GP webpay API WS

Technical specification for developers

Version: 1.0

Global Payments Europe, s.r.o.

Created **13.06.2016**

Last update **29.7.2016**

SERVICE. DRIVEN. COMMERCE

globalpaymentsinc.com

Author	GPE Product
Manager	GPE Application Development
Approved by	
Version	1.0
Confidentiality	Confidential

Document history:

Verze	Datum	Provedl	Komentář
0.1	13.06.2016	GPE Product	Initial document version – revision of the document GP_webpay_WS_standard_v2.1
1.0	17.06.2016	GPE Application Development	Document revision

Table of contents

1. Formula clause

This document including any possible annexes and links is intended solely for the needs of an e-shop service provider (hereinafter referred to as “Customer”).

Information included in this document (hereinafter referred to as "Information") are subject to intellectual property and copyright protection of the Global Payments Europe, s.r.o. (hereinafter referred to as "GPE") and are of a commercially confidential nature in accordance with the provisions of the section 504 of the Act No. 89/2012 Coll., Civil Code. The Customer is aware of the legal obligations in relation to the handling of Information.

Information or any part thereof may not be provided or in any way made available to third parties without the prior written consent of the GPE. At the same time, Information may not be used by the Customer for purposes other than for the purpose for which it serves. To avoid any doubts, without the prior written consent of the GPE, Information or any part thereof may be provided or in any way made available neither to companies providing payment processing services on the Internet.

The GPE to the extent permitted by applicable law retains all rights to this document and Information contained therein. Any reproduction, use, exposure, or other publication, or dissemination of Information or its part by methods known and as yet undiscovered without the prior written consent of the GPE is strictly prohibited. The GPE is not in any way responsible for any errors or omissions in Information. GPE reserves the right, without giving any reason, to amend or repeal any Information.

2. Introduction

Technical specification for developers “GP webpay API WS” aims at e-commerce developers of merchants (hereinafter referred to as the developer), who perform integration of the e-shop with the GP webpay payment gateway using the API WS.

Integration using the API HTTP is described in the technical specification for developers “GP webpay API HTTP”.

Important notice: it is the acquirer, who enables merchant to use individual payment methods and functionalities. Information regarding ordering the GP webpay payment gateway and contacts to all acquirers are available at www.gpwebpay.cz.

3. Process of communication via Web Services

A request sent to the GP webpay payment gateway interface API WS has to comply necessarily with the following conditions:

- The request is created in compliance with the Web Services standard defined by the W3C organization (for details go to www.w3.org).
- The request is sent to the WS server end points according to the used environment:
  1. Client test environment:

     https://test.3dsecure.gpwebpay.com/pay-ws/PaymentService
  2. Production environment:

     https://3dsecure.gpwebpay.com/pay-ws/PaymentService

Individual request formats are described below. The following table lists a complete list of requests:

Požadavek	Popis
echo	WS interface availability test
getOrderState	GP webpay receives a request to display the current payment status. Payment status list – Annex 3 – List of payment statuses
getOrderDetail	GP webpay receives a request to display payment details. Payment details and individual details, which are not defined in separate fields, will be sent in the “simpleValueHolder” field. Payment status – Annex 3 – List of payment statuses
processAuthorizationReverse	GP webpay receives a request to cancel the authorization.
processDeposit	GP webpay receives a request to capture the payment.
processDepositReverse	GP webpay receives a request to cancel capturing of the payment.
processCredit	GP webpay receives a request for refund of the payment.
processCreditReverse	GP webpay receives a request to cancel the refund.
processBatchClose	GP webpay receives a request to close a batch.
processOrderClose	GP webpay receives a request to close a payment. Further financial operations with the payment will not be possible any more.
processOrderDelete	GP webpay receives a request to delete a payment. No more further operations will be enabled with the payment. The payment will not be included in the list, but it will be displayed after selecting the option to display deleted payments.

processRecurringPayment	GP webpay receives a request for creation of a recurring payment to the registration payment.
createPaymentLink	GP webpay receives a request for the creation PUSH payment. This link represents the URL for capturing the payment. Link is valid until payment, respectively within the validity period defined by the merchant / GP webpay system

Technical description of the WS is given in WSDL files (Annex no. 4) and underlay generating client application.

Important notice: Examples given in this document are only of a demonstrative character, it is not possible to simply change the values and to send these requests to the server. With regard to the used technology (WS), the resulting request is prepared by the WS framework in the background and then it is sent for processing. Similarly, the response is received and transmitted to the application on the client's side. There is no guarantee that responses will have the same structure as those showed in the given examples.

4. List of Web Services (WS)

4.1 echo – WS interface availability test

4.1.1 Description of the request processing

Action	Description
echo	WS interface availability test

4.1.2 Course of the request processing

Result:

GP webpay responds to the echo request.

Error statuses:

- Request cannot be processed – technical problems.

4.1.3 Request format

Request	echo			
Input parameter	**Type**	**Length**	**Mandatory**	**Description**
Method has no input parameters.				

4.1.4 Response format

Response	EchoResponse			
Attribute	**Type**	**Length**	**Mandatory**	**Description**
Method has no output parameters.				

4.1.5 Example of a request sent to the system and a response received by the merchant

Request:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:core="http://gpe.cz/pay/pay-ws/core">
```

```
   <soapenv:Header/>
   <soapenv:Body>
      <core:echo/>
   </soapenv:Body>
</soapenv:Envelope>
```

Response:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
       <soapenv:Body>
              <ns2:echoResponse xmlns:ns2="http://gpe.cz/pay/pay-ws/core"
xmlns="http://gpe.cz/pay/pay-ws/core/type"/>
       </soapenv:Body>
</soapenv:Envelope>
```

4.2 getOrderState – getting status of payment

4.2.1 Description of the request processing

Action	Description
getOrderState	GP webpay receives a request to display the current payment status. Payment status list – Annex 3 – List of payment statuses

4.2.2 Course of the request processing

Processing:

GP webpay checks the validity of the data entered by:

- searching for the merchant requested;
- checking the signature of the request;
- checking the validity of the contents (length, type, and value) of all elements.

Result:

GP webpay returns information about the payment status.

Error statuses:

- Request cannot be processed – merchant not found;
- Request cannot be processed – invalid operation;
- Request cannot be processed – incorrect data signature;
- Request cannot be processed – element XXX does not contain the requested type;
- Request cannot be processed – element XXX does not contain the requested length;
- Request cannot be processed – element XXX does not contain the requested value;
- Request cannot be processed – technical problems.

4.2.3 Request format

Request	OrderStateRequest			
Input parameter	**Type**	**Length**	**Mandatory**	**Description**

messageId	character	16-256	yes	May contain small/upper case letters, numbers, symbols "+" character "/" character "=". This field must be unique in this combination: `messageId+ acquirer+merchantNumber+<name of the ws operation>` If this condition is not met, the error code PRCODE=80 is returned.
acquirer	character	4	yes	Merchant's bank identification – 4 digits, e .g. 0100 for KB, 0300 for ČSOB
merchantNumber	character	1-10	yes	Merchant number assigned by bank
orderNumber	numeric	1-15	yes	Ordinal number of the payment. Every request from a merchant has to contain a unique payment number.
signature	character base64	1024	yes	A check signature for verification of all the above mentioned fields. For a description of the algorithm used to generate the DIGEST field, see Annex 1 – Signing requests

4.2.4 Response format

Response	**OrderStateResponse**			
Attribute	**Type**	**Length**	**Mandatory**	**Description**
messageId	character	16-256	yes	Field from request
orderNumber	numeric	15	yes	Field from request
state	numeric		yes	Payment status – Annex 3 – List of payment statuses
signature	character base64	1024	yes	A check signature for verification of all the above mentioned fields. For a description of the algorithm used to generate the DIGEST field, see Annex 1 – Signing requests

4.2.5 Example of a request sent to the system and a response received by the merchant

Request:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:core="http://gpe.cz/pay/pay-ws/core" xmlns:type="http://gpe.cz/pay/pay-ws/core/type">
   <soapenv:Header/>
   <soapenv:Body>
      <core:getOrderState>
         <core:orderStateRequest>
            <type:messageId>A111111111111111</type:messageId>
            <type:acquirer>0100</type:acquirer>
            <type:merchantNumber>9999999022</type:merchantNumber>
            <type:orderNumber>1</type:orderNumber>
            <type:signature>KGU4751QSU12 ... </type:signature>
         </core:orderStateRequest>
      </core:getOrderState>
   </soapenv:Body>
</soapenv:Envelope>
```

Response:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
   <soapenv:Body>
      <ns2:getOrderStateResponse xmlns:ns2="http://gpe.cz/pay/pay-ws/core"
xmlns="http://gpe.cz/pay/pay-ws/core/type"
xmlns:ns3="http://gpe.cz/gpwebpay/additionalInfo/response">
         <ns2:orderStateResponse>
            <messageId>A111111111111111</messageId>
            <state>100</state>
            <signature>KE6ULVS9Q5 … </signature>
         </ns2:orderStateResponse>
      </ns2:getOrderStateResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

4.3 getOrderDetail – getting payment details

4.3.1 Description of the request processing

Action	Description
getOrderDetail	GP webpay receives a request to get payment details. Payment details and individual details, which are not defined in separate fields, will be sent in the "simpleValueHolder" field. Payment status – Annex 3 – List of payment statuses

4.3.2 Course of the request processing

Necessary precondition:

The payment must exist.

Processing:

GP webpay checks the validity of the data entered by:

- searching for the merchant requested;
- checking the signature of the request;
- checking the validity of the contents (length, type, and value) of all elements.

Result:

GP webpay returns detailed information about the payment.

Error statuses:

- Request cannot be processed – merchant not found;
- Request cannot be processed – invalid operation;
- Request cannot be processed – incorrect data signature;
- Request cannot be processed – element XXX does not contain the requested type;
- Request cannot be processed – element XXX does not contain the requested length;
- Request cannot be processed – element XXX does not contain the requested value;
- Request cannot be processed – technical problems.

4.3.3 Request format

Request	OrderDetailRequest			
Input parameter	**Type**	**Length**	**Mandatory**	**Description**
messageId	character	16-256	yes	May contain small/upper case letters, numbers, symbols "+" character "/" character "=". This field must be unique in this combination: `messageId+ acquirer+merchantNumber+<name of the ws operation>` If this condition is not met, the error code PRCODE=80 is returned.
acquirer	character	4	yes	Merchant's bank identification – 4 digits, e .g. 0100 for KB, 0300 for ČSOB

merchantNumber	character	1-10	yes	Merchant number assigned by bank
orderNumber	numeric	1-15	yes	Ordinal number of the payment. Every request from a merchant has to contain a unique payment number.
signature	character base64	1024	yes	A check signature for verification of all the above mentioned fields. For a description of the algorithm used to generate the DIGEST field, see Annex 1 – Signing requests

4.3.4 Response format

Response	**OrderDetailResponse**			
Attribute	**Type**	**Length**	**Mandatory**	**Description**
messageId	character	16-256	yes	Field from request
orderNumber	numeric	15	yes	Field from request
state	numeric		yes	Payment status number – Annex 3 – List of payment statuses
panMasked	character	19	no	Masked PAN
orderAmount	numeric		no	Amount of payment
approveAmount	numeric		no	Authorized amount
depositAmount	numeric		no	Captured amount
creditAmount	numeric		no	Refunded amount
approveCode	character		no	Authorization code
orderTime	character		no	Payment creation time
approveTime	character		no	Payment authorization time
depositTime	character		no	Payment capture time
additionalInfoResponse	XML		no	Additional information from system or wallet (MasterPass)
simpleValueHolder			no	Composite type for the transmission of information not defined by separate elements
name	character		yes	Item name
value	character		yes	Item value
signature	character base64	1024	yes	A check signature for verification of all the above mentioned fields. For a description of the algorithm used to generate the DIGEST field, see Annex 1 – Signing requests

4.3.5 Example of a request sent to the system and a response received by the merchant

Request:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:core="http://gpe.cz/pay/pay-ws/core" xmlns:type="http://gpe.cz/pay/pay-ws/core/type">
   <soapenv:Header/>
   <soapenv:Body>
      <core:getOrderDetail>
         <core:orderDetailRequest>
            <type:messageId>A111111111111111</type:messageId>
            <type:acquirer>0100</type:acquirer>
            <type:merchantNumber>9999999022</type:merchantNumber>
            <type:orderNumber>1</type:orderNumber>
            <type:signature>KGU4751QSU12 ... </type:signature>
         </core:orderDetailRequest>
      </core:getOrderDetail>
   </soapenv:Body>
</soapenv:Envelope>
```

Response:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
   <soapenv:Body>
      <ns2:getOrderDetailResponse xmlns:ns2="http://gpe.cz/pay/pay-ws/core"
xmlns="http://gpe.cz/pay/pay-ws/core/type"
```

```
xmlns:ns3="http://gpe.cz/gpwebpay/additionalInfo/response">
         <ns2:orderDetailResponse>
            <messageId>A111111111111111</messageId>
            <state>100</state>
            <orderAmount>100</orderAmount>
            <approveAmount>0</approveAmount>
            <depositAmount>0</depositAmount>
            <creditAmount>0</creditAmount>
            <orderTime>2014-12-12 10:36:37</orderTime>
            <signature>LH7qxjeeiP … </signature>
         </ns2:orderDetailResponse>
      </ns2:getOrderDetailResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

4.4 processAuthorizationReverse – cancellation of payment authorization

4.4.1 Description of the request processing

Action	Description
processAuthorizationReverse	GP webpay receives a request to cancel the authorization of payment.

4.4.2 Course of the request processing

Necessary precondition:

The payment has to be in the status: **AUTHORIZED**.

Processing:

GP webpay checks the validity of the data entered by:

- searching for the merchant requested,
- checking the signature of the request,
- checking the validity of the contents (length, type, and value) of all elements.

Result:

GP webpay cancels the authorization of payment and its status is changed to **REVERSED**.

Error statuses:

- Request cannot be processed – merchant not found;
- Request cannot be processed – invalid operation;
- Request cannot be processed – incorrect data signature;
- Request cannot be processed – element XXX does not contain the requested type;
- Request cannot be processed – element XXX does not contain the requested length;
- Request cannot be processed – element XXX does not contain the requested value;
- Request cannot be processed – technical problems.

4.4.3 Request format

Request	AuthorizationReverseRequest			
Input parameter	**Type**	**Length**	**Mandatory**	**Description**
messageId	character	16-256	yes	May contain small/upper case letters, numbers, symbols "+" character "/" character "=". This field must be unique in this combination: `messageId+ acquirer+merchantNumber+<name of the ws operation>` If this condition is not met, the error code PRCODE=80 is returned.
acquirer	character	4	yes	Merchant's bank identification – 4 digits, e .g. 0100 for KB, 0300 for ČSOB
merchantNumber	character	1-10	yes	Merchant number assigned by bank
orderNumber	numeric	1-15	yes	Ordinal number of the payment. Every request from a merchant has to contain a unique payment number.
signature	character base64	1024	yes	A check signature for verification of all the above mentioned fields. For a description of the algorithm used to generate the DIGEST field, see Annex 1 – Signing requests

4.4.4 Response format

Response	AuthorizationReverseResponse			
Attribute	**Type**	**Length**	**Mandatory**	**Description**
messageId	character	16-256	yes	Field from request
signature	character base64	1024	yes	A check signature for verification of all the above mentioned fields. For a description of the algorithm used to generate the DIGEST field, see Annex 1 – Signing requests

4.4.5 Example of a request sent to the system and a response received by the merchant

Request:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:core="http://gpe.cz/pay/pay-ws/core" xmlns:type="http://gpe.cz/pay/pay-ws/core/type">
   <soapenv:Header/>
   <soapenv:Body>
      <core:processAuthorizationReverse>
         <core:authorizationReverseRequest>
            <type:messageId>A111111111111111</type:messageId>
            <type:acquirer>0100</type:acquirer>
            <type:merchantNumber>9999999022</type:merchantNumber>
            <type:orderNumber>1</type:orderNumber>
            <type:signature>KGU4751QSU12 ... </type:signature>
         </core:authorizationReverseRequest>
      </core:processAuthorizationReverse>
   </soapenv:Body>
</soapenv:Envelope>
```

Response:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
   <soapenv:Body>
      <ns2:processAuthorizationReverseResponse xmlns:ns2="http://gpe.cz/pay/pay-ws/core"
xmlns="http://gpe.cz/pay/pay-ws/core/type"
xmlns:ns3="http://gpe.cz/gpwebpay/additionalInfo/response">
         <ns2:authorizationReverseResponse>
            <messageId>A111111111111111</messageId>
            <signature>SWW4mD6AJE … </signature>
         </ns2:authorizationReverseResponse>
      </ns2:processAuthorizationReverseResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

4.5 processDeposit – capturing payment (money withholding from the cardholder's account)

4.5.1 Description of the request processing

Action	Description
processDeposit	GP webpay receives a request to capture the payment.

4.5.2 Course of the request processing

Necessary preconditions:

The payment has to be in the status: **AUTHORIZED**

Processing:

GP webpay checks the validity of the data entered by:

- searching for the merchant requested;
- checking the signature of the request;
- checking the validity of the contents (length, type, and value) of all elements.

Result:

GP webpay performs capturing of the requested amount of payment. The amount captured must not exceed the authorized amount. The payment will be in the status: **CAPTURED**.

Error statuses:

- Request cannot be processed – merchant not found;
- Request cannot be processed – invalid operation;
- Request cannot be processed – incorrect data signature;
- Request cannot be processed – element XXX does not contain the requested type;
- Request cannot be processed – element XXX does not contain the requested length;
- Request cannot be processed – element XXX does not contain the requested value;
- Request cannot be processed – technical problems.

4.5.3 Request format

Request	`DepositRequest`			
Input parameter	**Type**	**Length**	**Mandatory**	**Description**
messageId	character	16-256	yes	May contain small/upper case letters, numbers, symbols "+" character "/" character "=". This field must be unique in this combination: `messageId+ acquirer+merchantNumber+<name of the ws operation>` If this condition is not met, the error code PRCODE=80 is returned.

acquirer	character	4	yes	Merchant’s bank identification – 4 digitss, e .g. 0100 for KB, 0300 for ČSOB
merchantNumber	character	1-10	yes	Merchant number assigned by bank
orderNumber	numeric	1-15	yes	Number of the payment for which entering into accounts is requested.
amount	numeric		yes	Amount in the smallest units of the currency
signature	character base64	1024	yes	A check signature for verification of all the above mentioned fields. For a description of the algorithm used to generate the DIGEST field, see Annex 1 – Signing requests

4.5.4 Response format

Response	**DepositResponse**			
Attribute	**Type**	**Length**	**Mandatory**	**Description**
messageId	character	16-256	yes	Field from request
signature	character base64	1024	yes	A check signature for verification of all the above mentioned fields. For a description of the algorithm used to generate the DIGEST field, see Annex 1 – Signing requests

4.5.5 Example of a request sent to the system and a response received by the merchant

Request:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:core="http://gpe.cz/pay/pay-ws/core" xmlns:type="http://gpe.cz/pay/pay-ws/core/type">
   <soapenv:Header/>
   <soapenv:Body>
      <core:processDeposit>
         <core:depositRequest>
            <type:messageId>A111111111111111</type:messageId>
            <type:acquirer>0100</type:acquirer>
            <type:merchantNumber>9999999022</type:merchantNumber>
            <type:orderNumber>1</type:orderNumber>
            <type:amount>100</type:amount>
            <type:signature>KGU4751QSU12 ... </type:signature>
         </core:depositRequest>
      </core:processDeposit>
   </soapenv:Body>
</soapenv:Envelope>
```

Response:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
   <soapenv:Body>
      <ns2:processDepositResponse xmlns:ns2="http://gpe.cz/pay/pay-ws/core"
xmlns="http://gpe.cz/pay/pay-ws/core/type"
xmlns:ns3="http://gpe.cz/gpwebpay/additionalInfo/response">
         <ns2:depositResponse>
            <messageId>A111111111111111</messageId>
            <signature>SWW4mD6AJEqb … </signature>
         </ns2:depositResponse>
      </ns2:processDepositResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

4.6 processDepositReverse – cancellation of capturing the payment

4.6.1 Description of the request processing

Action	**Description**
processDepositReverse	GP webpay receives a request to cancel capturing the payment.

4.6.2 Course of the request processing

Necessary precondition:

The payment must be in the status **CAPTURED**.

Processing:

GP webpay checks the validity of the data entered by:

- searching for the merchant requested;
- checking the signature of the request;
- checking the validity of the contents (length, type, and value) of all elements.

Result:

GP webpay cancels capturing the payment. The payment will be in the status **AUTHORIZED.**

Error statuses:

- Request cannot be processed – merchant not found;
- Request cannot be processed – invalid operation;
- Request cannot be processed – incorrect data signature;
- Request cannot be processed – element XXX does not contain the requested type;
- Request cannot be processed – element XXX does not contain the requested length;
- Request cannot be processed – element XXX does not contain the requested value;
- Request cannot be processed – technical problems.

4.6.3 Request format

Request	`DepositReverseRequest`			
Input parameter	**Type**	**Length**	**Mandatory**	**Description**
messageId	character	16-256	yes	May contain small/upper case letters, numbers, symbols "+" character "/" character "=". This field must be unique in this combination: `messageId+ acquirer+merchantNumber+<name of the ws operation>` If this condition is not met, the error code PRCODE=80 is returned.
acquirer	character	4	yes	Merchant's bank identification – 4 digits, e .g. 0100 for KB, 0300 for ČSOB
merchantNumber	character	1-10	yes	Merchant number assigned by bank
orderNumber	numeric	1-15	yes	Number of the payment for which cancellation is requested.
signature	character base64	1024	yes	A check signature for verification of all other above mentioned fields. For a description of the algorithm used to generate the DIGEST field, see Annex 1 – Signing requests

4.6.4 Response format

Response	DepositReverseResponse			
Attribute	**Type**	**Length**	**Mandatory**	**Description**
messageId	character	16-256	yes	Field from request
signature	character base64	1024	yes	A check signature for verification of all the above mentioned fields. For a description of the algorithm used to generate the DIGEST field, see Annex 1 – Signing requests

4.6.5 Example of a request sent to the system and a response received by the merchant

Request:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:core="http://gpe.cz/pay/pay-ws/core" xmlns:type="http://gpe.cz/pay/pay-ws/core/type">
   <soapenv:Header/>
   <soapenv:Body>
      <core:processDepositReverse>
         <core:depositReverseRequest>
            <type:messageId>A111111111111111</type:messageId>
            <type:acquirer>0100</type:acquirer>
            <type:merchantNumber>9999999022</type:merchantNumber>
            <type:orderNumber>1</type:orderNumber>
            <type:signature>KGU4751QSU12 ... </type:signature>
         </core:depositReverseRequest>
      </core:processDepositReverse>
   </soapenv:Body>
</soapenv:Envelope>
```

Response:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
   <soapenv:Body>
      <ns2:processDepositReverseResponse xmlns:ns2="http://gpe.cz/pay/pay-ws/core"
xmlns="http://gpe.cz/pay/pay-ws/core/type"
xmlns:ns3="http://gpe.cz/gpwebpay/additionalInfo/response">
         <ns2:depositReverseResponse>
            <messageId>A111111111111111</messageId>
            <signature>SWW4mD6AJEqb … </signature>
         </ns2:depositReverseResponse>
      </ns2:processDepositReverseResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

4.7 processCredit – creating a refund to the payment

4.7.1 Description of the request processing

Action	Description
processCredit	GP webpay receives a request for refund to the payment.

4.7.2 Course of the request processing

Necessary precondition:

The payment has to be in the status **PROCESSED** or **CREDITED**

Processing:

GP webpay checks the validity of the data entered by:

- searching for the merchant requested,
- checking the signature of the request,
- checking the validity of the contents (length, type, and value) of all elements.

Result:

GP webpay creates a credit in the given amount to the payment.

Error statuses:

- Request cannot be processed – merchant not found;
- Request cannot be processed – invalid operation;
- Request cannot be processed – incorrect data signature;
- Request cannot be processed – element XXX does not contain the requested type;
- Request cannot be processed – element XXX does not contain the requested length;
- Request cannot be processed – element XXX does not contain the requested value;
- Request cannot be processed – technical problems.

4.7.3 Request format

Request	**`CreditRequest`**			
Input parameter	**Type**	**Length**	**Mandatory**	**Description**
messageId	character	16-256	yes	May contain small/upper case letters, numbers, symbols "+" character "/" character "=". This field must be unique in this combination: `messageId+ acquirer+merchantNumber+<name of the ws operation>` If this condition is not met, the error code PRCODE=80 is returned.
acquirer	character	4	yes	Merchant's bank identification – 4 digits, e .g. 0100 for KB, 0300 for ČSOB
merchantNumber	character	1-10	yes	Merchant number assigned by bank
orderNumber	numeric	1-15	yes	Ordinal number of the payment. Every request from a merchant has to contain a unique payment number.
amount	numeric		yes	Amount to be refunded.
signature	character base64	1024	yes	A check signature for verification of all the above mentioned fields. For a description of the algorithm used to generate the DIGEST field, see Annex 1 – Signing requests

4.7.4 Response format

Response	**`CreditResponse`**			
Attribute	**Type**	**Length**	**Mandatory**	**Description**
messageId	character	16-256	yes	Field from request
signature	character base64	1024	yes	A check signature for verification of all the above mentioned fields. For a description of the algorithm used to generate the DIGEST field, see Annex 1 – Signing requests

4.7.5 Example of a request sent to the system and a response received by the merchant

Request:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:core="http://gpe.cz/pay/pay-ws/core" xmlns:type="http://gpe.cz/pay/pay-ws/core/type">
   <soapenv:Header/>
   <soapenv:Body>
      <core:processCredit>
```

```
         <core:creditRequest>
            <type:messageId>A111111111111111</type:messageId>
            <type:acquirer>0100</type:acquirer>
            <type:merchantNumber>9999999022</type:merchantNumber>
            <type:orderNumber>1</type:orderNumber>
            <type:amount>50</type:amount>
            <type:signature>KGU4751QSU12 ... </type:signature>
         </core:creditRequest>
      </core:processCredit>
   </soapenv:Body>
</soapenv:Envelope>
```

Response:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
   <soapenv:Body>
      <ns2:processCreditResponse xmlns:ns2="http://gpe.cz/pay/pay-ws/core"
xmlns="http://gpe.cz/pay/pay-ws/core/type"
xmlns:ns3="http://gpe.cz/gpwebpay/additionalInfo/response">
         <ns2:creditRequestResponse>
            <messageId>A111111111111111</messageId>
            <signature>SWW4mD6AJEqb … </signature>
         </ns2:creditRequestResponse>
      </ns2:processCreditResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

4.8 processCreditReverse – cancellation of unprocessed refund

4.8.1 Description of the request processing

Action	Description
processCreditReverse	GP webpay receives a request to cancel the refund to the payment.

4.8.2 Course of the request processing

Necessary precondition:

The payment must be in the status **CREDITED** and the credit to be cancelled must not be included in an already closed batch.

Processing:

GP webpay checks the validity of the data entered by:

- searching for the merchant requested;
- checking the signature of the request;
- checking the validity of the contents (length, type, and value) of all elements.

Result:

GP webpay cancels the requested credit to the payment.

Error statuses:

- Request cannot be processed – merchant not found;
- Request cannot be processed – invalid operation;
- Request cannot be processed – incorrect data signature;
- Request cannot be processed – element XXX does not contain the requested type

- Request cannot be processed – element XXX does not contain the requested length
- Request cannot be processed – element XXX does not contain the requested value
- Request cannot be processed – technical problems.

4.8.3 Request format

Request	**`CreditReverseRequest`**			
Input parameter	**Type**	**Length**	**Mandatory**	**Description**
messageId	character	16-256	yes	May contain small/upper case letters, numbers, symbols "+" character "/" character "=". This field must be unique in this combination: `messageId+ acquirer+merchantNumber+<name of the ws operation>` If this condition is not met, the error code PRCODE=80 is returned.
acquirer	character	4	yes	Merchant's bank identification – 4 digits, e .g. 0100 for KB, 0300 for ČSOB
merchantNumber	character	1-10	yes	Merchant number assigned by bank
orderNumber	numeric	1-15	yes	Ordinal number of the payment. Every request from a merchant has to contain a unique payment number.
creditNumber	numeric		yes	A unique identifier for the invalidation of refund within the merchant's payments.
signature	character base64	1024	yes	A check signature for verification of all the above mentioned fields. For a description of the algorithm used to generate the DIGEST field, see Annex 1 – Signing requests

4.8.4 Response format

Response	**`CreditReverseResponse`**			
Attribute	**Type**	**Length**	**Mandatory**	**Description**
messageId	character	16-256	yes	Field from request
signature	character base64	1024	yes	A check signature for verification of all the above mentioned fields. For a description of the algorithm used to generate the DIGEST field, see Annex 1 – Signing requests

4.8.5 Example of a request sent to the system and a response received by the merchant

Request:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:core="http://gpe.cz/pay/pay-ws/core" xmlns:type="http://gpe.cz/pay/pay-ws/core/type">
   <soapenv:Header/>
   <soapenv:Body>
      <core:processCreditReverse>
         <core:creditReverseRequest>
            <type:messageId>A111111111111111</type:messageId>
            <type:acquirer>0100</type:acquirer>
            <type:merchantNumber>9999999022</type:merchantNumber>
            <type:orderNumber>1</type:orderNumber>
            <type:creditNumber>1</type:creditNumber>
            <type:signature>KGU4751QSU12 ... </type:signature>
         </core:creditReverseRequest>
      </core:processCreditReverse>
   </soapenv:Body>
</soapenv:Envelope>
```

Response:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
   <soapenv:Body>
      <ns2:processCreditReverseResponse xmlns:ns2="http://gpe.cz/pay/pay-ws/core"
xmlns="http://gpe.cz/pay/pay-ws/core/type"
xmlns:ns3="http://gpe.cz/gpwebpay/additionalInfo/response">
         <ns2:creditReverseResponse>
            <messageId>A111111111111111</messageId>
            <signature>SWW4mD6AJEqb … </signature>
         </ns2:creditReverseResponse>
      </ns2:processCreditReverseResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

4.9 processBatchClose – close batch with payments

4.9.1 Description of the request processing

Action	Description
processBatchClose	GP webpay receives a request to close a batch.

4.9.2 Course of the request processing

Necessary precondition:

The batch must be in the status: **OPEN**

Processing:

GP webpay checks the validity of the data entered by:

- searching for the merchant requested;
- checking the signature of the request,
- checking the validity of the contents (length, type, and value) of all elements.

Result:

GP webpay closes the actual merchant’s batch.

Error statuses:

- Request cannot be processed – merchant not found;
- Request cannot be processed – invalid operation;
- Request cannot be processed – incorrect data signature;
- Request cannot be processed – element XXX does not contain the requested type;
- Request cannot be processed – element XXX does not contain the requested length;
- Request cannot be processed – element XXX does not contain the requested value;
- Request cannot be processed – technical problems.

4.9.3 Request format

Request	`BatchCloseRequest`			
Input parameter	**Type**	**Length**	**Mandatory**	**Description**

messageId	character	16-256	yes	May contain small/upper case letters, numbers, symbols "+" character "/" character "=". This field must be unique in this combination: `messageId+ acquirer+merchantNumber+<name of the ws operation>` If this condition is not met, the error code PRCODE=80 is returned.
acquirer	character	4	yes	Merchant’s bank identification – 4 digits, e .g. 0100 for KB, 0300 for ČSOB
merchantNumber	character	1-10	yes	Merchant number assigned by bank
signature	character base64	1024	yes	A check signature for verification of all the above mentioned fields. For a description of the algorithm used to generate the DIGEST field, see Annex 1 – Signing requests

4.9.4 Response format

Response	`BatchCloseResponse`			
Attribute	**Type**	**Length**	**Mandatory**	**Description**
messageId	character	16-256	yes	Field from request
signature	character base64	1024	yes	A check signature for verification of all the above mentioned fields. For a description of the algorithm used to generate the DIGEST field, see Annex 1 – Signing requests

4.9.5 Example of a request sent to the system and a response received by the merchant

Request:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:core="http://gpe.cz/pay/pay-ws/core" xmlns:type="http://gpe.cz/pay/pay-ws/core/type">
   <soapenv:Header/>
   <soapenv:Body>
      <core:processBatchClose>
         <core:batchClose>
            <type:messageId>A111111111111111</type:messageId>
            <type:acquirer>0100</type:acquirer>
            <type:merchantNumber>9999999022</type:merchantNumber>
            <type:signature>KGU4751QSU12  ... </type:signature>
         </core:batchClose>
      </core:processBatchClose>
   </soapenv:Body>
</soapenv:Envelope>
```

Response:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
   <soapenv:Body>
      <ns2:processBatchCloseResponse xmlns:ns2="http://gpe.cz/pay/pay-ws/core"
xmlns="http://gpe.cz/pay/pay-ws/core/type"
xmlns:ns3="http://gpe.cz/gpwebpay/additionalInfo/response">
         <ns2:batchCloseResponse>
            <messageId>A111111111111111</messageId>
            <signature>SWW4mD6AJEqb … </signature>
         </ns2:batchCloseResponse>
      </ns2:processBatchCloseResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

4.10 processOrderClose – closing a payment

4.10.1 Description of the request processing

Action	Description
processOrderClose	GP webpay receives a request to close a payment. Further financial operations with the payment will not be possible any more.

4.10.2 Course of the request processing

Necessary precondition:

The payment is in the status **PROCESSED** or **CREDITED**

Processing:

GP webpay checks the validity of the data entered by:

- searching for the merchant requested;
- checking the signature of the request,
- checking the validity of the contents (length, type, and value) of all elements.

Result:

GP webpay closes the payment. The payment status will be **CLOSED**

Error statuses:

- Request cannot be processed – merchant not found;
- Request cannot be processed – invalid operation;
- Request cannot be processed – incorrect data signature;
- Request cannot be processed – element XXX does not contain the requested type;
- Request cannot be processed – element XXX does not contain the requested length;
- Request cannot be processed – element XXX does not contain the requested value;
- Request cannot be processed – technical problems.

4.10.3 Request format

Request	`OrderCloseRequest`			
Input parameter	**Type**	**Length**	**Mandatory**	**Description**
messageId	character	16-256	yes	May contain small/upper case letters, numbers, symbols "+" character "/" character "=". This field must be unique in this combination: `messageId+ acquirer+merchantNumber+<name of the ws operation>` If this condition is not met, the error code PRCODE=80 is returned.
acquirer	character	4	yes	Merchant’s bank identification – 4 digits, e .g. 0100 for KB, 0300 for ČSOB
merchantNumber	character	1-10	yes	Merchant number assigned by bank
orderNumber	numeric	1-15	yes	Number of the payment for which closing is requested.
signature	character base64	1024	yes	A check signature for verification of all the above mentioned fields. For a description of the algorithm used to generate the DIGEST field, see Annex 1 – Signing requests

4.10.4 Response format

Response	OrderCloseResponse			
Attribute	**Type**	**Length**	**Mandatory**	**Description**
messageId	character	16-256	yes	Field from request
signature	character base64	1024	yes	A check signature for verification of all the above mentioned fields. For a description of the algorithm used to generate the DIGEST field, see Annex 1 – Signing requests

4.10.5 Example of a request sent to the system and a response received by the merchant

Request:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:core="http://gpe.cz/pay/pay-ws/core" xmlns:type="http://gpe.cz/pay/pay-ws/core/type">
   <soapenv:Header/>
   <soapenv:Body>
      <core:processOrderClose>
         <core:orderCloseRequest>
            <type:messageId>A111111111111111</type:messageId>
            <type:acquirer>0100</type:acquirer>
            <type:merchantNumber>9999999022</type:merchantNumber>
            <type:orderNumber>1</type:orderNumber>
            <type:signature>KGU4751QSU12 ... </type:signature>
         </core:orderCloseRequest>
      </core:processOrderClose>
   </soapenv:Body>
</soapenv:Envelope>
```

Response:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
   <soapenv:Body>
      <ns2:processOrderCloseResponse xmlns:ns2="http://gpe.cz/pay/pay-ws/core"
xmlns="http://gpe.cz/pay/pay-ws/core/type"
xmlns:ns3="http://gpe.cz/gpwebpay/additionalInfo/response">
         <ns2:orderCloseResponse>
            <messageId>A111111111111111</messageId>
            <signature>SWW4mD6AJEqb … </signature>
         </ns2:orderCloseResponse>
      </ns2:processOrderCloseResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

4.11processOrderDelete – final payment deletion

4.11.1 Description of the request processing

Action	Description
processOrderDelete	GP webpay receives a request to delete a payment. Further financial operations with the payment will be possible any more. The payment will not be included in the statement, but it will be displayed after selecting the option to display deleted payments.

4.11.2 Course of the request processing

Processing:

GP webpay checks the validity of the data entered by:

- searching for the merchant requested;
- checking the signature of the request;
- checking the validity of the contents (length, type, and value) of all elements.

Result:

GP webpay deletes the payment. The payment status will be **DELETED**.

Error statuses:

- Request cannot be processed – merchant not found;
- Request cannot be processed – invalid operation;
- Request cannot be processed – incorrect data signature;
- Request cannot be processed – element XXX does not contain the requested type;
- Request cannot be processed – element XXX does not contain the requested length;
- Request cannot be processed – element XXX does not contain the requested value;
- Request cannot be processed – technical problems.

4.11.3 Request format

Request	**OrderDeleteRequest**			
Input parameter	**Type**	**Length**	**Mandatory**	**Description**
messageId	character	16-256	yes	May contain small/upper case letters, numbers, symbols "+" character "/" character "=". This field must be unique in this combination: `messageId+ acquirer+merchantNumber+<name of the ws operation>` If this condition is not met, the error code PRCODE=80 is returned.
acquirer	character	4	yes	Merchant's bank identification – 4 digits, e .g. 0100 for KB, 0300 for ČSOB
merchantNumber	character	1-10	yes	Merchant number assigned by bank
orderNumber	numeric	1-15	yes	Number of the payment for which deletion is requested
signature	character base64	1024	yes	A check signature for verification of all the above mentioned fields. For a description of the algorithm used to generate the DIGEST field, see Annex 1 – Signing requests

4.11.4 Response format

Response	**OrderDeleteResponse**			
Attribute	**Type**	**Length**	**Mandatory**	**Description**
messageId	character	16-256	yes	Field from request
signature	character base64	1024	yes	A check signature for verification of all the above mentioned fields. For a description of the algorithm used to generate the DIGEST field, see Annex 1 – Signing requests

4.11.5 Example of a request sent to the system and a response received by the merchant

Request:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:core="http://gpe.cz/pay/pay-ws/core" xmlns:type="http://gpe.cz/pay/pay-ws/core/type">
   <soapenv:Header/>
   <soapenv:Body>
      <core:processOrderDelete>
         <core:orderDeleteRequest>
            <type:messageId>A111111111111111</type:messageId>
            <type:acquirer>0100</type:acquirer>
            <type:merchantNumber>9999999022</type:merchantNumber>
            <type:orderNumber>1</type:orderNumber>
            <type:signature>KGU4751QSU12 ... </type:signature>
         </core:orderDeleteRequest>
      </core:processOrderDelete>
   </soapenv:Body>
</soapenv:Envelope>
```

Response:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
   <soapenv:Body>
      <ns2:processOrderDeleteResponse xmlns:ns2="http://gpe.cz/pay/pay-ws/core"
xmlns="http://gpe.cz/pay/pay-ws/core/type"
xmlns:ns3="http://gpe.cz/gpwebpay/additionalInfo/response">
         <ns2:orderDeleteResponse>
            <messageId>A1111111111111111</messageId>
            <signature>SWW4mD6AJEqb … </signature>
         </ns2:orderDeleteResponse>
      </ns2:processOrderDeleteResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

4.12 processRecurringPayment – creating a recurring payment to the registration payment

4.12.1 Description of the request processing

Action	Description
processRecurringPayment	GP webpay receives a request for creating a recurring payment to the registration payment

4.12.2 Course of the request processing

Necessary preconditions:

Merchant has to have enabled recurring payments.

In the system, there has to be registered corresponding “master” payment with positive authorization.

Processing:

GP webpay checks the validity of the data entered by:

- searching for the merchant requested,
- checking the signature of the request,
- checking the validity of the contents (length, type, and value) of all elements.

Result:

GP webpay creates and process a new payment based on the registration (master) payment.

Error statuses:

- Request cannot be processed – merchant not found;
- Request cannot be processed – invalid operation;
- Request cannot be processed – incorrect data signature;
- Request cannot be processed – element XXX does not contain the requested type;
- Request cannot be processed – element XXX does not contain the requested length;

- Request cannot be processed – element XXX does not contain the requested value;
- Request cannot be processed – technical problems.

4.12.3 Request format

Request	**RecurringPaymentRequest**			
Input parameter	**Type**	**Length**	**Mandatory**	**Description**
messageId	character	16-256	yes	May contain small/upper case letters, numbers, symbols "+" character "/" character "=". This field must be unique in this combination: `messageId+ acquirer+merchantNumber+<name of the ws operation>` If this condition is not met, the error code PRCODE=80 is returned.
acquirer	character	4	yes	Merchant's bank identification – 4 digits, e .g. 0100 for KB, 0300 for ČSOB
merchantNumber	character	1-10	yes	Merchant number assigned by bank
orderNumber	numeric	1-15	yes	Ordinal number of the payment. Every request from a merchant has to contain a unique payment number.
masterOrderNumber	numeric	1-15	yes	Number of the registration payment
merchantOrderNumber	numeric	1-30	yes	Payment identification for the merchant. *If not entered, value of masterOrderNumber is used.* *Will be printed on the bank statement.* *Each bank has its own approach/limit – see Addendum no. 2 – Maximum length of merchantOrderNumber field*
amount	numeric		no	Amount of the payment. If not entered, values from the registration payment are used.
currencyCode	numeric	3	yes/no	Currency of the transaction. It has to be entered concurrently and only with the "amount" field. Currency identifier in accordance with ISO 4217. Multicurrency is enabled only if bank-assisted. Contact your bank for details.
signature	character base64	1024	yes	A check signature for verification of all the above mentioned fields. For a description of the algorithm used to generate the DIGEST field, see Annex 1 – Signing requests

4.12.4 Response format

Response	**RecurringPaymentResponse**			
Attribute	**Type**	**Length**	**Mandatory**	**Description**
messageId	character	16-256	yes	Field from request
authCode	character	6	yes	Authorization code for payment
signature	character base64	1024	yes	A check signature for verification of all the above mentioned fields. For a description of the algorithm used to generate the DIGEST field, see Annex 1 – Signing requests

4.12.5 Example of a request sent to the system and a response received by the merchant

Request:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:core="http://gpe.cz/pay/pay-ws/core" xmlns:type="http://gpe.cz/pay/pay-ws/core/type">
   <soapenv:Header/>
   <soapenv:Body>
      <core:processRecurringPayment>
         <core:recurringPaymentRequest>
            <type:messageId>A111111111111111</type:messageId>
            <type:acquirer>0100</type:acquirer>
            <type:merchantNumber>9999999022</type:merchantNumber>
            <type:orderNumber>2</type:orderNumber>
            <type:masterOrderNumber>1</type:masterOrderNumber>
            <type:merchantOrderNumber>2</type:merchantOrderNumber>
            <!--Optional:-->
            <type:amount>80</type:amount>
            <!--Optional:-->
            <type:currencyCode>203</type:currencyCode>
            <type:signature>KGU4751QSU12 ... </type:signature>
         </core:recurringPaymentRequest>
      </core:processRecurringPayment>
   </soapenv:Body>
</soapenv:Envelope>
```

Response:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
   <soapenv:Body>
      <ns2:processRecurringPaymentResponse xmlns:ns2="http://gpe.cz/pay/pay-ws/core"
xmlns="http://gpe.cz/pay/pay-ws/core/type"
xmlns:ns3="http://gpe.cz/gpwebpay/additionalInfo/response">
         <ns2:recurringPaymentResponse>
            <messageId>A111111111111111</messageId>
            <authCode>123456</authCode>
            <signature>VQDawVrring … </signature>
         </ns2:recurringPaymentResponse>
      </ns2:processRecurringPaymentResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

4.13 createPaymentLink – creating a payment link for PUSH payments

4.13.1 Description of the request processing

Action	Description
createPaymentLink	GP webpay receives a request to create a PUSH payment. This link represents the URL address for capturing the payment. The link is valid until the payment is captured, respectively within the validity period defined by the merchant / GP webpay system

4.13.2 Course of the request processing

Necessary precondition:

Merchant has to be enabled to create PUSH payments.

Processing:

GP webpay checks the validity of the data entered by:

- searching for the merchant requested,
- checking the signature of the request,
- checking the validity of the contents (length, type, and value) of all elements.

Result:

GP webpay creates PUSH payment in the system and returns URL link.

Error statuses:

- Request cannot be processed – merchant not found;
- Request cannot be processed – invalid operation;
- Request cannot be processed – incorrect data signature;
- Request cannot be processed – element XXX does not contain the requested type;
- Request cannot be processed – element XXX does not contain the requested length;
- Request cannot be processed – element XXX does not contain the requested value;
- Request cannot be processed – technical problems.

4.13.3 Request format

Request	**`PaymentLinkRequest`**			
Input parameter	**Type**	**Length**	**Mandatory**	**Description**
messageId	character	16-256	yes	May contain small/upper case letters, numbers, symbols "+" character "/" character "=". This field must be unique in this combination: `messageId+ acquirer+merchantNumber+<name of the ws operation>` If this condition is not met, the error code PRCODE=80 is returned.
acquirer	character	4	yes	Merchant's bank identification – 4 digits, e .g. 0100 for KB, 0300 for ČSOB
merchantNumber	character	1-10	yes	Merchant number assigned by bank
orderNumber	numeric	1-15	yes	Ordinal number of the payment. Every request from a merchant has to contain a unique payment number.
amount	numeric	15	yes	The amount in the smallest units of the relevant currency For CZK = in hellers, for EUR = in cents
currencyCode	numeric	3	yes	Currency identifier according to ISO 4217 (see Addendum ISO 4217). Multicurrency (using of various currencies) depends on support provided by the respective bank. It is necessary to address your bank in this respect.
depositFlag	numeric	1	yes	Specifies if the payment has to be paid for automatically. Values allowed: 0 = instant payment not required 1 = payment required
merchantOrderNumber	numeric	30	no	Payment identification for the merchant. If not specified, the orderNumber value is used It is printed in the bank statement. *Each bank has its own approach/limit – see Addendum no. 2 – Maximum length of merchantOrderNumber field*
url	character	300	no	The successful result is to be sent to this address.

description	character	255	no	Description of the purchase. The field content is transferred to the 3-D system for a later check by the card holder in the course of the authentication with the issuer's bank Access Control Server. The field may contain only ASCII characters ranging from 0x20 to 0x7E.
merchantData	character	255	no	Any merchant's data returned to the merchant in the response in the unchanged form – only "whitespace" characters are removed from both sides. The field is used to satisfy various demands of the e-shops. The field may only contain ASCII characters ranging from 0x20 to 0x7E. If it is necessary to transmit any other data, BASE64 encoding must be used. The field must not contain any personal data. **The resulting length of the data must not exceed 255 B.**
fastPayId	number	15	no	A unique ORDERNUMBER of the payment, which was used in the past and should serve as a basis to pre-fill card number. The payment should be captured and cannot be older than 12 (18) months, as it may have been automatically removed from the system.
defaultPayMethod	character	255	no	Preferred payment method Supported values: MCM – MasterCard Mobile
disabledPayMethods	character	255	no	Prohibited payment method, although it has allowed merchant. **It has higher priority than the field „PAYMETHOD".** Supported values: MCM – MasterCard Mobile
email	character	6-255	yes	Customer e-mail for sending URL with payment link
merchantEmail	character	6-255	no	Merchant e-mail for successful payment notification
orderExpiry	datum		yes	The maximum validity of payments is limited by setting the system (currently 90 days). It can specify a shorter validity. After the specified date status changes from **PENDING** to **EXPIRED**.
language	character	2	no	Payment gateway default language.
signature	character base64	1024	yes	A check signature for verification of all the above mentioned fields. For a description of the algorithm used to generate the DIGEST field, see Annex 1 – Signing requests

4.13.4 Response format

Response	`PaymentLinkResponse`			
Attribute	**Type**	**Length**	**Mandatory**	**Description**
messageId	character	16-256	yes	Field from request
orderNumber	numeric	15	yes	Field from request
orderLink	character		yes	URL of created PUSH payment

signature	character base64	1024	yes	A check signature for verification of all the above mentioned fields. For a description of the algorithm used to generate the DIGEST field, see Annex 1 – Signing requests

4.13.5 Example of a request sent to the system and a response received by the merchant

Request:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:core="http://gpe.cz/pay/pay-ws/core" xmlns:type="http://gpe.cz/pay/pay-ws/core/type">
   <soapenv:Header/>
   <soapenv:Body>
      <core:createPaymentLink>
         <core:paymentLinkRequest>
            <type:messageId>4234567890123465</type:messageId>
            <type:acquirer>0100</type:acquirer>
            <type:merchantNumber>9999999021</type:merchantNumber>
            <type:orderNumber>149582818701</type:orderNumber>
            <type:amount>100</type:amount>
            <type:currencyCode>978</type:currencyCode>
            <type:depositFlag>1</type:depositFlag>
            <type:merchantOrderNumber>54655554</type:merchantOrderNumber>
            <type:defaultPayMethod>MCM</type:defaultPayMethod>
            <type:email>dholovka@gpe.cz</type:email>
            <type:merchantEmail>dholovka@gpe.cz</type:merchantEmail>
            <type:orderExpiry>2014-06-24</type:orderExpiry>
            <type:signature>KGU4751QSU12 ... </type:signature>
         </core:paymentLinkRequest>
      </core:createPaymentLink>
   </soapenv:Body>
</soapenv:Envelope>
```

Response:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns2:createPaymentLinkResponse xmlns:ns2="http://gpe.cz/pay/pay-ws/core"
xmlns="http://gpe.cz/pay/pay-ws/core/type">
      <ns2:paymentLinkResponse>
        <messageId>4234567890123465</messageId>
        <orderNumber>149582818701</orderNumber>
        <orderLink>https://test.3dsecure.gpwebpay.com:443/pgw/pay/i9iGaEDDwD</orderLink>
        <signature>aSko72YZKN8jPaR+1l ... </signature>
      </ns2:paymentLinkResponse>
    </ns2:createPaymentLinkResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

4.14 Errors occurring at WS requests processing

If an error occurs at processing requests, the XML message (the so-called SOAP fault error) is returned.

4.14.1 Response format – incorrect processing

Response	**PaymentLinkResponse**			
Attribute	**Type**	**Length**	**Mandatory**	**Description**
messageId	character	16-256	yes	Field from request
primaryReturnCode	numeric	4	yes	PRCODE field, see Annex 2 – List of return codes
secondaryReturnCode	numeric	4	yes	SRCODE field, see Annex 2 – List of return codes
signature	character base64	1024	yes	A check signature for verification of all the above mentioned fields. For a description of the algorithm used to generate the DIGEST field, see Annex 1 – Signing requests

4.14.2 Example of a request sent to the system and a response received by the merchant

Request – incorrect input:

```
POST /pay-ws/PaymentService HTTP/1.1
Accept-Encoding: gzip,deflate
Content-Typee: text/xml;charset=UTF-8
SOAPAction: ""
Content-Length: 1140
Host: test.3dsecure.gpwebpay.com
Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.1.1 (java 1.5)

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:core="http://gpe.cz/pay/pay-ws/core" xmlns:type="http://gpe.cz/pay/pay-ws/core/type">
   <soapenv:Header/>
   <soapenv:Body>
      <core:createPaymentLink>
         <core:paymentLinkRequest>
            <type:messageId>4234567890123465</type:messageId>
            <type:acquirer>0100</type:acquirer>
            <type:merchantNumber>9999999021</type:merchantNumber>
            <type:orderNumber>149582818701</type:orderNumber>
            <type:amount>100</type:amount>
            <type:currencyCode>978</type:currencyCode>
            <type:depositFlag>1</type:depositFlag>
            <type:merchantOrderNumber>54655554</type:merchantOrderNumber>
               <type:defaultPayMethod>MCM</type:defaultPayMethod>
            <type:email>dholovka@gpe.cz</type:email>
             <type:merchantEmail>dholovka@gpe.cz</type:merchantEmail>
            <type:orderExpiry>2014-06-24</type:orderExpiry>
            <type:signature>test}</type:signature>
         </core:paymentLinkRequest>
      </core:createPaymentLink>
   </soapenv:Body>
</soapenv:Envelope>
```

Response:

```
HTTP/1.1 500 Internal Server Error
Date: Fri, 08 Aug 2014 22:45:10 GMT
Server: HTTP
Content-Length: 1000
Connection: close
Content-Typee: text/xml; charset=UTF-8
Content-Language: en-US

<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <soapenv:Fault>
      <faultcode>soapenv:Server</faultcode>
      <faultstring>Invalid message format</faultstring>
      <detail>
        <ns2:serviceException xmlns:ns2="http://gpe.cz/pay/pay-ws/core"
xmlns:axis2ns1="http://gpe.cz/pay/pay-ws/core/type">
          <axis2ns1:messageId>1167631408225932022</axis2ns1:messageId>
          <axis2ns1:primaryReturnCode>7</axis2ns1:primaryReturnCode>
          <axis2ns1:secondaryReturnCode>0</axis2ns1:secondaryReturnCode>
          <axis2ns1:signature>H8Di+01df4Ww5j9s ... </axis2ns1:signature>
        </ns2:serviceException>
      </detail>
    </soapenv:Fault>
  </soapenv:Body>
</soapenv:Envelope>
```

5. Annexes and addenda

5.1 Annex no. 1 – Signing messages

5.1.1 Signing a request

GP webpay accepts only requests for which it can be proved that the originator of the request is an authorized subject (i.e. merchant) with whom GPE, s.r.o. has signed a contract for GP webpay services.

The DIGEST field is used to prove the origin of the request. Its contents are generated based on the following data:

- Data sent – this data is used to prove that the contents of the fields have not been changed on the way to the system.
- Private key – the private key is used to prove that the request comes from the merchant.

At the moment of beginning the integration, the merchant using the GP webpay Portal generates a private key, which he/she stores securely and provides it to the developer for integration. The merchant’s public key is stored automatically on the GP webpay server and before the merchant,s request is accepted, it will be used for verifying, if the merchant has signed the request by his/her private key.

DIGEST parameter contained in transmitted requests contains electronic signature of all other fields of the request. The electronic signature guarantees integrity and undeniableness of the transmitted request.

Any request not containing the DIGEST field or with non-matching contents of the DIGEST field will be rejected with the following explanation:

- PRCODE=5 SRCODE=34 “Mandatory field missing, DIGEST” or
- PRCODE =31 “Invalid signature”.

To generate and verify the electronic signature, a string composed as a concatenation of the text interpretation of the values of all fields contained in the request sent, except from the DIGEST field. When compiling the input message, the merchant has to use the same order of fields as that used in the definition of the request and intersperse individual fields by delimiter “|” (pipe, ASCII 124, hexa 7C). The delimiter must not be preceded or followed by whitespace. URLEncode parameters are used only for data transmission, original data have to be used to generate a signature.

Source for generating the DIGEST field in case of method CREATE_ORDER is the value created by concatenation of the fields in the order given here:

MERCHANTNUMBER + | + OPERATION + | + ORDERNUMBER + | + AMOUNT + | + CURRENCY + | + DEPOSITFLAG + | + MERORDERNUM + | + URL + | + DESCRIPTION + | + MD

If the request does not contain any of optional fields, this field is skipped. If the field is sent empty, it is necessary to include it in generating for the DIGEST field and in the string, there will be two separators next to each other – ||.

If the merchant sends only obligatory parameters, for generating the DIGEST field serves the value: MERCHANTNUMBER + | + OPERATION + | + ORDERNUMBER + | + AMOUNT + | + CURRENCY + | + DEPOSITFLAG + | + URL

5.1.2 Response verification

All the responses from the GP webpay system also contain the DIGEST field. Its contents are generated as follows:

- based on the data contained in the response;
- and, at the same time based on the GP webpay private key.

At the moment of beginning the integration, the merchant downloads the GP webpay public key from the GP webpay Portal. It is used by the merchant to verify the contents of the DIGEST field.

This way the merchant can easily verify that:

- the response really comes from the GP webpay;
- the response has not been changed on the way to the merchant.

Furthermore, the response contains also the DIGEST1 parameter, which further enhances the security of the response. The DIGEST1 parameter is generated as the DIGEST parameter, but parameter "MERCHANTNUMBER" is added to the parameters for validation of the DIGEST parameter. This parameter is not sent in the response and the merchant has to add it by himself/herself because he/she knows its value.

The resulting string for validation of the DIGEST1 field looks like this:

<string for the field DIGEST> + | + MERCHANTNUMBER

5.1.3 Generating of the electronic signature

Inputs:

- Data message (message)
- Private RSA key (with a K-length modulus)

Outputs:

- Electronic signature (BASE64 encoded), approximate length K*1.5

The electronic signature is generated as follows

a) the value of the function SHA-1 [3] is derived from the message

b) the hash is encoded into the input value for the RSA signature, using the EMSA-PKCS1-v1_5-ENCODE algorithm as described in paragraph 9.2.1 [1]. The encoding is made as follows:

   01 | FF* | 00 | 30 21 30 09 06 05 2B 0E 03 02 1A 05 00 04 14 | hash

where FF characters are repeated as many times as necessary for the total length of the string to be one octet shorter than the key modulus. The character | is used for the strings concatenation.

c) the RSA signature is calculated using the output value from b), as described in 8.1.1 [1] RSASSA-PKCS1-V1_5-SIGN

d) The output from c) is encoded using BASE64

5.1.4 Verification of the electronic signature

Inputs:

- Data message
- Electronic signature (BASE64 encoded)
- Public RSA key

Outputs:

- Logical value - YES – the signature is valid
- Logical value - NO – the signature is invalid or its verification has not been possible.

The electronic signature is verified as described in 8.1.2 [1] in the following main steps:

a) depending on the settings for the merchant in the GPE system, the correct public key is selected and its integrity is verified;

b) the electronic signature is decoded using BASE64;

c) the output from b) is decrypted using the selected public key;

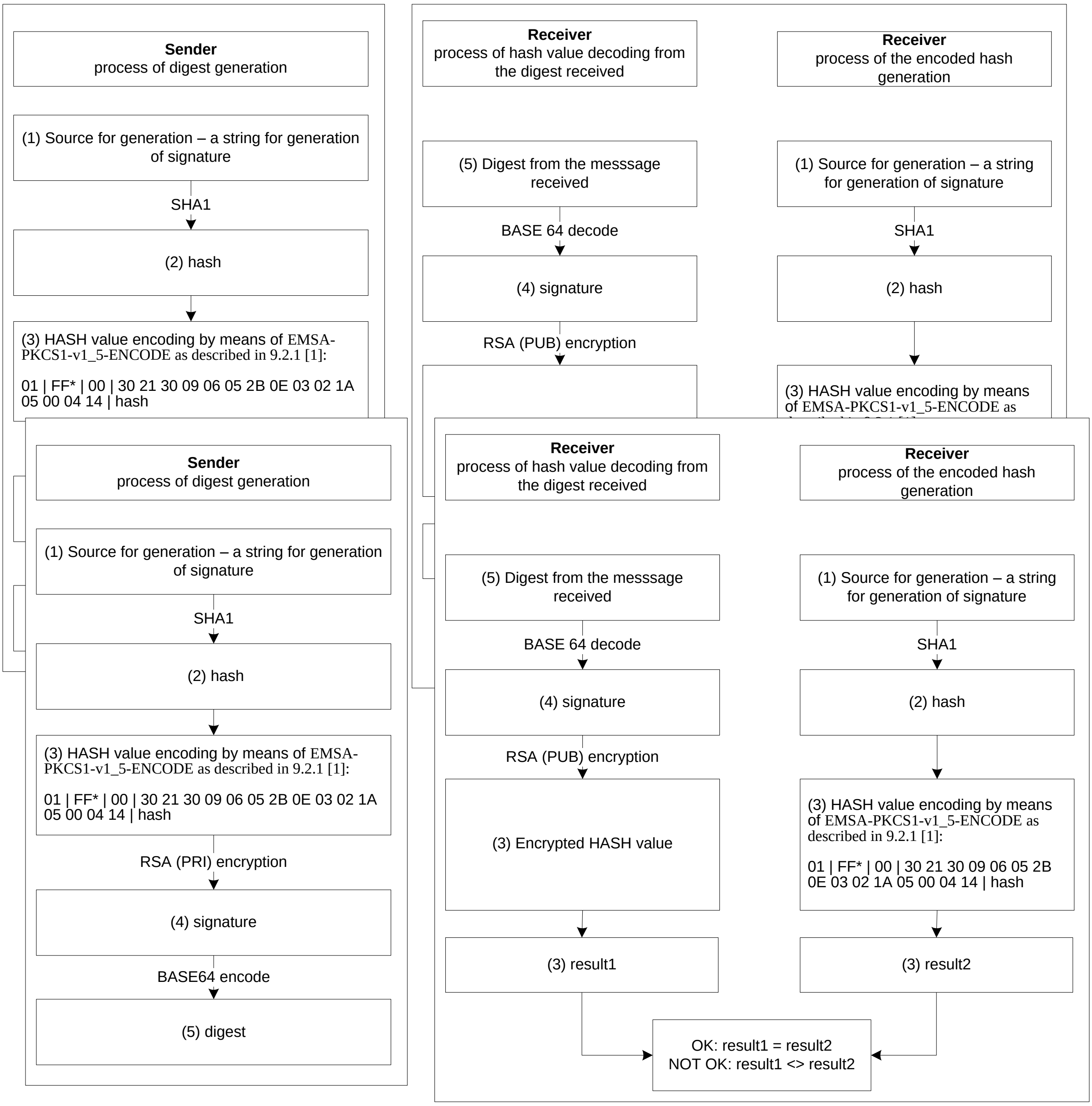
d) a miniature (hash) is generated based on the message and encoded as described in “Generating of the electronic signature”, paras a) and b);

e) the electronic signature decoded according to c) is compared with the result from d). If they are identical, the function returns a logical truth (the signature is valid).

Otherwise, the function returns a logical untruth (the signature is not valid).

The application used for verification of the electronic signature has to identify a signature as invalid also in the case, if verification of the signature has not been possible (for example, due to unavailability of the key).

5.1.5 Graphic representation of key generation and verification


Sender
process of digest generation
(1) Source for generation – a string for generation of signature
SHA1
(2) hash
(3) HASH value encoding by means of EMSA-PKCS1-v1_5-ENCODE as described in 9.2.1 [1]:
01 | FF* | 00 | 30 21 30 09 06 05 2B 0E 03 02 1A 05 00 04 14 | hash
Receiver
process of hash value decoding from the digest received
(5) Digest from the messsage received
BASE 64 decode
(4) signature
RSA (PUB) encryption
Receiver
process of the encoded hash generation
(1) Source for generation – a string for generation of signature
SHA1
(2) hash
(3) HASH value encoding by means of EMSA-PKCS1-v1_5-ENCODE as
Sender
process of digest generation
(1) Source for generation – a string for generation of signature
SHA1
(2) hash
(3) HASH value encoding by means of EMSA-PKCS1-v1_5-ENCODE as described in 9.2.1 [1]:
01 | FF* | 00 | 30 21 30 09 06 05 2B 0E 03 02 1A 05 00 04 14 | hash
RSA (PRI) encryption
(4) signature
BASE64 encode
(5) digest
Receiver
process of hash value decoding from the digest received
(5) Digest from the messsage received
BASE 64 decode
(4) signature
RSA (PUB) encryption
(3) Encrypted HASH value
(3) result1
Receiver
process of the encoded hash generation
(1) Source for generation – a string for generation of signature
SHA1
(2) hash
(3) HASH value encoding by means of EMSA-PKCS1-v1_5-ENCODE as described in 9.2.1 [1]:
01 | FF* | 00 | 30 21 30 09 06 05 2B 0E 03 02 1A 05 00 04 14 | hash
(3) result2
OK: result1 = result2
NOT OK: result1 <> result2

5.1.6 Keys used

To generate the electronic signature (DIGEST), RSA keys (keyPair) are used with a modulus length of 2048 bits. During the communication between GP webpay and the merchant, the following key pairs are used:

GPE's KeyPair	GPE's private key (GPE$_{PRI}$)	Used for the calculation of the electronic signature for messages sent by GPE.	
	GPE's public key (certificate) (GPE$_{PUB}$)	Used by the merchant to verify the electronic signature in messages sent by GPE.	Delivered in the form of a X509 certificate
Merchant's KeyPair	Merchant's private key (MERCH$_{PRI}$)	Used for generating the electronic signature for messages sent by the merchant.	

	Merchant's public key (certificate) (MERCH$_{PUB}$)	Used by GPE to verify the electronic signature in messages sent by the merchant.	Delivered in the form of a X509 self-signed certificate

The application used to generate a self-signed certificate is delivered to the merchant when the merchant applies GPE, s.r.o. for signing a contract. Commercially issued keys can be used as well, but their validity is limited to 1 or 2 years (in comparison with the key generated by the application, there the key validity is longer).

5.1.7 Logging

The application used to verify the electronic signature must store in its audit logs all information about successful and non-successful verification of the electronic signature.

For the purpose of verification of the audit logs, all data required for the verification and re-verification of the electronic signature must be logged. This data includes mainly the electronic signature, the fields, which have been used for its generation, and the result of its verification. If any logs are missing or incomplete, the authenticity of such transactions cannot be confirmed.

5.1.8 References

For further information about the mechanism used to generate the DIGEST field, see the following documents:

[1] RFC 2437, PKCS #1: RSA Cryptography Specifications, October 1998;

[2] XML-Signature Syntax and Processing, W3C Recommendation 12 February 2002,

http://www.w3.org/TR/xmldsig-core/;

[3] RFC 3174 - US Secure Hash Algorithm 1 (SHA1), September 2001;

[4] RFC 2459 – Internet X.509 Public Key Infrastructure Certificate and CRL Profile,

January 1999

The following cryptographic libraries and components may be used to generate the electronic signature:

- JCE Cryptix: Alternative JCE provider offering an algorithm for the RSA/SHA1/PKCS#1 signature, **www.cryptix.org**
- Bouncy Castle: Alternative JCA provider offering libraries for the generation of certificates and work with the PKCS#12 certificate storage, www.bouncycastle.org.
- Crypto++, a free C++ class library of cryptographic schemes supporting also the RSA/SHA1/PKCS#1 algorithm, www.cryptopp.com

5.2 Annex no. 2 – List of return codes

The result of the processing of the request in GP webpay is described as a pair of return codes. If these return codes are different from zero PRCODE describes the type of error. If SRCODE is different from zero it describes the error in detail.

Example:

PRCODE=1 SRCODE=8 means that the DEPOSITFLAG field in the request received has been too long. The RESULTTEXT code returned in this case is "Field too long, DEPOSITFLAG".

5.2.1 PRCODE / primary return code

Value	Meaning in Czech	Meaning in English
0	OK	OK
1	Pole příliš dlouhé	Field too long
2	Pole příliš krátké	Field too short
3	Chybný obsah pole	Incorrect content of field
4	Pole je prázdné	Field is null
5	Chybí povinné pole	Missing required field
11	Neznámý obchodník	Unknown merchant
14	Duplikátní číslo objednávky	Duplicate order number
15	Objekt nenalezen	Object not found
17	Částka k úhradě překročila autorizovanou částku	Amount to deposit exceeds approved amount
18	Součet kreditovaných částek překročil uhrazenou částku	Total sum of credited amounts exceeded deposited amount
20	Objekt není ve stavu odpovídajícím této operaci *Info: Pokud v případě vytváření objednávky (CREATE_ORDER) obdrží obchodník tento návratový kód, vytvoření objednávky již proběhlo a objednávka je v určitém stavu – tento návratový kód je zapříčiněn aktivitou držitele karty (například pokusem o přechod zpět, použití refresh…).*	Object not in valid state for operation
25	Uživatel není oprávněn k provedení operace	Operation not allowed for user
26	Technický problém při spojení s autorizačním centrem	Technical problem in connection to authorization centre
27	Chybný typ objednávky	Incorrect order type
28	*Zamítnuto v 3D* *Info: důvod zamítnutí udává SRCODE*	Declined in 3D
30	*Zamítnuto v autorizačním centru* *Info: Důvod zamítnutí udává SRCODE*	Declined in AC
31	Chybný podpis	Wrong digest
35	Expirovaná session Nastává při vypršení webové session při zadávání karty	Session expired
50	Držitel karty zrušil platbu	The cardholder cancelled the payment
200	Žádost o doplňující informace	Additional info request

1000	Technický problém	Technical problem

5.2.2 SRCODE / secondary return code

Value	Meaning in Czech	Meaning in English
0	Bez významu	No meaning
If PRCODE is 1 to 5, 15 and 20, the following SRCODE may return		
1	ORDERNUMBER	ORDERNUMBER
2	MERCHANTNUMBER	MERCHANTNUMBER
6	AMOUNT	AMOUNT
7	CURRENCY	CURRENCY
8	DEPOSITFLAG	DEPOSITFLAG
10	MERORDERNUM	MERORDERNUM
11	CREDITNUMBER	CREDITNUMBER
12	OPERATION	OPERATION
18	BATCH	BATCH
22	ORDER	ORDER
24	URL	URL
25	MD	MD
26	DESC	DESC
34	DIGEST	DIGEST
If PRCODE is 28, the following SRCODE may return		
3000	**Neověřeno v 3D. Vydavatel karty není zapojen do 3D nebo karta nebyla aktivována.** *Info: Ověření držitele karty bylo neúspěšné (neplatně zadané údaje, stornování autentikace, uzavření okna pro autentikaci držitele karty se zpětnou vazbou…).* *V transakci se nesmí pokračovat.*	**Declined in 3D. Cardholder not authenticated in 3D.** *Note: Cardholder authentication failed (wrong password, transaction cancelled, authentication window was closed…).* *Transaction Declined.*
3001	**Držitel karty ověřen.** *Info: Ověření držitele karty v 3D systémech proběhlo úspěšně. Pokračuje se autorizací objednávky.*	**Authenticated** *Note: Cardholder was successfully authenticated – transaction continue with authorization.*
3002	**Neověřeno v 3D. Vydavatel karty nebo karta není zapojena do 3D.** *Info: V 3D systémech nebylo možné ověřit držitele karty – karta, nebo její vydavatel, není zapojen do 3D.* *V transakci se pokračuje.*	**Not Authenticated in 3D. Issuer or Cardholder not participating in 3D.** *Note: Cardholder wasn't authenticated – Issuer or Cardholder not participating in 3D.* *Transaction can continue.*
3004	**Neověřeno v 3D. Vydavatel karty není zapojen do 3D nebo karta nebyla aktivována.** *Info: V 3D systémech nebylo možné ověřit držitele karty – karta není aktivována, nebo její vydavatel, není zapojen do 3D.* *V transakci je možné pokračovat.*	**Not Authenticated in 3D. Issuer not participating or Cardholder not enrolled.** *Note: Cardholder wasn't authenticated – Cardholder not enrolled or Issuer or not participating in 3D.* *Transaction can continue.*
3005	**Zamítnuto v 3D.Technický problém při ověření držitele karty.** *Info: V 3D systémech nebylo možné ověřit držitele karty – vydavatel karty nepodporuje 3D, nebo technický problém v komunikaci s 3D systémy finančních asociací, či vydavatele karty.* *V transakci není možné pokračovat, povoleno z důvodu*	**Declined in 3D. Technical problem during Cardholder authentication.** *Note: Cardholder authentication unavailable – issuer not supporting 3D or technical problem in communication between associations and Issuer 3D systems.* *Transaction cannot continue.*

	zabezpečení obchodníka před případnou reklamací transakce držitelem karty.	
3006	**Zamítnuto v 3D. Technický problém při ověření držitele karty.** *Info: V 3D systémech nebylo možné ověřit držitele karty – technický problém ověření obchodníka v 3D systémech, anebo v komunikaci s 3D systémy finančních asociací, či vydavatele karty.* *V transakci není možné pokračovat.*	**Declined in 3D. Technical problem during Cardholder authentication.** *Note: Technical problem during cardholder authentication – merchant authentication failed or technical problem in communication between association and acquirer.* *Transaction cannot continue.*
3007	**Zamítnuto v 3D. Technický problém v systému zúčtující banky. Kontaktujte obchodníka.** *Info: V 3D systémech nebylo možné ověřit držitele karty – technický problém v 3D systémech.* *V transakci není možné pokračovat.*	**Declined in 3D. Acquirer technical problem. Contact the merchant.** *Note: Technical problem during cardholder authentication – 3D systems technical problem.* *Transaction cannot continue.*
3008	**Zamítnuto v 3D. Použit nepodporovaný karetní produkt**. *Info: Byla použita karta, která není v 3D systémech podporována.* *V transakci není možné pokračovat.*	**Declined in 3D. Unsupported card product.** *Note: Card not supported in 3D.* *Transaction cannot continue.*
If PRCODE is 30, the following SRCODE may return		
1001	**Zamitnuto v autorizacnim centru, karta blokovana**[1] *Zahrnuje důvody, které naznačují zneužití platební karty – kradená karta, podezření na podvod, ztracená karta apod.* *Většinou pokus o podvodnou transakci.*	**Declined in AC, Card blocked** *Includes the reasons implying that the card has been misused – stolen card, suspected card fraud, lost card, etc.*
1002	**Zamitnuto v autorizacnim centru, autorizace zamítnuta** *Z autorizace se vrátil důvod zamítnutí "Do not honor".* *Vydavatel, nebo finanční asociace zamítla autorizaci BEZ udání důvodu.*	**Declined in AC, Declined** *Reason:* *Card Issuer or financial association rejected authorization (Do Not Honor)*
1003	**Zamitnuto v autorizacnim centru, problem karty** *Zahrnuje důvody:* *expirovaná karta, chybné číslo karty, nastavení karty - pro kartu není povoleno použití na internetu, nepovolená karta, expirovaná karta, neplatná karta, neplatné číslo karty, částka přesahuje maximální limit karty, neplatné CVC/CVV, neplatná délka čísla karty, neplatná expirační doba, pro kartu je požadována kontrola PIN.*	**Declined in AC, Card problem** *Possible reasons:* *Expired card, wrong card number, Internet transaction not permitted to Cardholder, invalid card, invalid card number, amount over card maximum limit, wrong CVC/CVV, invalid card number length, invalid expiry date, PIN control is required for used card*
1004	**Zamitnuto v autorizacnim centru, technicky problem** *Autorizaci není možné provést z technických důvodů – technické problémy v systému vydavatele karty, nebo finančních asociací a finančních procesorů.*	**Declined in AC, Technical problem in authorization process** *Authorization rejected – technical problem* *Technical problem in card Issuer systems or financial associations systems (Card Issuer unavailable)*
1005	**Zamitnuto v autorizacnim centru, Problem uctu** *Důvody: nedostatek prostředků na účtu, překročeny limity, překročen max. povolený počet použití…*	**Declined in AC, Account problem** *Possible reasons: finance absence, over account limit, over daily limit*

If authorization is rejected, the payment gateway receives the return code directly from the card issuer (or from the service provider, or financial association). If the rejected authorization is claimed, the cardholder has to contact his card issuing bank, which responses him directly, or this bank resolves a claim with the bank, which processed the transaction (merchant's bank).

[1] Only the bold part in this and the following cells of this column will be included in the RESULTTEXT field (optional field) in a response sent to the merchant. Other text is only the explanation for merchants.

5.3 Annex no. 3 – List of payment statuses

State value	Status	Description
1	REQUESTED	The payment has been successfully received by GP webpay – the system is waiting for the filling in form (providing sensitive data) by the card holder. If the card holder filled in sensitive data, the request is sent to the 3D system, if the authentication of the card holder is required. Waiting for the result of the 3D system. If the card holder cuts off the card data entering, it is the final state of the payment.
2	PENDING	
3	CREATED	
4	AUTHORIZED	Result of the card holder's authentication enables continuation. Request for authorization was sent to the authorization centre. Result of the payment authorization is successful.
5	APPROVE_REVERSED	Payment authorization has been invalidated. Authorized financial resources have been unblocked on the side of the card holder.
6	UNAPPROVED	Payment authorization has been unsuccessful, the payment cannot be paid. It is not possible to continue.
7	DEPOSITED_BATCH_OPENED	The payment has been marked to be paid in the course of the following batch processing. It is possible to invalidate capturing of the payment until the batch – in which the payment is included - is closed.
8	DEPOSITED_BATCH_CLOSED	Automatic process of closing batches and transmission of data to financial systems have been done.
9	ORDER_CLOSED	Payment closed. The only possible operation is deletion.
10	DELETED	Payment deleted.
11	CREDITED_BATCH_OPENED	Payment marked to be returned in the course of the following batch processing. It is possible to invalidate return of the payment until the batch – in which the payment is included - is closed. As the batch is closed, it remains in this state. For an payment it is possible to create more credits.
12	CREDITED_BATCH_CLOSED	
13	DECLINED	Card holder's authentication in 3D system result is unsuccessful. Card holder is not authenticated – it is not possible to continue. Payment cannot be deleted.
20	CANCELLED	Payment is cancelled by the card holder on the payment page.
21	AUTO_CANCELLED	Payment was cancelled automatically by the system. The merchant has not deposited the amount within the requested period.
100	PUSH_CREATED	New PUSH payment created; no attempt to pay has been made. After entering the card number, the status changes to any of the conditions defined above.
101	PUSH_EXPIRED	After some time, the validity of the payment expires and the payment cannot be used for payment.
102	PUSH_CANCELLED	The merchant has possibility to cancel – via GUI - the created payment;

		e.g. in incorrectly entered parameters.
103	PUSH_BLOCKED	Payment has been blocked automatically after the third unsuccessful attempt for payment.
110	PUSH_PROCESSED	Payment has been used/authorized/processed already.
200	WAIT_FOR_FINALIZE	Response with request for information completion has been sent to the customer – e.g. change of the amount after getting address from the wallet.
201	ABANDONED	The merchant has not completed payment from wallet within requested period. The payment has been invalidated automatically.
210	AUTO_CANCELLED	Automatically cancelled “authorized” payment after expiration of the 30-day period. Payment can be only deleted.
211	AUTO_CLOSED	Automatically closed “processed” or “credited” payment after the expiry of 6-month period. Payment can be only deleted.
220	REC_CREATED	Master payment is created in the system.
221	REC_VALID	Master payment goes to this state when it is processed in the extract. Only to payments in this state, it is possible to generate subsequent recurring payments. This state will return after deletion of information on the processed payment.
222	REC_CANCEL_MERCHANT	Master payment abrogated by the merchant. Used at automatic generation of payments in the GP webpay system according to a timetable defined by the merchant.
223	REC_CANCEL_ISSUER	Cancelled on the basis of token 04 – request by the issuing bank.
224	REC_EXPIRED	If a new payment is not created on the basis of the master payment for more than a year, then the master payment changes its status to EXPIRED.
1000	TECHNICAL_PROBLEM	Unspecified status – technical problem

5.4 Annex no. 4 – Descriptive WSDL

cws-standard.wsdl GPwebpayAdditionalInfoResponse.xsd swaref.xsd

5.5 Addendum no. 1 – Documentation and information sources

- ISO 639-1:2002 Codes for the representation of names of languages

  Part 1: Alpha-2 code

- ISO 639-2:1998 Codes for the representation of names of languages

  Part 2: Alpha-3 code

- ISO 4217:2001 Codes for the representation of currencies and funds
- RFC 3066 – Tags for the Identification of Languages

5.6 Addendum no. 2 – Maximum length of merchantOrderNumber field

Maximum length of merchantOrderNumber for particular banks as displayed in reports devoted for merchants:

Bank	Max. number of digits

	in merchantOrderNumber displayed in the bank’s report
Komerční banka	16
ČSOB CZ	
Raiffeisen bank	10
UniCredit bank	12
ČSOB SK	
ČSAS	