

CLASSIC MERCHANT API

SOPG (Service Oriented Prepaid Gateway - xml based protocol) Documentation

Table of contents

1	System overview.....	3
1.1	Prerequisites for using SOPG:	3
2	Classic payment overview.....	4
2.1	Life cycle of a normal payment transaction	4
3	SOPG Operations.....	7
3.1	Command and query operations	7
3.2	General Error Handling requirements.....	7
3.3	error messages description	7
3.4	Content-Type and charset.....	7
4	Payment notification to merchant's "pnUrl"	8
4.1	payment notification implementation.....	8
4.2	Technical specification	9
4.2.1	Definition of output parameters:	9
4.3	Resubmission of the payment notification	11
4.4	Timing of the payment notification	11
4.5	Security policy of the payment notification	11
5	Definition of paysafecard systems	12
5.1	paysafecard test environment	12
5.2	paysafecard productive environment.....	12
5.2.1	disposition time window	12
5.3	Operations Overview	13
5.4	Definition of parameters:	14
	payment functions – details	18
5.5	createDisposition	18
5.6	getCustomerPanel.....	20
5.6.1	paysafecard payment panel specification.....	21
5.6.2	paysafecard mobile payment panel specification	21
5.6.3	Usage of locale and language parameter	22
5.7	getMid (optional)	23
5.8	getSerialNumbers (optional).....	24
5.9	The following disposition states are returned by paysafecard:.....	24
5.10	modifyDispositionValue (optional)	25
5.11	executeDebit.....	26
6	Payment Test-Scenario	27
6.1	Payment Test-Scenario	27
6.2	createDisposition	28
6.3	getMID (optional)	29
6.4	getCustomerPanel.....	30
6.5	pnUrl request	31
6.6	executeDebit.....	32
6.7	Description of results:	33
7	Appendix A: Errorcodes.....	33

Contact:

For technical questions about the implementation please contact
techsupport@paysafecard.com

1 System overview

This document gives a detailed overview about the usage and parameters of paysafecard's Service Oriented Prepaid Gateway (SOPG).

The gateway is a SOAP XML Web Service which exposes API client functionalities that can be used with any SOAP capable client system.

If your webservice framework requires a WSDL at runtime, please download the SOPG WSDL from the WSDL URL and provide it from your local environment.

Do not fetch the WSDL from paysafecard servers at runtime.

1.1 Prerequisites for using SOPG:

- login credentials (user/password) provided by paysafecard
- authorization of your IP address (if you receive a "403 error" when trying to access the service, it is very likely because your IP address is not yet allowed to access)

PRODUCTIVE SYSTEM

- Service Endpoint URL
<https://soa.paysafecard.com/psc/services/PscService>
- Web Service Definition Language (WSDL) URL
<https://soa.paysafecard.com/psc/services/PscService?wsdl>

TEST SYSTEM

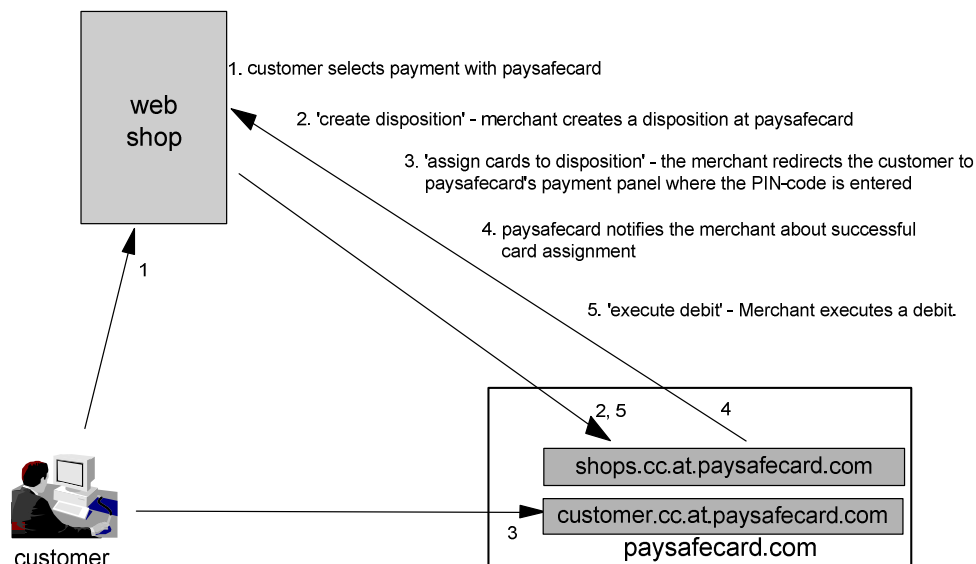
- Service Endpoint URL
<https://soatest.paysafecard.com/psc/services/PscService>
- Web Service Definition Language (WSDL) URL
<https://soatest.paysafecard.com/psc/services/PscService?wsdl>

2 Classic payment overview

In each payment there are three parties involved: a customer, a web shop (to whom we refer as "merchant") and the paysafecard ("PSC") company.

Payments take place in "payment transactions" or "dispositions", which are uniquely identified by a "merchant transaction ID" (MTID) and hold a value called "amount" that is typically the amount of money for which a customer buys something

2.1 Life cycle of a normal payment transaction



	SOPG Documentation Classic Merchant API	Document Version 2.2 August 30th, 2012
---	--	--

1. Customer selects paysafecard

When a customer indicates that they want to buy with paysafecard, this is typically done by clicking a button in the merchant's web shop.

2. Merchant creates the disposition and redirects the customer to paysafecard's payment panel

The merchant now initiates the payment process by sending a "createDisposition" request to paysafecard that creates a disposition at the paysafecard server. **It is the responsibility of the merchant to provide the unique "merchant transaction ID" (MTID) and remember it for future reference. In order to avoid multiple redemptions of promotional paysafecards, a "merchantclientid" has to be set. This parameter represents the customer's unique ID within the merchant's system.**

Now that paysafecard is notified about the payment, the merchant must forward the customer to the paysafecard payment panel (this is typically done automatically by redirecting the customer's web browser to the payment panel). Along with the forwarding HTTPS request, the merchant is responsible for providing the "merchant transaction ID" (MTID), "merchant ID" (MID) and some other parameters so that paysafecard can associate the incoming customer with the previously created disposition.

3. Customer assigns paysafecard to disposition

The customer is now presented with a page (payment panel) where they can enter the paysafecard PIN (16 digit voucher code). Upon completion, paysafecard sends the customer back to the merchant's site (the exact URL "**okURL**" is specified by the merchant during the creation of the payment transaction). The amount shall not be transferred in the "**okURL**".

4. paysafecard notifies about the state of the disposition

After successful assignment paysafecard notifies the merchant on their "pnUrl" about the state of the customer's disposition. This is an indicator that the merchant can debit the money, typically they thank the customer for the payment or invite them to shop further. **NOTE: At this point in time the money is reserved but not yet withdrawn from the paysafecard.**

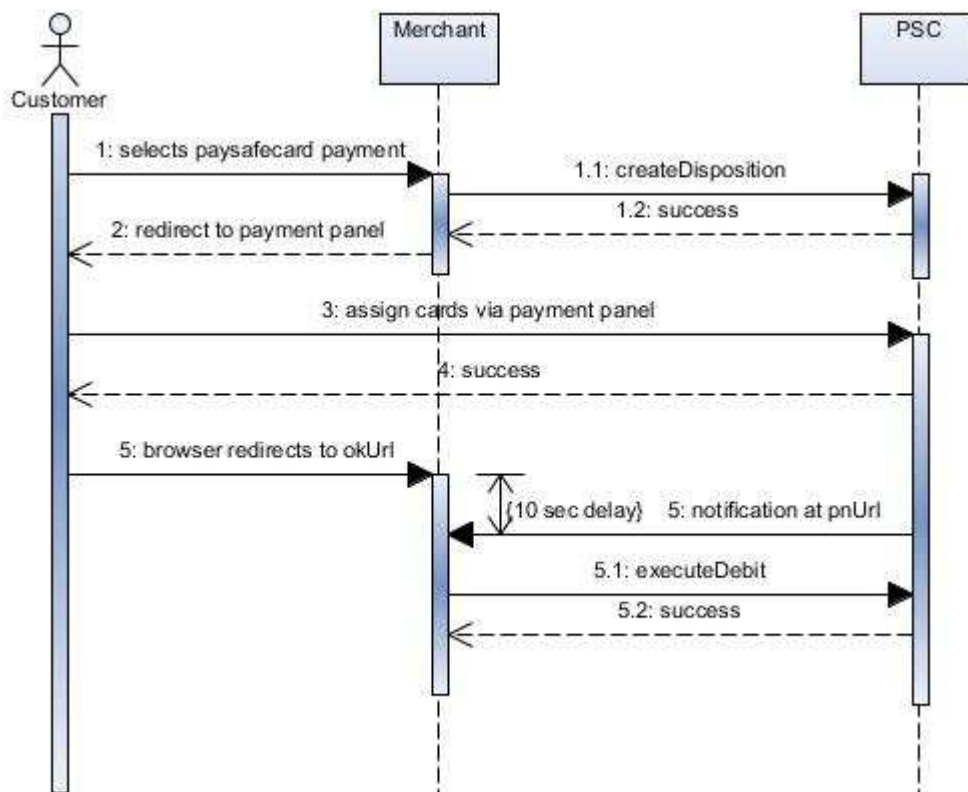
5. Merchant executes the debit

After successful assignment of cards the merchant now needs to send an "executeDebit" request and paysafecard transfers the money. Now it is possible to credit the customer's account or prepare the shipment of the ordered goods.

In case the merchant does not call the "executeDebit" request within a certain time period (disposition time window¹, which has to be agreed with paysafecard) the reserved amount will be credited back to the paysafecard and can be reused by the customer. The merchant has lost the amount and will not be able to capture it again.

For a basic payment procedure without delay, the merchant will send the "executeDebit" request with a "close=1" flag which indicates that no more activity will take place for this disposition.

The following sequence diagram below shows the sequence followed for payments.



3 SOPG Operations

The complete payment process is handled between the merchant-side system and paysafecard SOPG.

Although the SOPG WSDL service contract includes much more operations, please note that all required operations for the basic direct payment process are described in this document.

3.1 Command and query operations

The operations described in this chapter can be divided into query and command operations. A command operation will modify the state of an object whereas a query operation will return details about the current state of an object.

Command operations in SOPG are protected against unwanted side-effects of repeated executions (e.g. calling "executeDebit" twice on a transaction will return an error code on the second invocation).

3.2 General Error Handling requirements

All SOPG operations will return an "errorCode" and "resultCode".

A "resultCode" can have a value of "0" (successful), "1" (logical problem) or "2" (technical problem).

In general the following rules can be applied:

- **"1"** indicates that there is a problem with the submitted data (e.g. wrong credentials, transaction has expired, etc.) retries with the same request data won't be successful
- **"2"** (technical problem) means that the service is temporarily not available – the request can be retried

3.3 error messages description

Please consider that error messages to the customer will be only shown on the paysafecard payment panel. For other messages only the error code will be provided.

%1, %2, ... are only place holder for several values, eg. MTID, MID, ...

To get more information about the place holder you can contact us at techsupport@paysafecard.com

3.4 Content-Type and charset

Please make sure that the content type in the http header, when submitting requests, is set to:

Content-Type: text/xml; charset=UTF-8

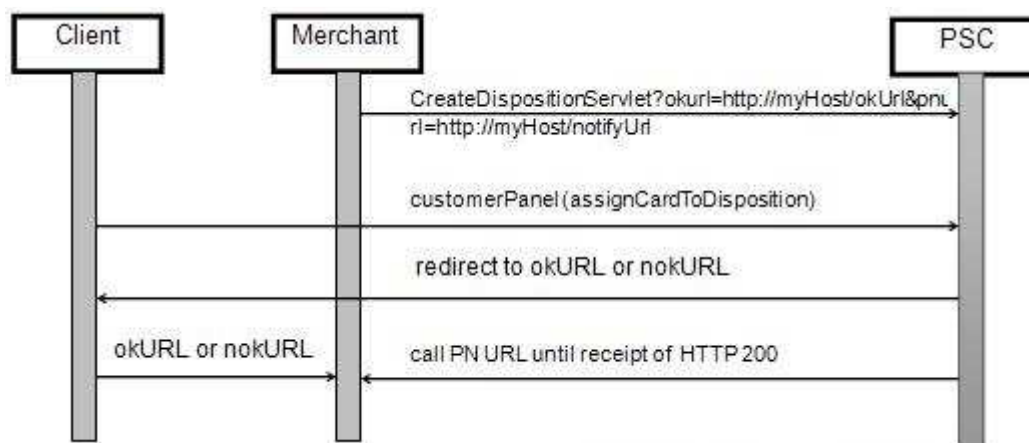
4 Payment notification to merchant's "pnUrl"

Payment notification gives notice on a successfully assigned paysafecard transaction independently from the customer's behavior after card assignment.

This service ensures that transactions can be completed before the top-up of customer's account and is therefore highly recommended in order to avoid incomplete transactions.

Payment notification URL, provided by the merchant, is submitted in addition to the "okURL" redirection by using "CreateDisposition" request.

In case of successful assignment, the notification is resubmitted to the merchant's webserver using a "HTTP POST" request as long as the webserver does not respond with a "HTTP 200".



4.1 payment notification implementation

- payment notification is enabled by submitting a "pnurl" parameter (where the "pnURL" is the URL that will receive the notification) when creating a disposition (by using "CreateDisposition" request)
- payment notification is only delivered in case of a successful card assignment
- payment notification is submitted **in addition** to the "okURL" redirection
- payment notification is resubmitted in case of technical or application errors on the merchants side

4.2 Technical specification

The payment notification is a simple "HTTP POST" request to the merchant's webserver.

The payment notification request contains the following parameters:

API function	Type	Output parameters	Merchant response elements
pnUrl	C	mtid eventType serialNumbers currency disposition amount cardTypeId	HTTP 200

4.2.1 Definition of output parameters:

- **mtid – (unique)** merchant transaction ID
 - max. length 60 characters
 - primarily provided by merchant
 - example: 3516-6s4dfsad41
- **eventType**
 - currently "ASSIGN_CARDS" is returned
- **serialNumbers** – the serialNumbers response contains semicolon separated details for each successfully assigned card
 - currency: ISO Currency Code
 - disposition amount: reserved amount of the customer's paysafecard associated with this disposition
 - cardTypeId: paysafecards are grouped into card types e.g. junior_paysafecard; adult_paysafecard; inhouse_paysafecard
 - examples:

```
0000000001200000;EUR;7.50;00002;
0000000001300000;EUR;5.50;00002;
```

The parameters are sent in request body in following format:

mtid=<Mtid>&eventType=<eventType>&serialNumbers=<serialNr1>;<currency1>;<amount1>;<cardTyp1>;<serialNr2>;<currency2>;<amount2>;<cardType2>;

The POST request is a default form POST request, i. e. with Content-Type "application/x-www-form-urlencoded".

Therefore the contents are URL encoded.

Please check this example for a URL encoded body:

mtid%3D3516-
6s4dfsad41%26eventType%3DASSIGN_CARDS%26serialNumbers%3D0000000001200000
%3BEUR%3B7.50%3B000002%3B0000000001300000%3BEUR%3B5.50%3B000002

If you decode this part you get:

mtid=3516-
6s4dfsad41&eventType=ASSIGN_CARDS&serialNumbers=0000000001200000;EUR;7.50;00
002;0000000001300000;EUR;5.50;00002

4.3 Resubmission of the payment notification

In case of technical errors (e.g. Socket timeout) or application errors (e.g. HTTP 500 response) at the merchant's side, the payment notification is resubmitted at a regular interval (approx. every 60 sec.)

Resubmission is performed until one of the following criteria is met:

- payment notification is successfully delivered (i.e. HTTP 200 response from merchant)
- max. number of retry attempts has been reached (currently configured: 3 retries)
- time window for retries has exceeded (currently configured: 10 minutes)

4.4 Timing of the payment notification

The payment notification will be submitted after the successful card assignment.

ATTENTION: If a "pnUrl" parameter is submitted as part of the create disposition request, the payment notification always will be delivered. If merchants perform the debit immediately after the "okURL" redirect, the payment notification will be sent anyway and can arrive before or after the merchant did already the debit.

4.5 Security policy of the payment notification

We suggest using **HTTP** instead of **HTTPS**. paysafecard does not send any sensitive data in payment notification request (only serial numbers with disposition amounts), these data must not need to be sent through SSL channels.

Standard ports for **HTTP (80)** and **HTTPS (443)** for payment notification URL should be used. **Please assume that our security policies do not support any other ports instead of standard ports.**

In case you would like to use HTTPS for payment notification URLs **(we kindly suggest using HTTP for payment notification URLs)**, only certificates created by world-known CAs (certificate authorities) are supported.

Self-issued certificates should not be used for payment notification URLs.

It would be also preferred not to use certificates where the host-name in certificate is different from host-name used in payment notification URL **(host-name verification policy)**.

5 Definition of paysafecard systems

5.1 paysafecard test environment

paysafecard provides the "paysafecard merchant test system" (SOATEST), a test environment for integration of new business partners.

Every merchant is integrated in this system first where the merchant support team provides support to new merchants as well as performs the acceptance tests.

Once the merchant integration is accepted, the merchant can be moved to the productive systems which, for the merchant, consist in the following steps:

- update to productive credentials (all provided by paysafecard)
- exchange of the Service Endpoint URL
- exchange of the WSDL URL

All data are provided by the merchant support team.

5.2 paysafecard productive environment

The productive system is identical to the test environment apart from:

- is only accessible with productive credentials
- disposition time window is enabled

5.2.1 disposition time window

Once a disposition is in status "DISPOSED", the merchants have to perform their debits within a certain time period (disposition time) which varies from merchant to merchant and is configurable by paysafecard. If this time window is exceeded, the disposition will automatically expire and the amount will be freed on the customer's paysafecard.

Furthermore, there is another job which sets all dispositions that have been created but not successfully debited to status "EXPIRED".

Please note, that these jobs are only active on the productive paysafecard servers.

5.3 Operations Overview

The following functions are available for a merchant:

Operation Name	Type	Request Elements	Response Elements
createDisposition	C	username [required] password [required] mtid [required] subId [required] amount [required] currency [required] merchantclientId [required] pnUrl [required] clientIp [required] dispositionrestrictions [required] shopId [required] shoplabel [required]	mtid, subId, mid, resultCode, errorCode
getMid ¹	Q	username [required] password [required] currency [required]	currency, mid, resultCode, errorCode
getSerialNumbers	Q	username [required] password [required] mtid [required] subId [required] currency [required]	mtid, subId, resultCode, errorCode, amount, currency, dispositionState, serialNumbers
executeDebit	C	username [required] password [required] mtid [required] subId [required] amount [required] currency [required] close [required] partialDebitId [optional]	mtid, subId, resultCode, errorCode
modifyDispositionValue ¹	C	username [required] password [required] mtid [required] subId [required] amount [required] currency [required]	mtid, subId, resultCode, errorCode,

¹ Optional operation

5.4 Definition of parameters:

Type:

C ... command operation

Q ... query operation

- **username** – custom account username
 - provided by paysafecard
- **password** – custom account password
 - provided by paysafecard
- **mtid – (unique)** transaction id
 - max. length 60 characters
 - recommended value up to 20 characters
 - provided by merchant
 - only the following is allowed: A-Z, a-z, 0-9 as well as – (hyphen) and _ (underline)
 - example: 3516-6s4dfsad41
- **subId** – mandatory parameter but value must be left empty if nothing else agreed
 - so called "reporting criteria", offers the possibility to classify transactions
 - max. length 8 characters
 - agreement with paysafecard needed
 - example: shop1
- **amount** – disposition amount
 - requested amount is not allowed to exceed 1000.00 EUR (or equivalent in different transaction currency) in value
 - max. 11 digit before – exactly 2 digit after decimal point
 - use a point as decimal separator
 - example: 100.00
- **currency** – disposition currency
 - max. length 3 characters all uppercase
 - ISO Currency Code
 - example: EUR
- **pnUrl – payment notification URL**
 - URL has to be absolute and URL encoded as they are sent as a parameter
 - URL has to be provided by the merchant
 - max. length 765 characters
- **okUrl** – the good case URL, here customer is sent back after successful entered PIN [16 digit paysafecard voucher code]
 - URL has to be absolute and URL encoded as they are send as a parameter
 - max. length 765 characters
- **nokUrl** – the bad case URL, here customer needs to be sent back after hitting "cancel" button
 - URL has to be absolute and URL encoded as they are send as a parameter
 - max. length 765 characters

IMPORTANT NOTE: It is crucial to send the "**okURL**", "**nokURL**" and optional "**pnUrl**" in an URL encoded (also called percent-encoding) form. Otherwise, the result will be a

	SOPG Documentation Classic Merchant API	Document Version 2.2 August 30th, 2012
---	--	--

wrong redirect of the customer to the merchant in addition to a possible failure of the payment.

- **merchantclientId** – a unique end customer identifier (e.g. the unique ID of the end customer as registered within the merchant database) or e-mail address
for promotional activities paysafecard checks the clientId and avoids multiple redemptions
 - **NOTE: for security reasons do not use the customer's registered username**
 - max. length 50 characters
 - example: client123
- **clientIp** – the IP address of the paysafecard customer
- **shopId** – Identification of the shop which is the originator of the request. This is most likely used by payment service providers who act as a proxy for other payment methods as well.
 - max. length 60 characters
 - recommended value up to 20 characters
 - provided by merchant
 - only the following is allowed A-Z, a-z, 0-9 as well as – (hyphen) and (underline)
 - example: 2568-B415rh_785
- **shopLabel** – Label or name of the shop which is the originator of the request, related to the "shopId". This is most likely used by payment service providers which act as a proxy for other payment methods as well.
 - max. length 60 characters
 - example: Pet Food Store Inc
- **mid** – merchant ID, unique ID of the merchant/currency pair
 - 10 digit long
 - provided by paysafecard
 - example: 1000001234
- **dispositionState** – current state of the disposition
- **serialNumbers** – serial number(s) of assigned paysafecard(s) by customer, after entering PIN at paysafecard payment panel
- **close** – the close flag of the disposition can be "0" or "1" to indicate if further actions will be executed or not
 - "0" [don't close transaction]
 - "1" [close transaction, this is the last debit]

- **locale** – is the locale of the payment page enabling country-specific interfaces ("en_UK", "de_DE" etc) – see chapter "Usage of locale and language parameter" for details about supported locale parameters and application behaviour when locale and language parameters are used
- **language** – Supported for backward compatibility only – language of the payment page ("en", "de", etc) - see chapter "Usage of locale and language parameter" for details about supported locale parameters and application behavior when locale and language parameters are used
- **partialDebitId** – this parameter allow the classification of partial debit
 - prerequisite: close flag=0
 - 1 digit long
 - example: 3
- **resultCode** – result code of the operation (see result codes chapter for details)
- **errorCode** – error code of the operation (see error codes chapter for details)

- **dispositionRestrictions** – disposition restrictions can be set by the merchant in order to restrict a payment transaction according to their individual needs.
 - multiple repeats possible
 - each restriction is a pair of key and value:
 - key – the name of the restriction
 - value – the value of the restriction

The following regional restrictions are available for a classic paysafecard payment without a paysafecard account (my paysafecard):

key	value example	possible values	description
COUNTRY	DE, FR, GR, AT	all countries where paysafecard is distributed	Restricts the payment to be fulfilled in i.e. germany. The value accepts ISO 3166-1 country codes.

The following restrictions are available for a paysafecard payment with a paysafecard account (my paysafecard):

key	value example	possible values	description
MIN_AGE	18	must be a positive number value	Restricts the my paysafecard user account to be at least 18 years old.
MIN_KYC_LEVEL	SIMPLE	SIMPLE or DOCUMENT or POSTIDENT	Restricts the my paysafecard user account to be at least in the state i.e.simple

Description of KYC-levels:

- **SIMPLE:**
 - text message verification (SMS)
 - e-mail verification
- **DOCUMENT:**
 - full document check with copy of e.g. passport ID
 - proof of address
- **POSTIDENT (only in germany enabled):**
 - full document check
 - form personally picked up at the post-office

payment functions – details

5.5 createDisposition

The merchant initiates the payment process by sending a "createDisposition" request to paysafecard that creates a disposition at the server.

IMPORTANT NOTE: maximum allowed amount per paysafecard transaction cannot exceed 1000.00 EUR (or equivalent in different transaction currency) in value.

The following data must be passed:

- "username": provided by paysafecard for the authentication
- "password": provided by paysafecard for the authentication
- "mtid": unique identifier for each disposition
- "subId": mandatory parameter but value must be left empty if nothing else agreed
- "amount": max. 11 digit before the decimal point – exactly 2 digit after the decimal point
- "currency": max. length 3 characters all uppercase [ISO CurrencyCode]
- "pnUrl": this is the URL at which paysafecard notifies the merchant as soon as assignment was successfully performed; merchant can now send "executeDebit" to withdraw the money from customer's paysafecards.
- "okURL": this is the URL to which the customers are forwarded by paysafecard after they successfully assigned their paysafecards and completed the payment. The merchant may include some information in the URL (e.g. the merchant Transaction ID) to get back the context of the payment transaction, e.g. to show an appropriate confirmation panel to the customer.
- "nokURL": this is the URL to which customers are forwarded by paysafecard when they click "cancel" on the paysafecard payment panel
- "merchantclientId" – max. length 50 characters; a unique end customer identifier (e.g. an account number/id of the end customer within the merchant's system; for security reasons it is not recommended to use the username of the client used for logon at the merchant, hash value of username would be fine)
- "shopId" – max. length of 60 characters; Identification of the shop which is the originator of the request. This is most likely used by payment service providers who act as a proxy for other payment methods as well

	SOPG Documentation Classic Merchant API	Document Version 2.2 August 30th, 2012
---	--	--

- "shopLabel" – max. length of 60 characters; Label or name of the shop which is the originator of the request, related to the "shopId".
- "dispositionRestrictions" – disposition restrictions can be set by the merchant in order to restrict a payment transaction according to their individual needs.

5.6 **getCustomerPanel**

The "getCustomerPanel" redirection allows the merchant to deliver the payment to the customer.

The customer has the choice to pay the disposition with one single PIN (16-digit paysafecard voucher code) or navigate to the "CustomerListPanel" where up to 10 PINs can be entered. Of course the customer can also "cancel" the payment transaction.

Depending on the outcome of the payment, the customer is then redirected to the merchant URL, which was sent at the "createDisposition" step.

The customer is forwarded to **"okURL"** in case of a successful card assignment to the disposition. This allows the merchant to immediately call "executeDebit" and complete the disposition.

In case the merchant cannot react on the call of the **"okURL"**, the "getSerialNumbers" request delivers the disposition state, but this may involve polling of the paysafecard servers, check the "getSerialNumbers" section for details.

If the customer hits "cancel" they get forwarded to the **"nokURL"**. This allows the merchant to know that the payment has failed and no further steps are needed.

The following data must be passed:

- "mid": unique ID of the merchant/currency pair
- "mtid": unique identifier for each disposition;
- "amount": max. 11 digit before the decimal point – exactly 2 digit after the decimal point
- "currency": max. length 3 characters all uppercase [ISO CurrencyCode]
- "subId": mandatory parameter but value must be left empty if nothing else agreed

	SOPG Documentation Classic Merchant API	Document Version 2.2 August 30th, 2012
--	--	--

5.6.1 paysafecard payment panel specification

The paysafecard payment panel can be presented in a popup window or alternatively in an iFrame.

To make sure the whole payment panel is visible to the user, please always allow vertical scrolling or dynamic sizing.

The width is fixed to **600px**

The height is fixed up to **1040px**

5.6.2 paysafecard mobile payment panel specification

paysafecard's payment panel is optimized for mobile devices, when a customer is using a mobile browser, paysafecard automatically redirects the customer to the optimized payment panel. The panel automatically uses the whole width of the device.

5.6.3 Usage of locale and language parameter

Basically, paysafecard customers are forwarded to the paysafecard payment panel according to their IP address, paysafecard uses a GeoIP service check.

It is not obligatory to set a locale parameter.

The parameter uses a combination of language and country code and will replace the language parameter in the long run, because it enables different payment interfaces depending on language AND country.

Please note:

- the language code must match ISO 639 standards
- the country code must match ISO 3166 standards

For backward compatibility all existing language parameters still yield the same result as in former versions of the API, but every language will be automatically transformed into a locale. We would therefore suggest to use only the locale parameter.

If more than one country exists for a language, a default country is defined; see examples below:

en → en_uk

de → de_de

The following rules apply for the usage of one or both parameters if the paysafecard GeoIP service check is disabled:

- If a **valid** locale is sent, it overrides any language parameter
- If no language **AND** no locale is sent, is set by default to "de_de"
- A **completely invalid** locale like "xy_xy" is set by default to "en_uk"
- In case of a **partly invalid** (or not yet supported) locale like "en_xy", the language part will still be used and the result page will be "en_uk"

Example:

locale	language	payment page displayed in
de_de	de	de_de (german "Germany")
de_at	de	de_at (german "Austria")
en_uk	en	en_uk (british english)
en_us	de	en_us (american english)

5.7 getMid (optional)

Every merchant currency has its own MID assigned.

With "getMid" the merchant can query the assigned MID for the requested currency.

The following data must be passed:

- "username": provided by paysafecard for the authentication
- "password": provided by paysafecard for the authentication
- "currency": max. length 3 characters all uppercase [ISO CurrencyCode]

5.8 getSerialNumbers (optional)

The "getSerialNumbers" function queries at the paysafecard server for the states of the transaction to verify the expected state before calling the next function in case peer implementation don't allow reacting on calls to **"okURL"**, **"nokURL"** or **"pnUrl"**.

"getSerialNumbers" is intended to be used if the merchant need to check the assigned serial numbers before executing the debit (e.g. to prevent usage of multiple "Promotional Campaign PIN's" for one user account)

The following data must be passed:

- "username": provided by paysafecard for the authentication
- "password": provided by paysafecard for the authentication
- "mtid": unique identifier for each disposition
- "subId": mandatory parameter but value must be left empty if nothing else agreed
- "currency": max. length 3 characters all uppercase [ISO CurrencyCode]

5.9 The following disposition states are returned by paysafecard:

One letter code	Meaning	Description
R	Created	the disposition has successfully been created. If nothing happens within 30 minutes the disposition will be transferred to state "X" by the paysafecard cleanup job
S	Disposed	customer's paysafecard have successfully been assigned to the disposition, the merchant can "executeDebit", no debits have taken place so far
O	Consumed	final debit with close=1 has been called, no further debits are possible
L	Cancelled	the disposition has actively been cancelled by the customer
X	Expired	the time window for this disposition has ended (either before paysafecard were assigned or before "executeDebit" call)
E	Debited	partially debited, the disposition is still open, further debits are possible

5.10 modifyDispositionValue (optional)

The merchant can reduce the originally disposed amount with the "modifyDispositionValue" request, e.g. if they cannot ship the ordered goods, they can partially cancel a part of the order. The disposition has to be in status "disposed" or "partially debited".

IMPORTANT NOTE: this functionality can only be made on disposition where customer already assigned their paysafecard (dispositionstate="S")

The following data must be passed:

- "username": provided by paysafecard for the authentication
- "password": provided by paysafecard for the authentication
- "mtid": unique identifier for each disposition
- "subId": mandatory parameter but value must be left empty if nothing else agreed
- "amount": max. 11 digit before the decimal point – exactly 2 digit after the decimal point
- "currency": max. length 3 characters all uppercase [ISO CurrencyCode]

The response represent result of the disposition amount modification, no query of the disposition state is needed!

5.11 executeDebit

The "executeDebit" function performs the process of withdrawing money from the customer's paysafecard.

The merchant can control how the money is debited from the customer's paysafecard.

The most common case is a complete debit of the disposition amount declared in the "createDisposition" step with a close flag set to "1" to close the disposition.

Another possibility is to debit only a part of the disposition amount declared in the "createDisposition" step and keep the disposition open for further debits as the merchant wants to debit only delivered products or service. To do so, the "executeDebit" call needs a "close" flag to be set to "0".

NOTE: while dispositions are kept open, the associated amount is reserved and the customer cannot access that amount of this card until the end of the disposition time. In case the merchant wants to close the disposition without executing anything, the "executeDebit" must be called with amount=0.00

This step concludes the payment if the "close" flag is set to "1".

For each disposition <x> debits can be executed. Each debit reduces the open disposition amount. It will differ from merchant to merchant whether the whole disposition amount will be withdrawn by a single debit or several debits will be executed.

The following data must be passed:

- "username": provided by paysafecard for the authentication
- "password": provided by paysafecard for the authentication
- "mtid": unique identifier for each disposition
- "subId": mandatory parameter but value must be left empty if nothing else agreed
- "amount": max. 11 digit before the decimal point – exactly 2 digit after the decimal point
- "currency": max. length 3 characters all uppercase [ISO CurrencyCode]
- "close": indicator if further debits will be executed or not. If the close flag is "1" the disposition will be set to totally consumed and no further debits are possible.

The response represent result of the debit execution, no query of the disposition state is needed!

6 Payment Test-Scenario

The following scenario is a test scenario including example data. In practice, it will differ from merchant to merchant whether one or more debits or status-requests will be executed.

Do not use the enclosed example data!

For testing each merchant will receive a consistent set of test data.

6.1 Payment Test-Scenario

The scenario described in the following chapter contains the following steps:

- create disposition: The merchant creates a disposition, which means the customer initiated the payment process.
- get MID: The merchant makes sure using the correct MID for the redirection of the customer to the paysafecard payment panel.
- get customer panel: Redirection to paysafecard customer panel. One or more paysafecards are assigned to the disposition. This is carried out by the customer. The respective amount is reserved on the paysafecard.
- payment notification to merchant's "pnUrl": paysafecard notifies the merchant on their "pnUrl" about the current status of a payment transaction.
- execute debit: An "executeDebit" request is performed by the merchant. The debit represents the actual withdrawal of credit from the paysafecard.

6.2 createDisposition

Who:

Merchant

Description:

The merchant creates a disposition by sending the "createDisposition" request passing all necessary parameters.

Example Request:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:urn="urn:pscservice">
  <soapenv:Header/>
  <soapenv:Body>
    <urn:createDisposition>
      <urn:username>USER</urn:username>
      <urn:password>PASSWORD</urn:password>
      <urn:mtid>18b02d230-a6822f-4cbb-ae9-0bc07d90cfa4</urn:mtid>
      <!--Zero or more repetitions:-->
      <urn:subId></urn:subId>
      <urn:amount>10.00</urn:amount>
      <urn:currency>EUR</urn:currency>
      <urn:okUrl>http%3a%2f%2fwww%2epaysafecardokURL%2ecom</urn:okUrl>
      <urn:nokUrl>http%3a%2f%2fwww%2epaysafecardnokURL%2ecom</urn:nokUrl>
      <!--Optional:-->
      <urn:merchantclientid>cID_919191</urn:merchantclientid>
      <!--Optional:-->
      <urn:pnUrl> http%3a%2f%2fwww%2emerchantpnURL%2ecom </urn:pnUrl>
      <!--Zero or more repetitions:-->
      <urn:dispositionRestrictions>
        <urn:key>COUNTRY</urn:key>
        <urn:value>FR</urn:value>
      </urn:dispositionRestrictions>
      <urn:dispositionRestrictions>
        <urn:key>MIN_AGE</urn:key>
        <urn:value>18</urn:value>
      </urn:dispositionRestrictions>
      <!--Optional:-->
      <urn:shopId>3516-6s4dfsad41</urn:shopId>
      <!--Optional:-->
      <urn:shopLabel> Pet Food Store Inc.</urn:shopLabel>
    </urn:createDisposition>
  </soapenv:Body>
</soapenv:Envelope>
```

Example Response:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns1:createDispositionResponse xmlns:ns1="urn:pscservice">
      <ns1:createDispositionReturn>
        <ns1:mtid>18b02d230-a6822f-4cbb-ae9-0bc07d90cfa4</ns1:mtid>
        <ns1:mid>1000001234</ns1:mid>
        <ns1:resultCode>0</ns1:resultCode>
        <ns1:errorCode>0</ns1:errorCode>
      </ns1:createDispositionReturn>
    </ns1:createDispositionResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

6.3 getMID (optional)

Who:

Merchant

Description:

Every merchant currency has its own MID assigned.

With "getMID" the merchant can query the assigned MID for the requested currency.

Example Request:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:urn="urn:pscservice">
  <soapenv:Header/>
  <soapenv:Body>
    <urn:getMid>
      <urn:username>USER</urn:username>
      <urn:password>PASSWORD</urn:password>
      <urn:currency>EUR</urn:currency>
    </urn:getMid>
  </soapenv:Body>
</soapenv:Envelope>
```

Example Response:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns1:getMidResponse xmlns:ns1="urn:pscservice">
      <ns1:getMidReturn>
        <ns1:currency>EUR</ns1:currency>
        <ns1:mid>1000001234</ns1:mid>
        <ns1:resultCode>0</ns1:resultCode>
        <ns1:errorCode>0</ns1:errorCode>
      </ns1:getMidReturn>
    </ns1:getMidResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

6.4 getCustomerPanel

Who:

Customer

Prerequisite:

The command "createDisposition" was successfully executed and returned resultCode=0 and errorCode=0. Thus, the customer can be forwarded to the paysafecard payment panel for assigning cards to the disposition.

Description:

One or more cards are assigned to the created disposition.

If customer hits "pay now" button, they will be forwarded to the "okURL" which is defined by the merchant.

If customer hits "cancel" button without assigning one or more paysafecards to the disposition, they will be forwarded to the "nokURL" which is defined by the merchant

Example URL and parameters:**TEST SYSTEM**

[https:// customer.test.at.paysafecard.com/pssccustomer/GetCustomerPanelServlet](https://customer.test.at.paysafecard.com/pssccustomer/GetCustomerPanelServlet)

PRODUCTIVE SYSTEM

[https:// customer.cc.at.paysafecard.com/pssccustomer/GetCustomerPanelServlet](https://customer.cc.at.paysafecard.com/pssccustomer/GetCustomerPanelServlet?mid=1000001234&mtid=18b02d230-a6822f-4cbb-ae9-0bc07d90cfa4&amount=10.00¤cy=EUR&language=de&locale=de_at)
?mid=1000001234
&mtid=18b02d230-a6822f-4cbb-ae9-0bc07d90cfa4
&amount=10.00
¤cy=EUR
&language=de (optional parameter)
&locale=de_at (optional parameter)

Input parameters:

PIN: 0000 0000 1234 5678
Terms of Use: <checkbox, default unchecked>

6.5 pnUrl request

Who:

paysafecard system

Prerequisite:

The command "createDisposition" and card assignment were successfully executed.

Description:

paysafecard system sends an "HTTP POST" request to the merchant's system ("pnUrl") in order to notify about the successful assignment of the customer's paysafecards.

URL and parameters:

```
http://www.merchantpnURL.com/notifyME
    ?mtid=3516-6s4dfsad41
    &eventType=ASSIGN_CARDS
    &serialNumbers=0000000001200000;EUR;100.00;00002
```

Input:

```
mtid:          3516-6s4dfsad41
eventType:     ASSIGN_CARDS
serialNumbers: 0000000001200000;EUR;100.00;00002
```

Merchant's systems return status code:

HTTP 200

6.6 executeDebit

Who:

Merchant

Prerequisite:

The assignment of the paysafecards to disposition was executed successfully.

Description:

The debiting is the process of withdrawing money from the customer's paysafecard.

Example Request:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:urn="urn:pscservice">
  <soapenv:Header/>
  <soapenv:Body>
    <urn:executeDebit>
      <urn:username>USER</urn:username>
      <urn:password>PASSWORD</urn:password>
      <urn:mtid>18b02d230-a6822f-4cbb-ae9-0bc07d90cfa4</urn:mtid>
      <!--Zero or more repetitions:-->
      <urn:subId></urn:subId>
      <urn:amount>10.00</urn:amount>
      <urn:currency>EUR</urn:currency>
      <urn:close>1</urn:close>
    </urn:executeDebit>
  </soapenv:Body>
</soapenv:Envelope>
```

Example Response:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns1:executeDebitResponse xmlns:ns1="urn:pscservice">
      <ns1:executeDebitReturn>
        <ns1:mtid>18b02d230-a6822f-4cbb-ae9-0bc07d90cfa4</ns1:mtid>
        <ns1:subId/>
        <ns1:resultCode>0</ns1:resultCode>
        <ns1:errorCode>0</ns1:errorCode>
      </ns1:executeDebitReturn>
    </ns1:executeDebitResponse>
  </soapenv:Body>
</soapenv:Envelope>
```


6.7 Description of results:

Result Name	Result Description
resultcode	0 : successful, 1 : logical problem, 2 : technical problem
errorcode	contains an error number if resultcode is not equal to 0.

7 Appendix A: Errorcodes

If an error code appears that is not listed here, please contact techsupport@paysafecard.com

general messages - errors: 0001 - 0141

1=No data selected.
 2=%1 is not numeric.
 3=Mandatory field %1 is empty.
 4=Decimal field with name %1 and value %2 has no decimal point.
 5=Decimal field with name %1 and value %2 has no digits before the decimal point.
 6=Decimal field with name %1 and value %2 has too many digits before the decimal point (max. %3 allowed).
 7=Decimal field with name %1 and value %2 has too few digits after the decimal point (must have %3).
 8=Decimal field with name %1 and value %2 has too many digits after the decimal point (max. %3 allowed).
 9=Decimal field with name %1 and value %2 is not a number with format N.M (where N are 1 to %3 digits, M are exactly %4 digits and M and N are numeric).
 10=Decimal field with name %1 and value %2 has no digits after the decimal point (must have at least 1 and at most %3).
 11=Decimal field with name %1 and value %2 must not be negative.
 12=Decimal field with name %1 and value %2 is not a number with format N.M (where N are 1 to %3 digits, M are 1 to %4 digits and M and N are numeric).
 13=Decimal field with name %1 is empty.
 14=Cannot process more than %1 objects per transaction.
 15=Answer to Challenge Question is empty.
 16=Answer to Challenge Question is wrong.
 17=Answer to Challenge Question contains invalid characters.

 20=Challenge Question is empty.
 21=Challenge Question with value %1 is too long (max. %2 characters are allowed).

 50=Merchant ID is empty.
 51=Merchant ID with value %1 is too long (max. %2 characters are allowed).

 55=Merchant-transaction ID is empty.
 56=Merchant-transaction ID with value %1 is too long (max. %2 characters are allowed).

 60=nokURL is empty.

 65=okURL is empty.

 75=Serial number is empty.
 76=Serial number with value %1 is too long (max. %2 characters are allowed).
 77=Serial number %1 is not numeric.

 80=Card State %1 is invalid.
 81=Submitted Card State %1 of field %2 is not equal to the expected Card State %3.

 85=Card Type %1 is invalid.

 90=Debit State %1 is invalid.
 95=Disposition State %1 is invalid.
 96=Submitted Disposition State %1 of field %2 is not equal to the expected Disposition State %3.

 120=Close Debit-flag %1 is invalid (must be 0 or 1).

 125=Currency is empty.
 126=Currency with value %1 has invalid length (must have 3 characters).

140=Currency Name is empty.

141=Currency Name with value %1 is too long (max. %2 characters are allowed).

general messages - success messages 0601 - 0603

601=The command completed successfully.

602=The command completed successfully, no data found.

603=The command completed successfully, more data match filter criteria (change filter criteria to reduce amount of data returned).

card messages - error messages: 1001 - 1600

1004=Card with Serial Number %1 has an unexpected state %2, expected is %3.

1005=Card with Serial Number %1 has not a location "%3", but is at "%2".

1006=Card with Serial Number %1 is not assigned to %2.

1007=Card with Serial Number %1 does not exist.

1008=Access denied.

1009=%1 is not allowed to have Cards assigned.

1012=Card State %1 is not valid for this request, expected Card State is %2.

1015=Access denied because of a repeated recent access violation.

1020=Challenge Question Answer 1 and Challenge Question Answer 2 are different.

1025=Card is in an invalid State, expected State is 'GENERATED'.

1026=Number of copies printed is invalid, Card is set to State 'INVALID'.

1029=You need to specify question, answer and answer verification to set the Challenge Question.

1035=The card status of at least one card used for this inquiry is not valid.

1046=There is currently no available credit on this card. In the case of reserved amounts.

1049=At least one of the PINs used is not valid.

1050=Card does not exist

payment messages - error messages: 2001 - 2600

2001=Transaction (%1/%2) already exists. Please contact your webshop.

2002=Transaction (%1/%2) does not exist. Please contact your webshop.

2003=Transaction (%1/%2) is in invalid state %3, expected is %4.

2004=Insufficient funds for payment, open amount is %1.

2006=Transaction currency %1 is invalid for transaction (%2/%3). Please contact your webshop.

2007=The amount %1 is invalid for the used card.

2008=The amount %1 exceeds the card balance.

2009=The amount %1 is invalid for the transaction (%2/%3). Please contact your webshop.

2010=The amount %1 is insufficiently disposed for the transaction (%2/%3).

2011=The Currency %1 is invalid for this transaction, expected is %2.

2012=Payment transaction failed.

2013=Error finding transaction: %1.

2014=No disposition has been made for this payment transaction.

2015=Payment transaction failed.

2016=Error finding Merchant: %1.

2017=Transaction (%1/%2) is in invalid State %3, expected is %4 or %5.

2018=MicroDebits for transaction (%1/%2) are not sequential.

2019=MicroDebit for transaction (%1/%2) does not exist.

2020=Business type of transaction (%1/%2) is %3. Amount cannot be modified.

2021=The amount %1 is invalid for the transaction (%2/%3). The minimum transaction amount is %4.

2022=Transaction (%1/%2) has no cards assigned.

2023=Business type of transaction (%1/%2) is %3, expected: %4. Please contact your webshop.

2024=Debit cannot be performed, business type of transaction (%1/%2) is %3.

2025=Transaction amount is empty. Please contact your webshop.

2026=Transaction amount %1 is not numeric. Please contact your webshop.

2027=Transaction amount %1 is invalid. Please contact your webshop.

2028=Business type %1 is invalid for transaction.

2029=An error has occurred with this transaction – the amount must be greater than zero.

2039=Invalid restriction parameters.

2044=customerdetailsrequested {0} is invalid (must be 0 or 1).

2623=Shop ID is greater than 60 characters.

2624=Shop Label is greater than 60 characters.

payment messages - success messages: 2601 - 2900

2601=Payment completed successfully.

2602=Command completed successfully. No transactions found.

master reference - error message: 3001 - 3600

3001=Merchant %1 is not active. Please contact your webshop.

3002=Currency %1 is not valid for merchant %2. Please contact your webshop.

3003=Merchant %1 does not exist. Please contact your webshop.

3006=Card Type %1 is not accepted by the merchant.

3007=Merchant %1 exceeded time window to debit the transaction.

3012=Merchant %1 exceeded time window for Micropayment.

3013=Merchant %1 already exists.

3014=Reporting Criterion %1 for Merchant %2 doesn't exist.

3015=Reporting Criterion %1 for Merchant %2 is in state %3, expected %4 or %5.

feature messages: 3901 - 4000

3901=Feature with primary key (%1 %2 %3) cannot be found.

3902=The user %1 is not allowed to access the feature %2.

merchant API technical messages - error messages: 4001 - 4600

4001=SSL error.

4002=Invalid function request.

4003= above maximum disposition amount (1000.00 EUR or equivalent in different transaction currency)

4004=Invalid proxy request.

4005=Connection error.

4006=Unexpected response from server.

4007=Undefined error - this should not happen.

4008=Error reported from backend.

4010=Error opening configuration file.

4011=Configuration file is no regular readable file.

4012=Incorrect syntax in configuration file.

4013=Incorrect value in configuration file.

4014=Error HTTP response from API proxy: %1.

technical error messages: 5001-5500

5001=General technical error.

5002=MAC check.

SOPG specific error messages: 10000-20001

10003= HTTPS request error

10004= General technical error

10005= General technical error

10006= PIN validation failed

10007= Unexpected error

10008= Authentication failed

10010= cancelPayment too late

10011= Insufficient Balance

10012= Zero Balance

10013= Card not active

10014= Method not allowed for SOPG User

10015= Currency not valid for SOPG User