XML
GATEWAY

Contents

Types used in XML gateways

The following documents define the operations, their parameters and types of parameters:

- Request.xsd – description of the format of request to XML gateway
- Response.xsd – description of the format of response to XML gateway

The following information is based on these documents.

Request to XML gateway

Format of request to XML gateway is described by XSD of Request.xsd scheme. The scheme is available at: http://xs2.x-plat.ru/Request.xsd

signType

is enumeration which indicates the type of signature which has been used by a client for authentication.
Possible types of signType enumeration:

Type	Description
capi	Authentication applies a digital signature on the basis of Microsoft Crypto API 2.
md5	Authentication is performed after the check of MD5 fingerprint and a secret phrase.

authInfo

contains information on user authentication and authorization:

Parameter	Type	Description
authInfo.point	int	Pay point.
authInfo.login	string	Operator's login indicated on the pay point.
authInfo.password	string	SHA1 fingerprint of the operator's password at the indicated pay point.
authInfo.signature.type	signType	Signature type used by client XML.
authInfo.signature	string	Signature contents in Base64 Encoding.
authInfo.disposablecode	int	Confirmation code. The field is optional, sent if required.

Example:

```
<header>
    <point>3392</point>
    <login>login</login>
    <password>fEqNCco3Yq9h5ZUglD3CZJT4lBs=</password>
    <signature type="capi">EhA…Ul4dn03BpNmfL0=</signature>
</header>
```

field

contains information on a payment field of a payment:

Paramater	Type	Description
field	string	Payment field value
field.name	string	Payment field name

Example:

```
<field name="phone">9035174909</field>
```

paymentInfo

contains information on a payment sent for processing to XML gateway.

Parameter	Type	Description
paymentInfo.id	long	Unique XML client's payment identifier.
paymentInfo.provider	char(4)	Provider's identifier (up to 4 symbols)
paymentInfo.amount	decimal	Amount for further credit
paymentInfo.user_amount	decimal	Amount charged from client (amount for further credit + commission). The field is optional.
paymentInfo.field	field	The list of payment fields of a payment. Each payment field includes name and value of a payment field.

Example of a payment with one payment field:

```
<payment id="6437282" provider="bee" amount="1.00">
    <field name="phone">9035174909</field>
</payment>
```

Example of a payment with two payment fields and an amount charged from client:

```
<payment id="6437283" provider="test" amount="15.00"
         user_amount="20.00">
    <field name="firstfield">983513424722</field>
    <field name="secondfield">12</field>
</payment>
```

registeredPaymentInfo

contains information of a payment already registered in X-plat system.

Parameter	Type	Description
registeredPaymentInfo.id	Long	Unique XML client's payment identifier.

Example:

```
<payment id="6437282" />
```

checkCommand

describes the command to check the possibility to perform the specified payment.

Parameter	Type	Description
checkCommand.timeout	int	Maximum timeout of payment processing termination by the executing X-plat server. It is not specified if it is necessary to get an immediate response.
checkCommand.payment	paymentInfo	Description of the payment, the possibility of which is to be checked.

Example of asynchronous request for check:

```
<check>
    <payment id="6437282" provider="bee" amount="1.00">
        <field name="phone">9035174909</field>
    </payment>
</check>
```

Example of non-asynchronous request for check:

```
<check timeout="100">
    <payment id="6437282" provider="bee" amount="1.00">
        <field name="phone">9035174909</field>
    </payment>
</check>
```

payCommand

describes a command for payment whose possibility was verified by checkCommand.

Parameter	Type	Description
payCommand.timeout	Int	Maximum timeout of payment processing termination by the executing X-plat server.
payCommand.payment	registeredPaymentInfo	Description of a payment to be executed.

Example of asynchronous request:

```
<pay>
    <payment id="6437282" />
</pay>
```

Example of non-asynchronous request:

```
<pay timeout="100">
    <payment id="6437282" />
</pay>
```

statusCommand

describes the command for request of status of a registered payment.

Parameter	Type	Description
statusCommand.payment	registeredPaymentInfo	Description of the registered payment whose status shall be requested.

Example:

```
<status>
    <payment id="6437282" />
</status>
```

balanceCommand

describes the command to request dealer's balance status.

Example:

```
<balance />
```

provlistCommand

describes the command to request the list of available providers.

Example:

```
<provlist />
```

request

desribes requests to XML gateway.

Parameter	Type	Description
request.guid	Guid	Request GUID
request.header	authInfo	Information for request authentification.
request.check or request.pay or request.status or request.balance or request.provlist or	checkCommand payCommand statusCommand balanceCommand provlistCommand	One of the available commands for XML gateway.

XML gateway response

Format of XML gateway response is described by XSD scheme Response.xsd. The scheme is available at: http://xs2.x-plat.ru/Response.xsd

requestResultCode

is enumeration indicating the results of request processing by the XML gateway.
Possible attributes of enumeration:

Attribute	Description
Success	Request has been successfully processed by XML gateway.
NotPostRequest	Not a POST request has been made, XML gateway has ignored it.
XmlParseError	Error occurred while receiving XML document from POST content. Error text is contained in the Description field.
XmlSchemaError	Error in XML scheme of the document. Error text is contained in the Description field.
AuthError	Operator authenticaton error. Invalid point – login – password.
UserLock	Operator has been locked.
EdsError	DS error.
Denied	Call of this method is not available for the authorized user.
DisposableCode	In order to have the request processed successfully, it shall be called with indication of a confirmation code.
InternalError	Internal XML gateway error. Address integration@x-plat.ru for explanations.

paymentResultCode

is enumeration indicating the results of payment processing by XML gateway.
Possible attributes of enumeration:

Attribute	Description	Action
Success	Payment has been added to processing.	Proceed to processing of 'state' tag.
ProviderNotExistsOrLock	Provider does not exist or is not available to dealer.	Ask your manager to specify the transmitted provider identifier
AmountMinError	The specified amount of payment is not within the valid range.	Update the description of the service provider (Provlist method).
FieldsError	Invalid payment fields of a payment.	Update the description of the service provider (Provlist method).
RequiredFieldsError	Not all the required payment fields have been indicated.	Update the description of the service provider (Provlist method).
DealerBalanceLimit	Dealer has not enough funds to perform a payment.	Make sure there are enough funds to perform a payment. Otherwise consult your manager.
PaymentNotFound	Payment with an indicated identifier has not been found. Occurs only at the stages 'pay' and 'status' (does not occur at the 'check' stage). .	Make sure that 'check' command has been performed successfully.
PaymentNotCheck	Payment can not be completed because the check has not been	Makes sure that PsChecked status has been received.

	performed. Occurs only at the 'pay' stage (does not occur at the 'check' and 'status' stages).	
InternalError	Internal XML service error.	Turn to integration@x-plat.ru for clarification.

paymentStateCode

is enumeration indicating the current payment status on X-plat server.
Possible attributes of enumeration:

Attribute	Final	Description
ServerOk	No	Payment is accepted for processing by X-plat server.
PsChecking	No	Possibility of performing a payment is checked.
PsCheckError	Yes	Error occurred while checking payment possibility.
PsWaitDecision	No	Payment checked. Further payment is awaiting permission.
PsChecked	No Yes (in the case of the first phase)	Payment checked. Payment is possible. Final state under two-phase payment.
PsPaying	No	Payment is sent for execution.
PsStatus	No	Request for payment status from an external payment system.
PsPaid	No	Payment is being performed.
PsPayError	Yes	Payment has not been completed.
PsOk	Yes	Payment has been completed.
Canceled	Yes	Payment has been cancelled by a dealer.

paymentStateType

is enumeration indicating the type of payment status.
Possible attributes:

Attribute	Description
NotFinal	The state is not final – payment processing is under way.
FinalFatal	The state is final. There is no reason to repeat the payment as it will lead to the same result. E.g., an error of the following type – "the indicated number does not exist".

paymentResult

describes the results of payment processing by XML gateway.

Parameter	Type	Description
paymentResult	string	Verbal description of the payment processing result. May be left blank.
paymentResult.code	paymentResultCode	Code of the payment processing result.
paymentResult.fatal	Boolean	Shall not be used for response processing. Refer to 'code' attribute value.

Example:

```
<result code="Success" fatal="true" />
```

paymentState

describes payment condition on X-plat server.

Parameter	Type	Description
paymentState	string	Verbal description of a payment state. Contains a detailed error description, if there is such a description.
paymentState.code	paymentStateCode	Current payment state.
paymentState.type	paymentStateType	Type of payment state.
paymentState.date	DateTime	Time of payment state update.

Example without state description:

```
<state code="PsChecked" type="FinalFatal" date="2009-03-23T09:55:57.723" />
```

Example with state description:

```
<state code="PsChecking" type="NotFinal" date="2009-03-23T09:55:57.723">Check on the provider's service is in the process</state>
```

paymentStatus

describes the results of payment processing by XML gateway.

Parameter	Type	Description
paymentStatus.result	paymentResult	Result of payment processing.
paymentStatus.id	Long	XML client's payment identifier.
paymentStatus.pt_id	Int	Payment identifier of X-plat.
paymentStatus.post_date	DateTime	Time of payment registration on X-plat server.
paymentStatus.state	paymentState	Current payment status.
paymentStatus.parameters		Output payment parameters.

Example:

```
<payment id="290361">
    <result code="Success" fatal="true" />
    <pt_id>83401874</pt_id>
    <post_date>2008-09-16T00:27:18.95</post_date>
    <state code="PsChecked" type="FinalFatal" date="2009-03-23T09:55:57.723" />
</payment>
```

Example with parameters:

```
<payment id="290361">
    <result code="Success" fatal="true" />
    <pt_id>83401874</pt_id>
    <post_date>2008-09-16T00:27:18.95</post_date>
    <state code="PsChecked" type="FinalFatal" date="2009-03-23T09:55:57.723" />
    <parameters>
      <parameter name="ProviderPaymentId">167304946-ntUW</parameter>
    </parameters>
</payment>
```

paymentStatusList

is a list of paymentStatus objects.

requestResult

describes the results of request processing by XML gateway.

Parameter	Type	Description
requestResult	String	Verbal description of the result of request processing by XML gateway.
requestResult.code	requestResultCode	Code of the result of request processing by XML gateway.
requestResult.fatal	Boolean	Indicates whether the error is fatal or not. If yes, interaction with XML gateway shall be seized.

response

describes response of XML gateway.

Parameter	Type	Description
response.guid	Guid	GUID of the request to which the response has been sent.
response.result	requestResult	Result of request processing.
response.payment	paymentStatus	Result of the executed command. Applicable if the code of the request processing result is equal to ‘Success’.
response.signature	String	Signature of the response in Base64 Encoding.

Authentication at XML gateway

Authentication is based on the information from header tag. First, the operator with the specified login is checked at the certain pay point, then, if the operator is not found, SHA1 verification of password fingerprint is launched. At this stage XML gateway may return the following error:

- AuthError – authentication error, invalid point – login – password triplet.

After that, one shall choose the information about the dealer who owns the pay point and the operator. In accordance with this information, the following is checked: whether the dealer is blocked or not; whether the user is blocked or not; whether it's allowed to work with XML gateway or not. At this stage, the XML gate can return the following errors:

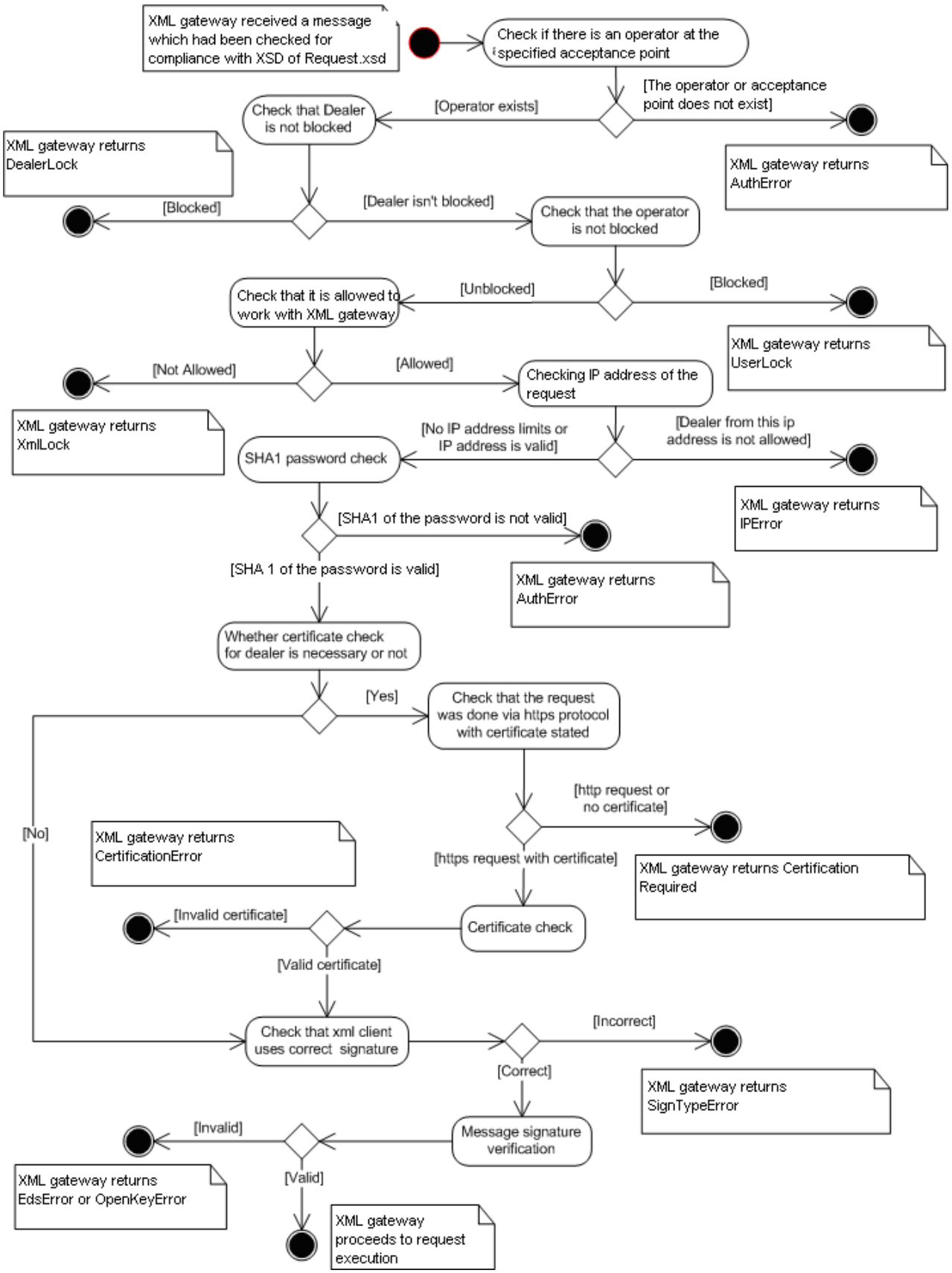
- DealerLock – dealer who owns the specified pay point is blocked
- UserLock – specified operator is blocked.
- XmlLock – specified operator is not allowed to work with the XML gateway.

After that, an open key (a secret key for MD5 signature) and type of the operator's signature are selected. On the basis of this information the following is checked: if the XML client uses the right type of signature or not and if the message is signed correctly or not. At this stage XML gateway may return the following errors:

- SignTypeError – XML client uses incorrect signature type
- OpenKeyError – error occurred while receiving the operator's open key
- EdsError – incorrect signature

When the above mentioned steps are completed successfully, the message is supposed to pass the authentication.

General check scheme is as follows:



Message signature

If CAPI or MD5 signature type is used, a corresponding signature is generated additionally for the message. This signature is included into the message later (tag request.header.signature).
This section of the document covers signature generation. Generally, this process consists of two parts: generating a signature string and signing the string.

Creating signature text

Signature text is created by concatenating the following parameters: “Method title” + “Signature string from parameters” + “Request GUID”.
Note:
All letters in request GUID are changed to lowercase, i.e. a, b, c, d, e and f are used instead of A, B, C, D, E and F.

Method title

Method title depends on the called method and the 'async' parameter (if available).

XML gateway method	Method title
checkCommand	Check
payCommand	Pay
statusCommand	Status
balanceCommand	Balance
provlistCommand	Provlist

Signature string from parameters

For Check command, the signature string is created from paymentInfo parameter. More details may be found in the section “Creating a signature string for PaymentInfo object”.
For Pay and Status commands, signature string from registeredPaymentInfo parameter is formed. To learn more, see the section “Signature string creation for RegisteredPaymentInfo object”.

Signature string formation for PaymentInfo object

Signature string for **PaymentInfo** object is formed by merging all attribute values stated in the string in the following order:
PaymentId + **Provider**+ **Amount** + **UserAmount** (*if transferred*) +
Field1.Name + **Field1.Value** + … + **FieldN.Name** + **FieldN.Value**.
Please note that string representation of the payment amount is a fractional number with 2 digits after the separator. The point is used as a separator. E.g. if the amount is 5.5, the string representation will be "5.50"; similarly, "95.34" for 95.34, and "90.00" for 90.

Example:

```
<payment id=”127823” provider=”mega” amount=”5.50”>
  <field name=”phone”>9225498599</field>
</payment>
```

Signature string: “127823mega5.50phone922549899”.

Signature string creation for RegisteredPaymentInfo object

Signature string for **registeredPaymentInfo** object is formed by merging all attribute values stated in the string in the following order: **PaymentId** + **“0”**.

Example:

```
<payment id=”127823” />
```

Signature string: “1278230”.

Signature with Crypto API

If CAPI is used as the signature method, digital signature obtained from encoding the signature string with MS Crypto API in Base 64 Encoding is placed into signature tag.
To use MS Crypto API, one should use the operator’s private key.

Signature with MD5

When MD5 signature method is used, MD5 fingerprint of the string in WIN-1251 encoding received from concatenation of the operator’s secret phrase is placed into 'signature' tag. It is brought to string through transformation of every byte of the fingerprint into a hexadecimal number represented by two symbols (hex representation).

Operations supported by XML gateway

Xml gateway supports the following operations:

Operation	Function	Parameters	Returned value
checkCommand	Payment completion possibility check. At this stage, the payment is registered in the X-Plat system.	Payment description as a PaymentInfo XML object	Gateway response containing the PaymentStatus in XML format
payCommand	Completion of the registered payment in the X-plat system.	Payment description as a RegisteredPaymentInfo XML object	Gateway response containing the PaymentStatus object in XML format
statusCommand	Get current status on payment registered in the X-plat system.	Payment description as a RegisteredPaymentInfo XML object	Gateway response containing the PaymentStatus object in XML format
balanceCommand	Get dealer balance.	N/A	Gateway response containing the BalanceResult object in XML format
provlistCommand	Get information on available operators.	Whether it's necessary to receive images.	Gateway response containing the ProvlistResult object in XML format

Balance request

To get the dealer's current balance and the amount of available overdraft, BalanceCommand is used. Available funds are calculated as the sum of balance and overdraft.

Request format

Standard request to XML gateway containing balanceCommand.

Example:

```
<?xml version="1.0" encoding="utf-8"?>
<request xmlns="http://xs2.x-plat.ru/Request.xsd"
         guid="c17d8aae-ba95-46eb-911d-0b7d649c9a6b">
  <header>
    <point>3392</point>
    <login>login</login>
    <password>fEqNCco3Yq9h5ZUglD3CZJT4lBs=</password>
    <signature type="capi">EhA…Ul4dn03BpNmfL0=</signature>
  </header>
  <balance />
</request>
```

The signed message string is formed by splicing "Balance" string and the value of 'guid' attribute of the 'request' root tag.
For the example above, the signed string will be as follows:
«Balancec17d8aae-ba95-46eb-911d-0b7d649c9a6b»

Response format

The response from XML gateway returns in a standard form and contains balanceResult object in XML format.

Example of successful response:

```
<?xml version="1.0" encoding="utf-8"?>
<response xmlns="http://xs2.x-plat.ru/Response.xsd"
          guid="c17d8aae-ba95-46eb-911d-0b7d649c9a6b" >
  <result code="Success" fatal="false" />
  <balance over="0" currency_id="643">1749.5</balance>
  <signature>B5kixeZxz5d…TlPXurptQHkqiMA=</signature>
</response>
```

Get list of providers

To get the list of providers available for the dealer, provlistCommand method is used. The response to this command is the list of providers divided into groups.
The command has an optional 'logos' parameter with "normal" attribute value which can be omitted. If it is indicated, the service attempts to find logos for the group and providers of the required type and send them in the response.

Request format

Standard request to XML gateway containing provlistCommand.
Example of request to XML gateway, all parameters are omitted:

```
<?xml version="1.0" encoding="UTF-8"
standalone="no"?> <request xmlns= http://xs2.x-
plat.ru/Request.xsd
guid="99ce944a-5660-45a2-a6c5-9138e5ea64a8">
<header>
    <point>1577</point> <login>test</login>
    <password>4Hu11QbikToDFCa5Xs8q+f3UlIY=</password>
    <signature
  type="md5">23dbecd6b…b893d9081f9130ac</signature> </header>
< provlist logos="normal"/> </request>
```

The signed message string is formed by splicing “Provlist” method name, the value of 'logos' attribute (if it is transferred) and 'guid' attribute of the 'request' root tag.
For the example above, the signed string will be as follows:
«Provlistnormal99ce944a-5660-45a2-a6c5-9138e5ea64a8»

Response format

The response from XML gateway returns in a standard form and contains providersResult object in XML format.

Example of successful response to a request:

```
<?xml version="1.0" encoding="utf-8"?>
<response guid="99ce944a-5660-45a2-a6c5-9138e5ea64c1"
          xmlns="http://xs2.x-plat.ru/Response.xsd">
  <result code="Success" fatal="false"/>
  <provlist>
    <group id="1" title="Mobile communications">
      <logo url="http://services.x-plat.ru/logos/group/1"
hash="3af1a2a3243814c4b154270c9be5feaf" />
    </group>
    <group id="4" title="Other services">
      <logo url="http://services.x-plat.ru/logos/group/4"
hash="4722c93af7b078c1d87af901040afeb1" />
    </group>
    <group id="5" title="Payment systems">
      <logo url="http://services.x-plat.ru/logos/group/5"
hash="02c896a58e4d3235bb758263da73fe94" />
    </group>
    <group id="33" title="Banks">
      <logo  url="http://services.x-plat.ru/logos/group/33"
      hash="cef05bd5c6e57bd86bad64c8d59bd5a2" />
    </group>
```

```
    <provider id="_mts" title="MTS" group="1"
currency="643" min="1.00" max="15000.00">
      <logo url="http://services.x-
plat.ru/logos/logo/_mts?type=normal"
hash="8f9eb8418b97ae57d9e015996c2d3bc5" />
      <number id="phone" title="Phone number" min="10"
max="10" regex="" format="8 (000) 000-0000;0;." />
    </provider>
    <provider id="hkp" title="Repayment of loan provided by
any bank" group="33" currency="643" min="50.00"
max="14999.99">
      <logo url="http://services.x-
plat.ru/logos/logo/hkp?type=normal"
hash="e5250aa92aa2c9cf9bc802e9531971f5" />
      <number id="phone" title="Mobile phone" min="10"
max="10" format="8 (000) 000-0000;0;." />
      <text id="lname" title="Last name" min="2" max="30"
      /> <text id="fname" title="Name" min="2" max="30" />
      <text id="mname" title="Middle name" min="1" max="30"
      /> <number id="bik" title="BIK" min="9" max="9" />
      <number id="account" title="Account number" min="20"
max="20" regex="^\d{20}$" />
      <text id="additional" title="Agreement number\card number
(complementarily)" optional="true" min="0" max="200" />
      <number id="passport" title="Passport number and
      series" min="10" max="255" regex=""/> </provider>
    <provider id="tour" title="iTour" group="4" currency="643"
min="1.00" max="15000.00" schema="itur" tags="default:check">
<logo url=http://services.x-plat.ru/logos/logo/tour?type=normal
hash="854a32dcae45accf2671d5355866c784" />
      <number id="dogovor_id" title="Agreement number"
min="1" max="50" />
      <text id="dogovor_surname" title="Tourist last name"
min="1" max="255" />
    </provider>
  </provlist>
  <signature>AD808BCC42804252232A2F8D375A2B56</signature>
</response>
```

Inside of 'provlist' there may be tags 'group' and 'provider', both of them have the following parameters:

- 'id' is an object identifier unique within the type (a group may have an identifier same to provider's, but two groups/providers can not have the same identifier).
- 'title' is the name of the object to be displayed for the client
- 'group' is an identifier for a group or groups with an object nested. Elements may be missing 'group'. The groups are listed through space.
- 'logo' is a logo description, if it is available for the group/provider. The logo contains the following fields:
  - url – where the logo can be downloaded from
  - hash - hash file with a logo (it helps to check if the logo shall be updated)

Example of nesting of groups and providers:

```
<provlist>
  <group id="1" title="Mobile communications" />
```

```
  <group id="3" title="Internet" />
  <group id="24" title="Dalsvyaz" group="1" />
  <provider id="mts" title="MTS" group="1" ...
  <provider id="bee" title="Beeline" groups="1 3" ...
  <provider id="d001" title="Dalsvyaz Center" group="24" ...
  <provider id="d002" title="Dalsvyaz East" group="24" ...
  <provider id="d003" title="Dalsvyaz West" group="24" ...
</provlist>
```

For the client it shall be displayed in the following way:

```
- g.Mobile communications
-- g.Dalsvyaz
---- p.Dalsvyaz Center
---- p.Dalsvyaz East
---- p.Dalsvyaz West
-- p.MTS
-- p.Beeline
- g.Internet
-- p.Beeline
```

'Provider' object has the following additional attributes:

- currency – provider's currency identifier;
- min – minimum payment amount set by provider;
- max - maximum payment amount set by provider;
- schema – indicates the scheme of payment acceptance for the specified provider. By default (if there is no such attribute in the response) it is considered that provider operates according to the 'default' scheme. Otherwise, one shall specify the name of the scheme applied by the provider. If the client does not support the specified scheme, he shall be recommended to ignore the provider. Several schemes may be listed (separated by space) in the order of application priority. If a client does not support the first scheme, he shall check the second scheme and so on;
- tags – special flags which provider may have. Currently it is implied that two of them are used:
  - default:check - for default-scheme online verification is required;
  - default:online - for default-scheme online payment is required.

Besides, inside of 'provider' tag there are tags describing payment fields of he provider. Currently three types are supported:

- number – a payment field. Digital keyboard shall be displayed to input the field.
- text - a payment field. Full keyboard shall be displayed to input the field;
- list – a payment field. Its value is selected from the specific list.

All three fields have the following attributes:

- id – a payment field identifier. It is used as a 'name' value of the payment field of the payment during transmission;
- title – a name of the payment field to be displayed for the client;
- optional – means that the field is optional. If it is not sent, the field is mandatory and the value of the field is 'false'.

Each type of the payment field has its additional attributes:

- number and text:
  - min – minimum length of payment field;
  - max – maximum length of payment field;;
  - regex – regular expression validating the payment field value. Is not sent if missing;

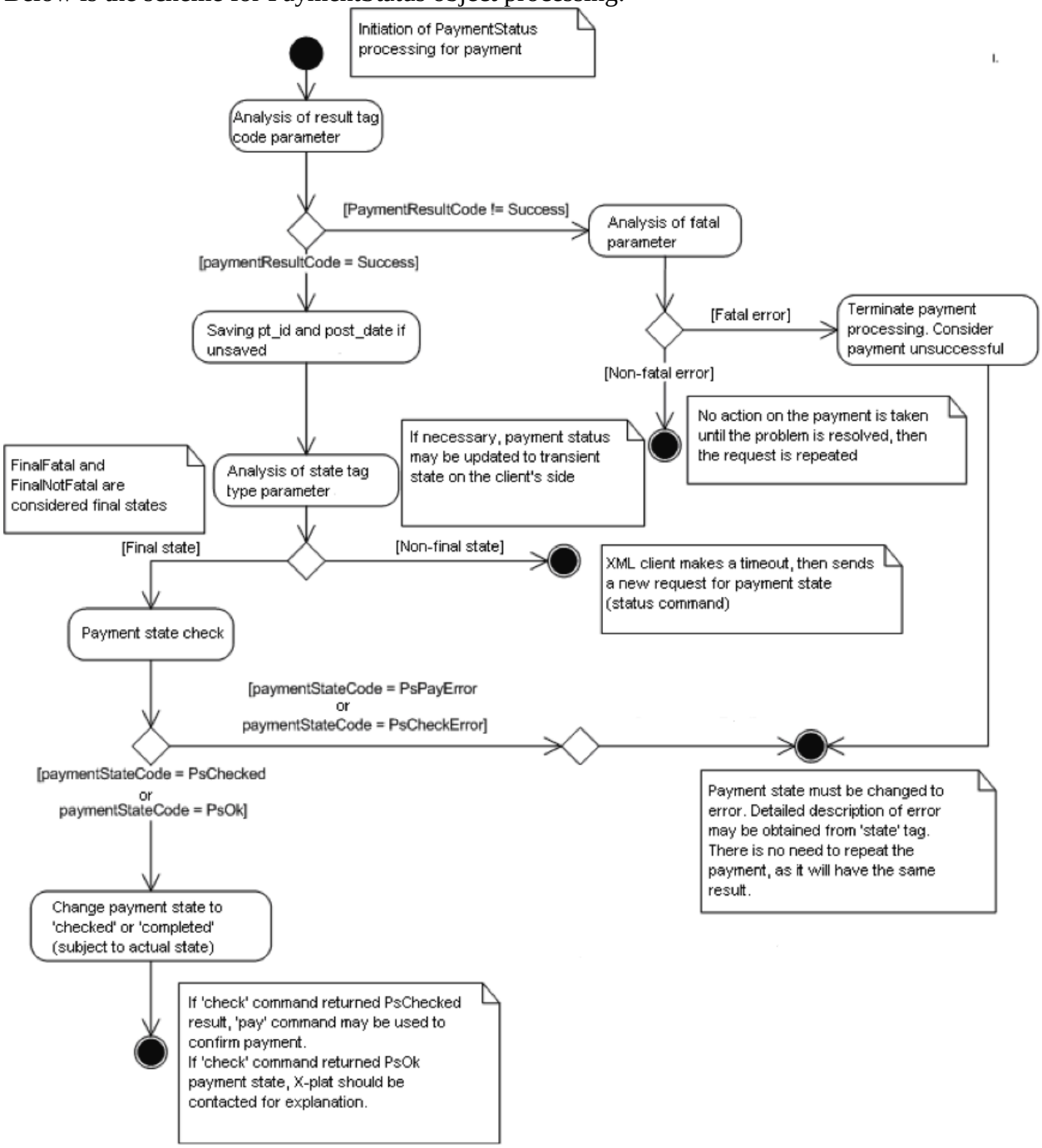
  - format – display format, current format is applied;
- list – inside there is a list of possible values in 'item' tags. Each of the tags contains:
  - key – payment field value, which shall be transmitted to 'value' when the payment is sent;
  - text() – name of the selected payment field to be displayed for the client.

Payment processing

PaymentStatus object processing

All responses to payment requests contain PaymentStatus object, which contains information on the payment's current status on the X-plat server. That is why it is essential that XML client understands how this object operates to comprehend the whole procedure of payment processing.

Below is the scheme for PaymentStatus object processing:

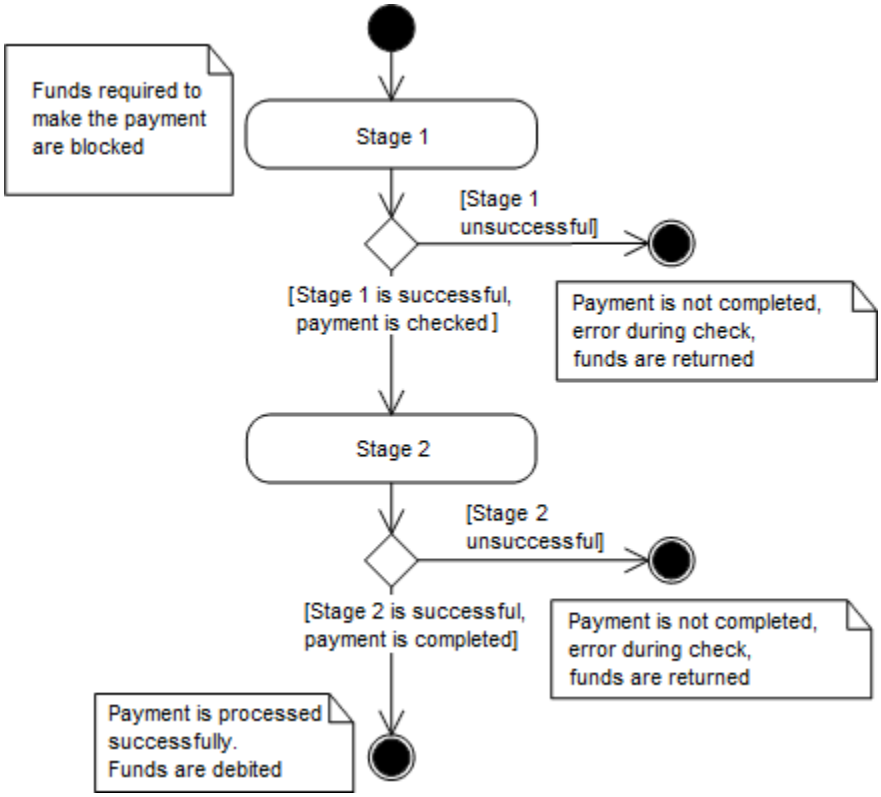


Two-phase payment processing

The two-phase payment processing comprises two stages. At the first stage, the possibility of making the payment is checked. If successful (the payment has correct details and may be processed), the second stage is executed, after which is followed the payment is completed.

At the first stage, the dealer's funds required to complete the payment are blocked on its account. If the first stage is unsuccessful, funds are returned to the dealer. If it is successful, funds are debited. If the second stage fails, funds are also returned to the dealer.

General payment processing scheme



Stage 1. Check the possibility of performing the payment

A request for sending checkCommand request with the description of checked payment is created. After that, the request is sent to XML gateway. If non final payment status is received, it is necessary to repeat the request of the registered payment status in a while and keep repeating it until the payment status is final. Then, the received response is processed by the XML client.

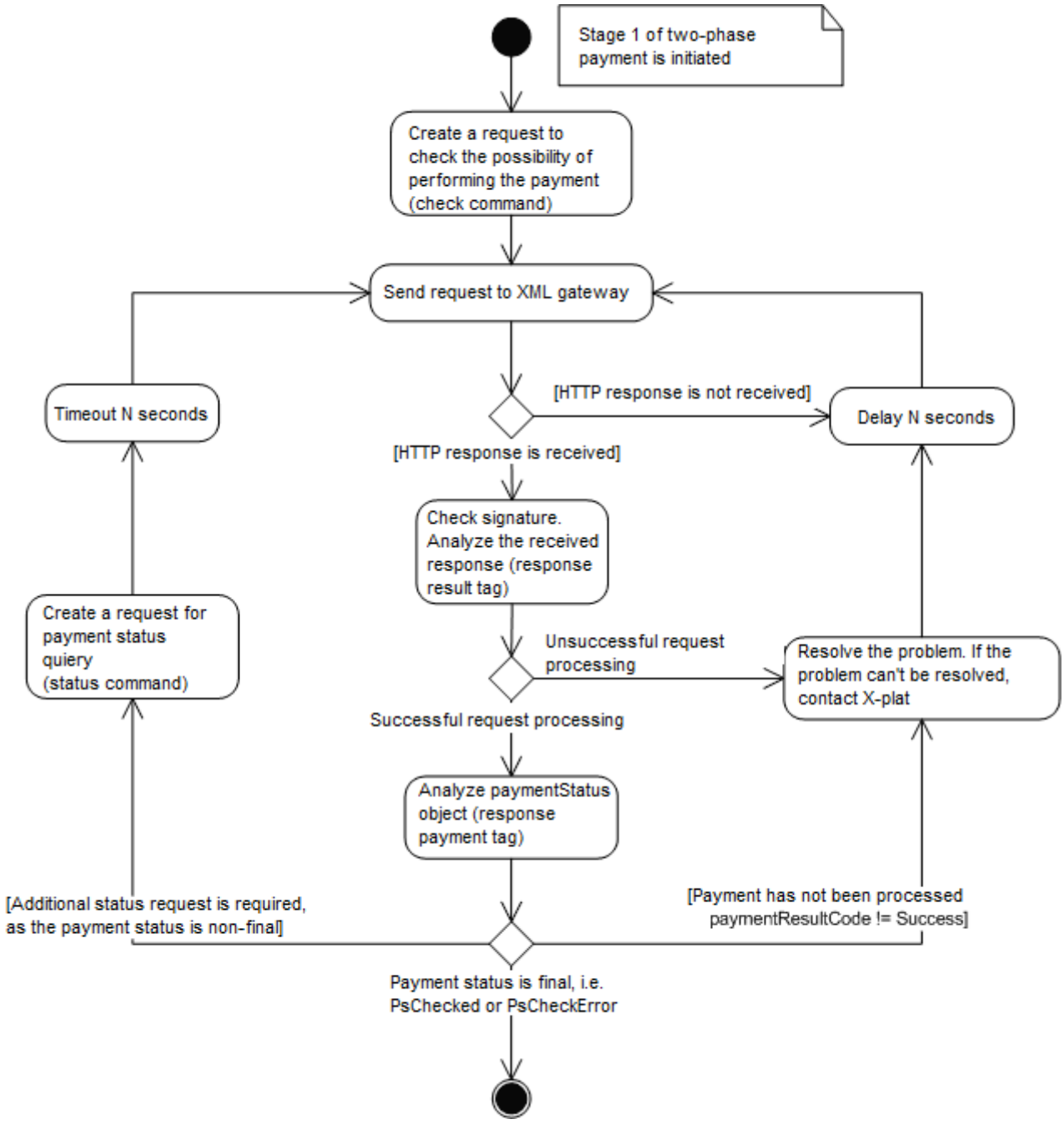
Example of request for stage 1:

```
<?xml version="1.0" encoding="utf-8"?>
<request xmlns=http://xs2.x-plat.ru/Request.xsd
         guid="c17d8aae-ba95-46eb-911d-0b7d649c9a6b" >
  <header>
    <point>3392</point>
    <login>login</login>
    <password>fEqNCco3Yq9h5ZUg1D3CZJT41Bs=</password>
    <signature type="capi">ULmQ9ga…I5OpNmfL0=</signature>
  </header>
  <check timeout="100">
    <payment id="6437282" provider="bee" amount="1.00">
      <field name="phone">9035174909</field>
    </payment>
  </check>
</request>
```

Example of successful response (final state):

```
<?xml version="1.0" encoding="utf-8"?>
<response xmlns="http://xs2.x-plat.ru/Response.xsd"
          guid="59863b6f-3e5-4812-b9f9-edff35e8dd78">
  <result code="Success" fatal="false" />
  <payment id="290361">
    <result code="Success" fatal="true" />
    <pt_id>83401874</pt_id>
    <post_date>2008-09-16T00:27:18.95</post_date>
    <state code="PsChecked" type="FinalFatal" date="2008-09-
16T00:27:19.95" />
  </payment>
  <signature>hFf44hpDigEegcIM…BS5CecB0uhaJQ8M=</signature>
</response>
```

Stage 1 scheme of two-phase payment:



Stage 2. Payment completion

If the first stage is successful (payment status: PsChecked), it is necessary to complete stage 2 to confirm the payment.
To do that, a request is created to send payCommand containing a description of the registered payment. After that, the request is sent to XML gateway. If non final payment status is received, it is necessary to repeat the request of the registered payment status in a while and keep repeating it until the payment status is final. Then, the received response is processed by the XML client.

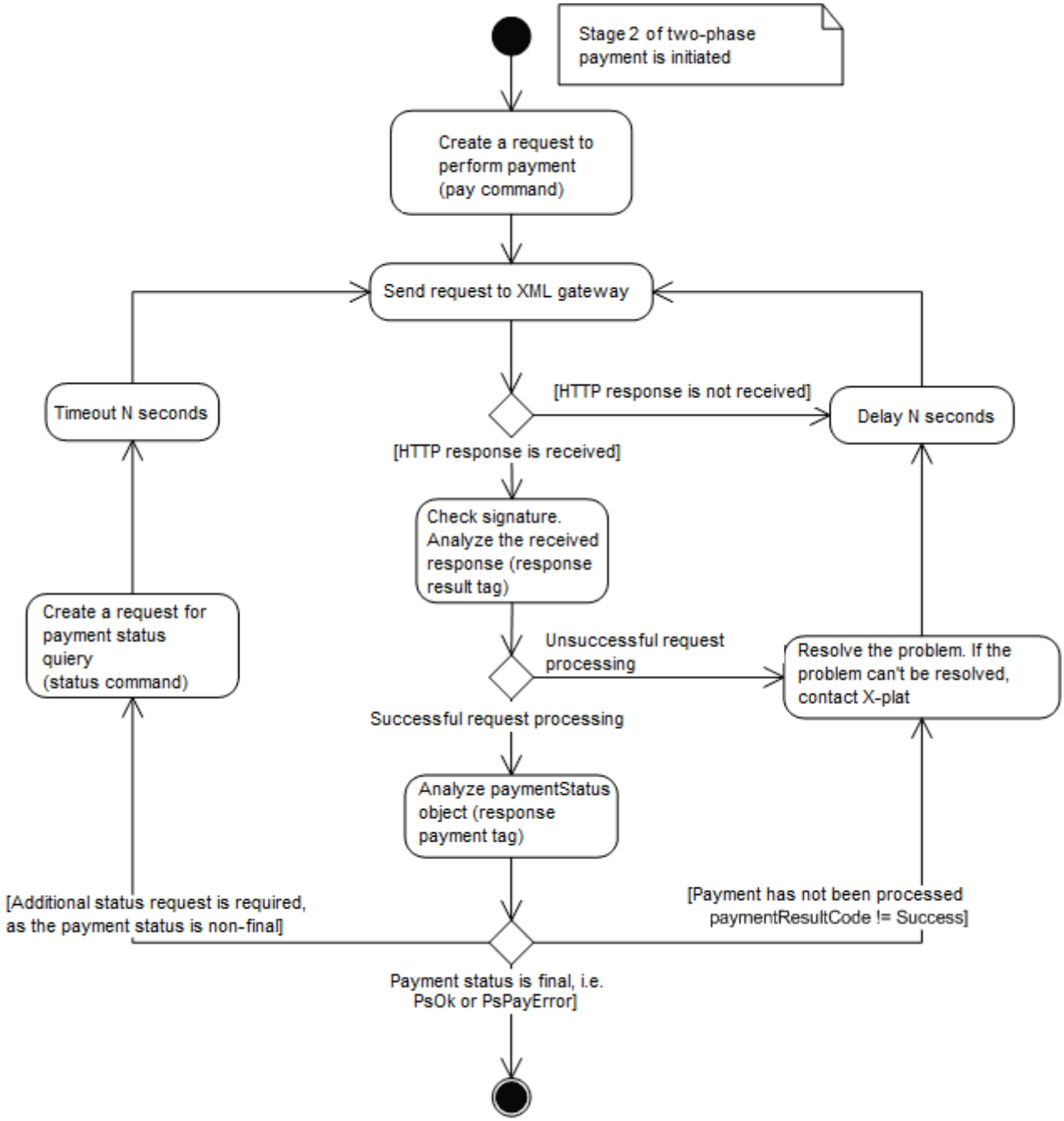
Example of request for stage 2:

```
<?xml version="1.0" encoding="utf-8"?>
<request xmlns=http://xs2.x-plat.ru/Request.xsd
         guid="c17d8aae-ba95-46eb-911d-0b7d649c9a6b" >
  <header>
    <point>3392</point>
    <login>login</login>
    <password>fEqNCco3Yq9h5ZUglD3CZJT4lBs=</password>
    <signature type="capi">ULmQ9ga…I5OpNmfL0=</signature>
  </header>
  <pay timeout="100">
    <payment id="290361" />
  </pay>
</request>
```

Example of successful response (final state):

```
<?xml version="1.0" encoding="utf-8"?>
<response xmlns="http://xs2.x-plat.ru/Response.xsd"
          guid="59863b6f-3e5-4812-b9f9-edff35e8dd78">
  <result code="Success" fatal="false" />
  <payment id="290361">
    <result code="Success" fatal="true" />
    <pt_id>83401874</pt_id>
    <post_date>2008-09-16T00:27:18.95</post_date>
    <state code="PsOk" type="FinalFatal" date="2008-09-
16T00:27:20.95" />
  </payment>
  <signature>hFf44hpDigEegcIM…BS5CecB0uhaJQ8M=</signature>
</response>
```

Stage 2 scheme of two-phase payment:



Processing of response and errors

All commands used for two-phase payment processing (check, pay, status) return one type of response containing PaymentStatus object.
In case of an error related to request processing (response.result tag), it is necessary to resolve the problem. After the problem is resolved, the request may be repeated.
If a payment processing error returns, one should act according to the error type.

Contact details

If you have any questions related to gateway operation, please send them to:
integration@x-plat.ru