

EasySoft-Gateway

2.6

Содержание

1	Введение	2
1.1	Общая структура запроса	2
2	Типы запросов	3
2.1	Проверка абонента.....	3
2.1.1	Дополнительная информация об абоненте	3
2.1.2	Банковские реквизиты	4
2.1.3	Оригинальный сервис услуги	4
2.2	Создание платежа	5
2.3	Подтверждение платежа	5
2.4	Проверка статуса.....	6
2.5	Покупка ваучера	7
2.6	Отмена платежа	7
2.7	Запрос информации партнёра	8
2.8	Получение списка услуг	8
2.9	Реестр платежей	10
3	Цифровая подпись	12
3.1	Общая схема работы с цифровой подписью	12
3.2	Создание и проверка подписи.....	12
3.3	Создание сертификата	12
4	Схемы оплаты.....	14
4.1	Online - оплата	14
4.2	Покупка ваучера	15
5	Дополнение А. Тестовые данные	16

1 Введение

Взаимодействие сторон базируется на основе xml через протокол HTTPS, для аутентификации клиента используется клиентский сертификат, который Вам выдаётся при подключении. Данные передаются через HTTP методом POST. Для неоспоримости платежей, все запросы и ответы подписываются приватным ключом RSA(SHA1). Для нелатинских кодировок используется utf-8.

Рабочий URL: <https://gateway.easysoft.com.ua:8448/op25.aspx>

Возможные статусы HTTP:

- 200 – Успешная обработка запроса;
- 400 – Неверные входные параметры;
- 403 – Неверная подпись или клиентский сертификат;
- 500 – Ошибка обработки запроса, повторите запрос позже или обратитесь в службу поддержки.

1.1 Общая структура запроса

Запрос

```
<Request>
  <DateTime>yyyy-MM-ddTHH:mm:ss</DateTime>
  <Sign>Sign</Sign>
  <Operation>
    <Parameter/>
    ...
    <Parameter/>
  </Operation>
</Request>
```

Ответ

```
<Response>
  <StatusCode>int</StatusCode>
  <StatusDetail>string</StatusDetail>
  <DateTime>yyyy-MM-ddTHH:mm:ss</DateTime>
  <Sign>Sign</Sign>
  <Parameter/>
  ...
  <Parameter/>
</Response>
```

где:

DateTime – дата запроса(ответа);

Sign – цифровая подпись в виде hex-строки;

Operation – тип операции, список всех доступных операций приведен ниже;

StatusCode – статус выполнения операции. 0 – операция выполнена успешно, -1 – ошибка;

StatusDetail – детальное описание статуса операции.

2 Типы запросов

- **Check** (проверка абонента);
- **Payment** (создание платежа);
- **Confirm** (подтверждение платежа);
- **Status** (проверка статуса платежа);
- **Voucher** (покупка ваучера);
- **Cancel** (отмена платежа);
- **PartnerInfo** (запрос информации партнера, такой как баланс, лимит);
- **Services** (список доступных услуг);
- **Clearing** (список проведенных платежей).

2.1 Проверка абонента

Перед созданием платежа нужно проверить, разрешено ли пополнение для данного номера (счета). Если в ответ пришёл статус 0, то пополнение возможно, можно создавать платёж. В противном случае завершаем операцию оплаты.

Запрос

```
<Request>
  <DateTime>yyyy-MM-ddTHH:mm:ss</DateTime>
  <Sign>Sign</Sign>
  <Check>
    <ServiceId>int</ServiceId>
    <Account>string</Account>
  </Check>
</Request>
```

Ответ

```
<Response>
  <StatusCode>int</StatusCode>
  <StatusDetail>string</StatusDetail>
  <DateTime>yyyy-MM-ddTHH:mm:ss</DateTime>
  <Sign>Sign</Sign>
</Response>
```

где:

Account – номер счета;
ServiceId – номер услуги.

В ответе также могут приходиться дополнительные параметры, для различных типов оплат.

2.1.1 Дополнительная информация об абоненте

Иногда (зависит от возможностей поставщика или биллинга услуги) есть возможность получать дополнительную информацию об абоненте.

```
<Response>
  <StatusCode>int</StatusCode>
  <StatusDetail>string</StatusDetail>
  <DateTime>yyyy-MM-ddTHH:mm:ss</DateTime>
```

```

    <Sign>Sign</Sign>
    <AccountInfo>
      <Name>string</Name>
      <Balance>string</Balance>
      ...
    </AccountInfo>
  </Response>

```

Параметры AccountInfo – могут быть разные, в зависимости от услуги(детальное описание доступно в списке услуг).

2.1.2 Банковские реквизиты

Для некоторых финансовых услуг, где при операции Check приходят банковские реквизиты, тег <BankingDetails>, они являются обязательными, их обязательно нужно передавать при создании платежа. Таким образом, если при проверке абонента вернулся тег BankingDetails, то его нужно передать в операции Payment. Эти же реквизиты нужно печатать на чеке.

```

<Response>
  <StatusCode>int</StatusCode>
  <StatusDetail>string</StatusDetail>
  <DateTime>yyyy-MM-ddTHH:mm:ss</DateTime>
  <Sign>Sign</Sign>
  <BankingDetails>
    <Payee>
      <Id>ЕГРПОУ или ИНН получателя</Id>
      <Name>Название или имя получателя</Name>
      <Bank>
        <Name>Название банка получателя</Name>
        <Mfo>МФО получателя</Mfo>
        <Account>Счёт получателя</Account>
      </Bank>
    </Payee>
    <Payer>
      <Id></Id>
      <Name>Название или имя плательщика</Name>
      <Bank>
        <Name>Название банка плательщика</Name>
        <Mfo>МФО плательщика</Mfo>
        <Account>Счёт плательщика</Account>
      </Bank>
    </Payer>
    <Narrative>
      <Name>Назначение платежа</Name>
      <Vat>0</Vat><!--НДС, если не берется то 0-->
    </Narrative>
  </BankingDetails>
</Response>

```

2.1.3 Оригинальный номер услуги

Если в ответе приходит параметр ServiceId, то при создании платежа нужно указывать данный идентификатор услуги. Такие ситуации бывают, когда есть одно название услуги (одна кнопка), но получателями средств, для разных регионов - являются разные субъекты, разные юридические лица, работающие под одним билингом.

```

<Response>

```

```

    <StatusCode>int</StatusCode>
    <StatusDetail>string</StatusDetail>
    <DateTime>yyyy-MM-ddTHH:mm:ss</DateTime>
    <Sign>Sign</Sign>
    <ServiceId>int</ServiceId>
</Response>

```

2.2 Создание платежа

При создании платежа идёт проверка всех параметров, платёж создаётся и ожидает подтверждения. Если в течение 10 мин платёж не подтверждён, он автоматически отклоняется. Если же при создании вернулся статус -1, или произошла временная задержка – нужно создать новый платёж или завершить операцию.

Запрос

```

<Request>
  <DateTime>yyyy-MM-ddTHH:mm:ss</DateTime>
  <Sign>Sign</Sign>
  <Payment>
    <ServiceId>int</ServiceId>
    <PartnerObjectId>string</PartnerObjectId>
    <OrderId>long</OrderId>
    <Account>string</Account>
    <Amount>decimal</Amount>
    <Commission>decimal</Commission>
  </Payment>
</Request>

```

Ответ

```

<Response>
  <StatusCode>int</StatusCode>
  <StatusDetail>string</StatusDetail>
  <DateTime>yyyy-MM-ddTHH:mm:ss</DateTime>
  <Sign>Sign</Sign>
  <PaymentId>long</PaymentId>
</Response>

```

OrderId – уникальный номер заказа(long 8-байт);

PartnerObjectId – номер торговой точки партнера, требование некоторых провайдеров услуг;

Amount – сумма платежа;

Commission – комиссия с клиента(по умолчанию ноль);

Account – номер телефона/счета;

PaymentId – уникальный номер платежа в нашей системе(long 8-байт).

2.3 Подтверждение платежа

Данная операция обеспечивает подтверждение платежа, списание с баланса партнёра и зачисление по лицевому счёту клиента в биллингах поставщиков. Возможны ситуации, когда от провайдера услуг получаем временную задержку или другую непредвиденную ошибку, в этом случае платёж находится в статусе InProcess (в очереди на выполнение). Для получения конечного статуса, нужно послать повторную операцию подтверждения или операцию проверки статуса.

Запрос

```

<Request>
  <DateTime>yyyy-MM-ddTHH:mm:ss</DateTime>
  <Sign>Sign</Sign>
  <Confirm>
    <PaymentId>long</PaymentId>
  </Confirm>
</Request>

```

Ответ

```

<Response>
  <StatusCode>int</StatusCode>
  <StatusDetail>string</StatusDetail>
  <DateTime>yyyy-MM-ddTHH:mm:ss</DateTime>
  <Sign>Sign</Sign>
  <OrderStatus>string</OrderStatus>
  <PaymentDate>yyyy-MM-ddTHH:mm:ss</PaymentDate>
</Response>

```

где:

PaymentDate – дата проведения платежа, если статус не конечный, значение может быть пустым;
OrderStatus – возможные статусы:

- **Inserted** – не конечный платеж создан, но не подтвержден;
- **Accepted** – конечный платеж подтвержден;
- **Declined** – конечный платеж отменен;
- **InProcess** – не конечный находится в очереди на обработку.

2.4 Проверка статуса

Если состояние платежа неизвестно, или когда платёж находится в очереди на выполнение (InProcess) – можно получить статус платежа.

Запрос

```

<Request>
  <DateTime>yyyy-MM-ddTHH:mm:ss</DateTime>
  <Sign>Sign</Sign>
  <Status>
    <OrderId>long</OrderId>
    <PaymentId>long</PaymentId>
  </Status>
</Request>

```

В запросе статуса можно указать один из параметров **OrderId** или **PaymentId**. Предпочтение имеет **PaymentId**.

Ответ

```

<Response>
  <StatusCode>int</StatusCode>
  <StatusDetail>string</StatusDetail>
  <DateTime>yyyy-MM-ddTHH:mm:ss</DateTime>
  <Sign>Sign</Sign>
  <OrderStatus>string</OrderStatus>

```

```

    <PaymentDate>yyyy-MM-ddTHH:mm:ss</PaymentDate>
</Response>

```

2.5 Покупка ваучера

Операция позволяет получать offline – ваучеры, которые заранее загружены в системе EasyPay. Проверить наличие ваучеров можно операцией Check.

Запрос

```

<Request>
  <DateTime>yyyy-MM-ddTHH:mm:ss</DateTime>
  <Sign>Sign</Sign>
  <Voucher>
    <ServiceId>int</ServiceId>
    <PartnerObjectId>string</PartnerObjectId>
    <OrderId>long</OrderId>
    <Amount>decimal</Amount>
  </Voucher>
</Request>

```

Ответ

```

<Response>
  <StatusCode>int</StatusCode>
  <StatusDetail>string</StatusDetail>
  <DateTime>yyyy-MM-ddTHH:mm:ss</DateTime>
  <Sign>Sign</Sign>
  <PaymentId>long</PaymentId>
  <OrderStatus>string</OrderStatus>
  <PaymentDate>yyyy-MM-ddTHH:mm:ss</PaymentDate>
  <VoucherInfo>
    <Serial>string</Serial>
    <Code>string</Code>
    <ExpireDate>string</ExpireDate>
    ...
  </VoucherInfo>
</Response>

```

где:

VoucherInfo – данные ваучера, в зависимости от услуги параметры могут отличаться.

2.6 Отмена платежа

Запрос

```

<Request>
  <DateTime>yyyy-MM-ddTHH:mm:ss</DateTime>
  <Sign>Sign</Sign>
  <Cancel>
    <PaymentId>long</PaymentId>
  </Cancel>
</Request>

```

Ответ

```

<Response>
  <StatusCode>int</StatusCode>
  <StatusDetail>string</StatusDetail>
  <DateTime>yyyy-MM-ddTHH:mm:ss</DateTime>
  <Sign>Sign</Sign>
  <OrderStatus>string</OrderStatus>
  <PaymentDate>yyyy-MM-ddTHH:mm:ss</PaymentDate>
</Response>

```

Операция отмены возможна для услуг, которые позволяют отмену, действует в течение текущего дня. Отмена прошла успешно, если вернулся статус Declined.

2.7 Запрос информации партнёра

Дополнительная информация партнёра.

Запрос

```

<Request>
  <DateTime>yyyy-MM-ddTHH:mm:ss</DateTime>
  <Sign>Sign</Sign>
  <PartnerInfo/>
</Request>

```

Ответ

```

<Response>
  <StatusCode>int</StatusCode>
  <StatusDetail>string</StatusDetail>
  <DateTime>yyyy-MM-ddTHH:mm:ss</DateTime>
  <Sign>Sign</Sign>
  <Balance>decimal</Balance>
  <Limit>decimal</Limit>
  <LimitActive>bool</LimitActive>
</Response>

```

где:

Balance – баланс партнёра;

Limit – лимит партнёра;

LimitActive – статус лимита: включён/отключён.

2.8 Получение списка услуг

Список услуг доступных партнёру с детальным описание полей ввода.

Запрос

```

<Request>
  <DateTime>yyyy-MM-ddTHH:mm:ss</DateTime>
  <Sign>Sign</Sign>
  <Services>
    <Lang>string</Lang>
  </Services>
</Request>

```

где:

Lang – язык описания услуг (доступны: ru-русский, uk-украинский, en-английский), по умолчанию ru.

Ответ

```

<Response>
  <StatusCode>int</StatusCode>
  <StatusDetail>string</StatusDetail>
  <DateTime>yyyy-MM-ddTHH:mm:ss</DateTime>
  <Sign>Sign</Sign>
  <Categories>
    <Category>
      <Name>string</Name>
      <Description>string</Description>
      <Services>
        <Service>
          <ServiceId>int</ServiceId>
          <Amount.Min>decimal</Amount.Min>
          <Amount.Max>decimal</Amount.Max>
          <Name>string</Name>
          <Description>string</Description>
          <AccountInfo>string</AccountInfo>
          <Template>string</Template>
          <Type>string</Type>
          <Icon>string</Icon>
          <Fields>
            <Field>
              <Name>string</Name>
              <Hint>string</Hint>
              <Heading>string</Heading>
              <Mask>string</Mask>
              <Filter>string,string,...</Filter>
              <Format>string</Format>
              <Regex>string</Regex>
              <Index>int</Index>
              <Min>int</Min>
              <Max>int</Max>
            </Field>
          </Fields>
        </Service>
        ...
      </Services>
      <SubCategories>
        <SubCategory>
          <Name>string</Name>
          <Description>string</Description>
          <Services>
            <Service>
              ...
            </Service>
            ...
          </Services>
        </SubCategory>
        ...
      </SubCategories>
    </Category>
  </Categories>
</Response>

```

Ответ возвращается в сжатой форме (Content-Encoding: gzip).

где:

Name – название категории, подкатегории, услуги;

Description – описание;

Serviceld – номер услуги;

Amount.Min – минимальная сумма платежа;

Amount.Max – максимальная сумма платежа;

AccountInfo – описание параметров, которые могут приходиться в операции Check;

Template – шаблон проведения платежа(GenericPayment(общая схема)/GenericCommunal(схема для коммунальных платежей));

Type – тип услуги(Online/Voucher);

Icon – название иконки для сервиса(список иконок доступен на <https://easypay.ua/content/images/logo/{Icon}.png>);

Fields – поля ввода(в большинстве услуг идет одно поле ввода – Account, но бывают услуги где их два и больше. В таком случае поле Account – содержит все поля, разделенные запятой);

Field.Name – название параметра который нужно передавать в операциях Check и Payment.(пример если есть два поля Name и Id, то нужно передавать: <Account>Иванов,123</Account> <Name>Иванов</Name> <Id>123</Id>);

Field.Hint – детальное описание поля ввода;

Field.Heading – надпись сверху поля ввода;

Field.Mask – маска ввода(пример: 2620\$\$\$\$\$\$\$\$\$, где \$ - вводимая переменная. На сервер нужно передавать все значение - 26201234567890. Значения параметров Min и Max – в данном случае действуют только на длину вводимой переменной: Min=10, Max=10);

Field.Filter – фильтр поля ввода(пример: 097,067 или !050-отрицание);

Field.Regex – регулярное выражение;

Field.Format – формат ввода(доступны следующие форматы: Phone(телефон в международном формате 380971112233), Number(числовое поле), Card(платежная карта 16 цифр), Regex(поле должно соответствовать регулярному выражению в параметре Field.Regex), Keyboard(любые символы));

Field.Index – порядковый номер поля для отображения(также номер в поле Account);

Field.Min – минимальное количество символов ввода;

Field.Max – максимальное количество символов ввода.

2.9 Реестр платежей

Данная операция позволит получить список успешных платежей.

Запрос

```
<Request>
  <DateTime>yyyy-MM-ddTHH:mm:ss</DateTime>
  <Sign>Sign</Sign>
  <Clearing>
    <ClearingDate>YYYY-MM-DD</ClearingDate>
  </Clearing>
</Request>
```

Ответ

```
<Response>
  <StatusCode>int</StatusCode>
  <StatusDetail>string</StatusDetail>
  <DateTime>yyyy-MM-ddTHH:mm:ss</DateTime>
  <Sign>Sign</Sign>
  <Payments>
    <Payment orderId="long" paymentId="long" serviceld="int" account="string" amount="decimal"
      paymentDate="yyyy-MM-ddTHH:mm:ss"/>
```

```
</Payments>  
</Response>
```

Ответ возвращается в сжатой форме (Content-Encoding: gzip).

3 Цифровая подпись

3.1 Общая схема работы с цифровой подписью

Для работы с подписью используются два сертификата: сертификат EasySoft – GateSign-EasySoft.cer и сертификат партнера – GateSign-Partner.cer. Сертификат EasySoft – передается вместе с протоколом, а партнера – передается Партнером после регистрации в нашей системе ответственным менеджером. Соответственно партнер подписывает все запросы приватным ключом – GateSign-Partner, а проверяет публичным – GateSign-EasySoft.

Для теста передаются три сертификата:

- Gateway-Test.pfx – для доступа по ssl(пароль test);
- GateSign-EasySoft.cer – сертификат проверки подписи;
- GateSign-Test.pfx – сертификат подписи(пароль test).

3.2 Создание и проверка подписи

Для подписи нужно использовать все тело запроса с пустым тегом `<Sign></Sign>`:

Запрос

```
<Request>
  <DateTime>2010-11-11T12:05:10</DateTime>
  <Sign></Sign>
  <Check>
    <ServiceId>208</ServiceId>
    <Account>0971234567</Account>
  </Check>
</Request>
```

```
buff=Utf8.GetBytes(Request)
Sign = RSA(SHA1(buff)) = 089D7F0F5DE5CACB16F54429EEA9...
```

в результате получаем:

```
<Request>
  <DateTime>2010-11-11T12:05:10</DateTime>
  <Sign>089D7F0F5DE5CACB16F54429EEA9...</Sign>
  <Check>
    <ServiceId>208</ServiceId>
    <Account>0971234567</Account>
  </Check>
</Request>
```

Для проверки подписи используется аналогичная процедура.

3.3 Создание сертификата

Для создания сертификата можно воспользоваться бесплатной утилитой openssl.

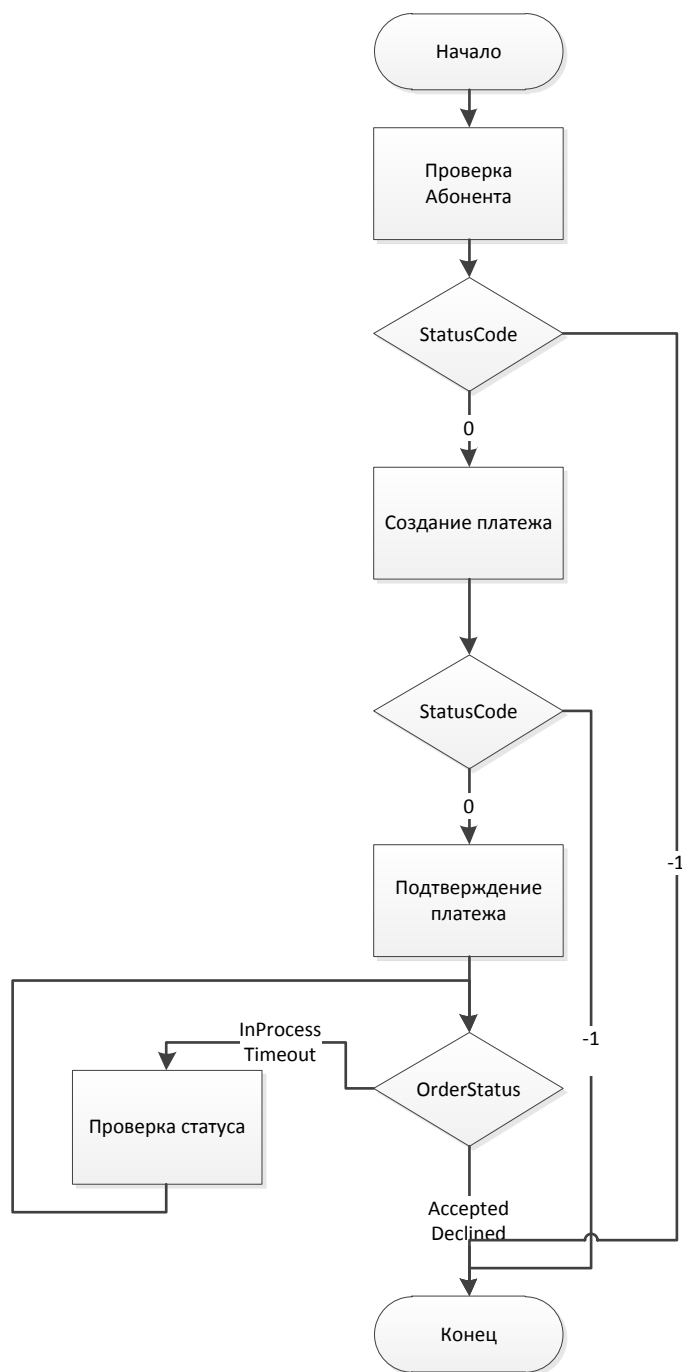
```
openssl genrsa -out partner.ppk 1024  
openssl req -new -key partner.ppk -out partner.req  
openssl x509 -req -days 730 -in partner.req -signkey partner.ppk -out GateSign-Partner.cer
```

В сертификате используем CN=GateSign-ObjectId-PartnerName, где GateSign – константа, ObjectId – номер терминала, который сообщается партнеру после заведения партнера в системе EasyPay, PartnerName – Ваше название латинскими буквами.

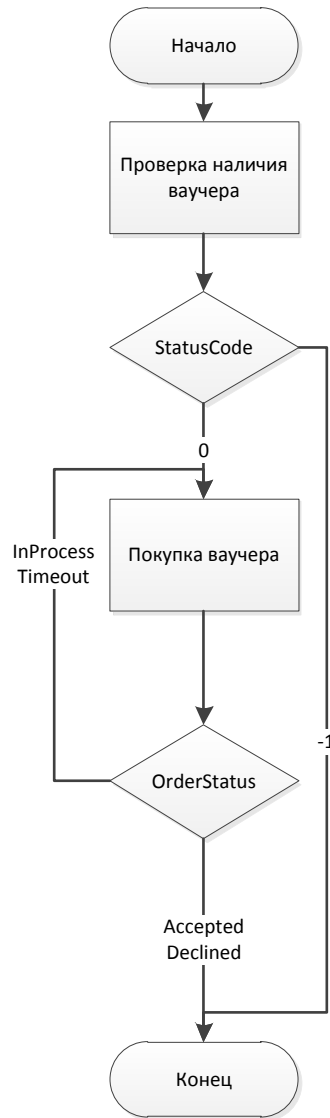
Пример: CN=GateSign-3000-SuperPay

4 Схемы оплаты

4.1 Online - оплата



4.2 Покупка ваучера



5 Дополнение А. Тестовые данные

Тестовый URL: <https://gateway.easysoft.com.ua:8448/op25.aspx>

Тестовые сертификаты: Gateway-Test.pfx, Sign-EasySoft.cer, GateSign-Test.pfx

Операция	Запрос(Account)	Ответ
Check	<code><Account>0100000001</Account></code>	<code><StatusCode>0</StatusCode></code>
	<code><Account>0100000002</Account></code>	<code><StatusCode>0</StatusCode></code> <code><AccountInfo></code> <code><Name>Test Name</Name></code> <code><Balance>100.00</Balance></code> <code></AccountInfo></code>
	<code><Account>0100000003</Account></code>	<code><StatusCode>0</StatusCode></code> <code><BankingDetails></code> <code><Payee></code> <code><Id>1111111111</Id></code> <code><Name>Test</Name></code> <code><Bank></code> <code><Name>Test Bank</Name></code> <code><Mfo>11111</Mfo></code> <code><Account>111111111111</Account></code> <code></Bank></code> <code></Payee></code> <code><Payer/></code> <code><Narrative></code> <code><Name>Оплата за услуги...</Name></code> <code><Vat>0</Vat></code> <code></Narrative></code> <code></BankingDetails></code>
	<code><Account>0100000004</Account></code>	<code><StatusCode>0</StatusCode></code> <code><ServiceId>2</ServiceId></code>
	<code><Account>0100000005</Account></code>	<code><StatusCode>-1</StatusCode></code>
Payment	<code><Account>0100000001</Account></code>	<code><StatusCode>0</StatusCode></code>
	<code><Account>0100000002</Account></code>	<code><StatusCode>0</StatusCode></code>
	<code><Account>0100000005</Account></code>	<code><StatusCode>-1</StatusCode></code>
Confirm	<code><Account>0100000001</Account></code>	<code><StatusCode>0</StatusCode></code> <code><OrderStatus>Accepted</OrderStatus></code>
	<code><Account>0100000002</Account></code>	<code><StatusCode>0</StatusCode></code> <code><OrderStatus>InProgress</OrderStatus></code>
	<code><Account>0100000005</Account></code>	<code><StatusCode>-1</StatusCode></code> <code><OrderStatus>Declined</OrderStatus></code>
Voucher	<code><Account>0100000001</Account></code>	<code><StatusCode>0</StatusCode></code> <code><OrderStatus>Accepted</OrderStatus></code> <code><VoucherInfo></code> <code><Serial>12345678</Serial></code> <code><Code>111144445555</Code></code> <code><ExpireDate>12.03.2011</ExpireDate></code> <code></VoucherInfo></code>
	<code><Account>0100000005</Account></code>	<code><StatusCode>-1</StatusCode></code> <code><OrderStatus>Declined</OrderStatus></code>

Cancel	<code><Account>010000001</Account></code>	<code><StatusCode>0</StatusCode></code> <code><OrderStatus>Declined</OrderStatus></code>
--------	---	---