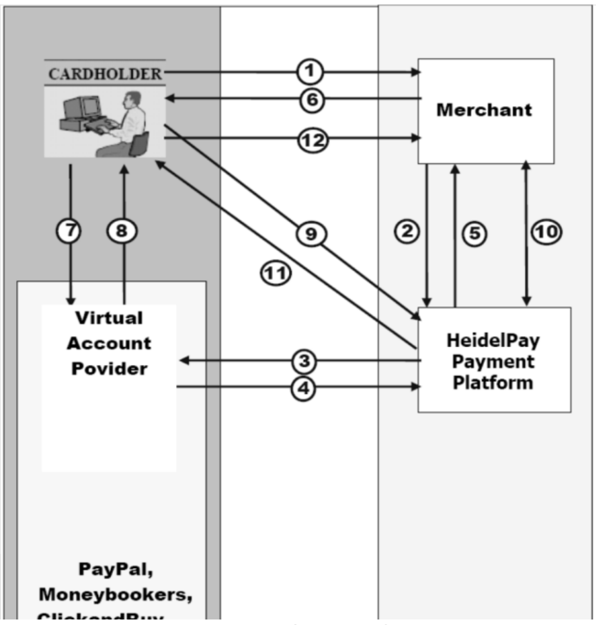

Figure 4 Adapted Architecture Overview for Virtual Accounts

Step	Description
1	Shopper browses at merchant site, adds items to shopping cart, then finalizes purchase.
2	Merchant sends payment request to the Payment Platform.
3	If the channel is configured for an asynchronous workflow and the payment data is sufficient for the workflow, the payment transaction is switched into asynchronous mode. The payment system sends a request to the Virtual Account provider
4	Virtual Account provider replies with URL to redirect and a number of necessary parameters that need to be posted to this URL by the browser.
5	In case of positive response from the Virtual Account provider a synchronous response to the original merchant request is sent back containing the redirect URL for the user. Besides that a number of parameters are returned as part of the XML response so that the merchant only has to add these parameters to the redirect to the user.
6	Merchant redirects the end user's browser to the redirect URL received in step 5. This might be done trough a JavaScript-based Redirect. Merchant may use inline windows or frames instead of pop-up windows to avoid confusion with shopper (pop-ups may be blocked).
7	Shopper enters his id and password for his Wallet account and goes through the necessary steps to finish the payment.
8	Virtual Account provider acknowledges payment.
9	Shopper is redirected back to the payment system with information about the payment status.
10	The payment system sends the asynchronous response message directly as XML to the ResponseUrl the merchant provided in the initial XML payment request. This message is URL encoded! Unlike described earlier the browser is not directly redirected to this URL. Instead of that the merchant must reply to this payment response message with the URL the browser should be redirected to. This URL should be sent back as plain text without any formatting (just written to the output stream).
11	The payment system redirects the browser to the URL received at the end of step 10.
12	Merchant presents shopper information on payment outcome.

5.3.2 VA Workflow

The general workflow for a Virtual Account is very similar to Online Transfer. The shopper is redirected to another page where he enters some data to finish the payment and is redirected back into the shop at the end.

The payment method used for Virtual Account is VA.

If the merchant wants to initiate a Virtual Account transaction, he must send in a transaction with the payment type DB Therefore the Payment tag looks like:

```
PAYMENT_CODE=VA.DB
```

Typically the Debit request will be in status WAITING first.

As soon as the end user has finished the payment process at the Virtual Account provider, the Debit is typically updated to either NEW (Successful) or REJECTED (Not successful). This final Debit response is then sent back to the merchant as a confirmation for the payment.

For more information on payment methods and types as well as the XML messages see the document “XML Transactions”.

5.3.3 VA Schedule Workflow with PayPal

To configure recurring payments via PayPal there are several steps to be done:

- Your web application has to send a registration (payment code VA.RG) with customer and payment data to the hPP
- The Response contains FRONTEND.REDIRECT_URL which your customers’ browser should be redirected to.
- The destination of the redirect is the PayPal page. Here your customer has to authenticate and agree with the payment.
- The result is posted to the FRONTEND.RESPONSE_URL your initial request has provided. If the registration is successful you receive the payment code VA.CF with the parameter IDENTIFICATION.REFERENCEID. The clients browser is redirected to the URL your application provide on the hPP request to FRONTEND.RESPONSE_URL
- To schedule the recurring payment your application has to send a job schedule (payment code VA.SD) which references the registration (IDENTIFICATION.REFERENCEID)

The registration process can be described in twelve steps as you can see in Figure 5

Step	Description
1	Shopper browses at merchant site, adds items to shopping cart, then finalizes purchase.
2	Merchant sends payment request to the hPP.
3	If the channel is configured for an asynchronous workflow and the payment data is sufficient for the workflow, the payment transaction is switched into asynchronous mode. The payment system sends a request to the Virtual Account provider
4	Virtual Account provider replies with URL to redirect and a number of necessary parameters that need to be posted to this URL by the browser
5	In case of positive response from the Virtual Account provider a synchronous response to the original merchant request is sent back containing the redirect URL for the user. Besides that a number of parameters are returned as part of the XML response so that the merchant only has to add these parameters to the redirect to the user.
6	Merchant redirects the end user’s browser to the redirect URL received in step 5. This might be done trough a JavaScript-based Redirect. Merchant may use inline windows or frames instead of pop-up windows to avoid confusion with shopper (pop-ups may be blocked).
7	Shopper enters his id and password for his Wallet account and goes through the necessary steps to finish the payment.
8	Virtual Account provider acknowledges payment.

9	Shopper is redirected back to the payment system with information about the payment status.
10	The payment system sends the asynchronous response message directly as XML to the ResponseUrl the merchant provided in the initial XML payment request. This message is URL encoded! Unlike described in earlier chapters the browser is not directly redirected to this URL. Instead of that the merchant must reply to this payment response message with the URL the browser should be redirected to. This URL should be sent back as plain text without any formatting (just written to the output stream).
11	The payment system redirects the browser to the URL received at the end of step 10.
12	Merchant presents shopper information on payment outcome.

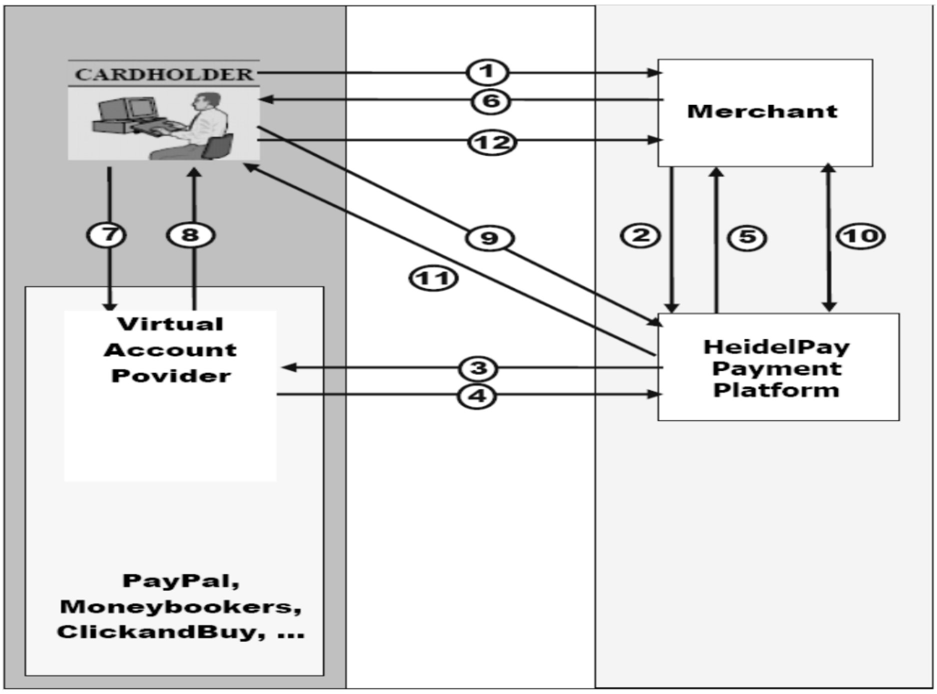
CARDHOLDER
Merchant
1
6
12
7
8
9
11
2
5
10
Virtual Account Povider
HeidelPay Payment Platform
3
4
PayPal, Moneybookers, ClickandBuy, ...

Figure 5 Registration process for Virtual Account

First request to the post interface https://test-heidelpay.hpcgw.net/ngw/post

```
RESPONSE.VERSION=1.0
IDENTIFICATION.TRANSACTIONID=MerchantAssignedID

SECURITY.SENDER=31HA07BC8142C5A171745D00AD63D182
USER.LOGIN=31ha07bc8142c5a171744e5aef11ffd3
USER.PWD=93167DE7
```

```
TRANSACTION.MODE=CONNECTOR_TEST
TRANSACTION.RESPONSE=SYNC
TRANSACTION.CHANNEL=31HA07BC8124365CA41D4BDA79CCCD22

PAYMENT.CODE=VA.RG
PRESENTATION.AMOUNT=1.00 €
PRESENTATION.CURRENCY=EUR
IDENTIFICATION.TRANSACTIONID=4711MyId4711
NAME.GIVEN=Maximilian
NAME.FAMILY=Mustermann
ADDRESS.STREET=Vangerowstrasse 18
ADDRESS.ZIP=69115
ADDRESS.CITY=Heidelberg
ADDRESS.STATE=BY
ADDRESS.COUNTRY=DE

FRONTEND.ENABLED=true
FRONTEND.MODE=WHITELABEL
FRONTEND.RESPONSE_URL=http://myShop.com/payment/responseProcessor.jsp?id=4711MyId4711&action=response
FRONTEND.SUCCESS_URL=http://myShop.com/payment/responseProcessor.jsp?id=4711MyId4711&action=success
FRONTEND.FAILURE_URL=http://myShop.com/payment/responseProcessor.jsp?id=4711MyId4711&action=fail
```

In the case of valid request we should get a repose back which contains the following parameter:

```
POST.VALIDATION=ACK
PROCESSING.RESULT=ACK
FRONTEND.REDIRECT_URL= https://test-heidelpay.hpcgw.net/ngw/whitelabel?state=81684b51ffffffffbdbaa451
```

As described in step 6 in Figure 5 the customers Browser should be redirected to FRONTEND.REDIRECT_URL. As soon as your customer has entered his PayPal login data and agreed the payment your application receive the transaction result.

```
ACCOUNT_BRAND=PAYPAL
CRITERION_PAYPAL_REG_TOKEN=EC-7LA73746L6002440J
IDENTIFICATION_REFERENCEID=31HA07BC815FABB68FDB9396645299C0
IDENTIFICATION_SHORTID=2712.3834.1368
IDENTIFICATION_UNIQUEID=31HA07BC815FABB68FDB4EBA8AC78ED7
NAME_GIVEN=Maximilian
NAME_FAMILY=Mustermann
PAYMENT_CODE=VA.CF
PRESENTATION_AMOUNT=1.00
PRESENTATION_CURRENC=  EUR
PRESENTATION_USAGE=Parameter
PROCESSING_CODE=VA.CF.90.00
PROCESSING_CONFIRMATION_STATUS=INITIAL
PROCESSING_REASON=SUCCESSFULL
PROCESSING_REASON_CODE=00
PROCESSING_REDIRECT_URL=https://www.sandbox.paypal.com/cgi-bin/webscr?useraction=commit&cmd=_express-
checkout&token=EC-7LA73746L6002440J
PROCESSING_RESULT=ACK
PROCESSING_RETURN=Request successfully processed in \'Merchant in Connector Test Mode\'
PROCESSING_RETURN_CODE=000.100.112
PROCESSING_STATUS=NEW
PROCESSING_STATUS_CODE=90
```

```
...
```

To setup a recurring debit of your customers PayPal account your application has to do a Job-Scheduling transaction with a reference to the registration transaction.

```
RESPONSE.VERSION=1.0
IDENTIFICATION.TRANSACTIONID=MerchantAssignedID

SECURITY.SENDER=31HA07BC8142C5A171745D00AD63D182
USER.LOGIN=31ha07bc8142c5a171744e5aef11ffd3
USER.PWD=93167DE7

TRANSACTION.MODE=CONNECTOR_TEST
TRANSACTION.RESPONSE=SYNC
TRANSACTION.CHANNEL=31HA07BC8124365CA41D4BDA79CCCD22

ACCOUNT.REGISTRATION=31HA07BC815FABB68FDB9396645299C0
```

```
PAYMENT.CODE=VA.SD
PRESENTATION.AMOUNT=1.00 €
PRESENTATION.CURRENCY=EUR
PRESENTATION.USAGE=a usage

JOB.NAME=TestJob1
JOB.ACTION=DB
JOB.EXECUTION_DAYOFMONTH
JOB.EXECUTION_DAYOFWEEK
JOB.EXECUTION_EXPRESSION
JOB.EXECUTION_HOUR
JOB.EXECUTION_MINUTE
JOB.EXECUTION_MONTH
JOB.EXECUTION_YEAR
```

@TODO

6 Appendix

6.1 POST.VALIDATION Codes

Validation Code	Validation name	Description
ACK	ACK	Request OK
2010	ERROR_AMOUNT	Parameter PRESENTATION.AMOUNT missing or not a number
2030	ERROR_CURRENCY	Parameter PRESENTATION.CURRENCY missing
2020	ERROR_PAYMENT_CODE	Parameter PAYMENT.CODE missing or wrong
3010	ERROR_FRONTEND_MODE	Parameter FRONTEND.MODE missing or wrong
3020	ERROR_FRONTEND_NEXT_TARGET	Parameter FRONTEND.NEXT_TARGET wrong
3040	ERROR_FRONTEND_LANGUAGE	Parameter FRONTEND.LANGUAGE wrong
3050	ERROR_FRONTEND_RESPONSE_URL	Parameter FRONTEND. RESPONSE_URL wrong
3070	ERROR_FRONTEND_POPUP	Parameter FRONTEND. POPUP wrong
3090	ERROR_FRONTEND_LINKS	Wrong FRONTEND.LINK parameter combination
3100	ERROR_FRONTEND_BANNERS	Wrong BANNERS information
3110	ERROR_FRONTEND_PM_METHOD	Wrong FRONTEND.PM_METHOD parameter combination
3120	ERROR_FRONTEND_BUTTON	Wrong FRONTEND.BUTTON parameter combination
4020	ERROR_IP	Parameter SECURITY.IP missing or wrong
4030	ERROR_SENDERID	Parameter SECURITY.SENDER missing or wrong
4040	ERROR_AUTHENTICATION	Wrong User/Password combination
4050	ERROR_USER	Parameter USER.LOGIN missing or wrong

4060	ERROR_PWD	Parameter USER.PWD missing or wrong
4070	ERROR_CHANNEL	Parameter TRANSACTION.CHANNEL missing or wrong
5010	ERROR_ACCOUNT_COUNTRY	Parameter ACCOUNT.COUNTRY is wrong or missing for WPF_LIGHT mode and payment code starts with DD

6.2 Login data to the Test system

6.2.1 Authentication

URL: https://test-heidelpay.hpcgw.net/ngw/whitelabel

Sender-ID: 31HA07BC8142C5A171745D00AD63D182

Login: 31ha07bc8142c5a171744e5aef11ffd3

Password: 93167DE7

6.2.2 Channels

Name	Kanal-ID
BillSafe	31HA07BC8142EE6D02715F4CA97DDD8B
MangirKart	31HA07BC8142EE6D0271011E4508C3F2
Moto	31HA07BC8199D92FA37A60592459C8EC
Postfinance	31HA07BC811E8AEF9AB2733D80C21DA8
Sofortueberweisung without account data	31HA07BC8142C5A171749CDAA43365D2
Credit card with 3D Secure	31HA07BC8142C5A171749A60D979B6E4
Credit card without 3D Secure	31HA07BC8142C5A171744F3D6D155865

6.2.3 Card Data

6.2.3.1 Credit card

Mastercard

Kartenummer:	5232050000010003
Ablaufdatum:	12.2014
CVV:	003
3D PWD:	secret3

VISA

Kartenummer:	4711100000000000

Ablaufdatum: 12.2014
CVV: 123
3D PWD: test123

6.2.3.2 Sofortüberweisung

BLZ = "88888888"
Kontonummer ="123456"
USER_PIN = "12345"
USER_TAN = "123456"

IBAN: DE06000000000023456789
BIC: SFRTDE20XXX

6.2.3.3 Giropay

BLZ = "12345679"
Kontonummer = "0000000300"
USER = "sepatest1"
USER_PIN = "12345"
USER_TAN = "123456"

IBAN: DE46940594210000012345
BIC: TESTDETT421

6.2.3.4 Lastschrift DE

Kto.-Nr. "1234567890"
BLZ "10000000"

Kto.-Nr. "5320130"
BLZ "37040044"
IBAN: "DE89370400440532013000"
BIC: "COBADEFFXXX"
Bank: Commerzbank

6.2.3.5 EPS

Login data:
username = 108256743
password = npydemo
Bank: EPS Testbank / "Erste Bank Connector Testystem"

TAN:
xx11111 always valid TAN
xx22222 always delivers the first error TAN
xx33333 always delivers the second error TAN
xx44444 always delivers the final try error TAN
xx55555 always delivers user is locked

6.2.3.6 Paypal

PayPal Buyer Accounts in alter Sandbox: (Kann an Händler weitergegeben werden)

Username: paypal-customer@heidelpay.de

Password: heidelpay