XS2-шлюз
Реализация агентского XML-протокола платежного сервиса«X-Plat»

Лист контроля версий

Версия	Дата	Внесенные изменения	Исполнитель
1.0		Первая внешняя версия	
1.1	30.01.18	Добавлено описание статуса платежа Canceled	Пахомов П.Д.

Оглавление

Введение

Назначение документа

Данный документ описывает регламент технического взаимодействия XML-шлюза платежного сервиса X-plat с XML-клиентом агента по протоколу HTTPS.
Документ предназначен для разработчиков XML-клиентов.

Словарь терминов

Термин	Определение
Агент	Организация, осуществляющая прием и проведение платежей на основании агентского договора с системой X-plat либо лицо, ответственное за работу пунктов приема платежей (осуществляет установку и активацию программы-клиента, создание точек, добавление операторов и т.д.).
Точка или пункт приема платежей (ТПП)	Терминал самообслуживания, рабочее место (компьютер + программа) или терминал с удаленным доступом клиента на базе собственного ПО агента, который не имеет фактического адреса, с которых осуществляется проведение платежей.
Оператор	Пользователь автоматизированного рабочего места, программы, который производит отправку платежей.
Поставщик услуг (Провайдер)	Коммерческая организация или индивидуальный предприниматель, предоставляющие потребителям услуги, продающие товары от собственного имени, а также благотворительная организация, созданная для осуществления благотворительной деятельности, в пользу которого оператор осуществляет зачисление средств (проводит платеж).
Система	Автоматизированная учетная система процессинга X-plat, предназначенная для обеспечения дистанционной оплаты потребителем услуг поставщика.
Плательщик	Как правило, физическое лицо, желающее оплатить услуги предоставляемые провайдером (провайдерами) через систему.
Платежная информация	Данные необходимые для зачисления денежных средств на лицевой счет плательщика в учетной системе провайдера (сумма, провайдер, номер телефона/договора и т.п.), вводится оператором, предоставляется плательщиком.
XML-клиент (Клиент)	Программное обеспечение на стороне агента, работающее с системой X-plat через XML-шлюз.
Платеж	Денежные средства, принимаемые агентом от плательщиков в счет оплаты услуг в пользу поставщиков услуг.
Квитанция	Фискальный документ, являющийся подтверждением приема платежа и состоящий из определенного набора платежных реквизитов.

Описание типов, используемых в XML-шлюзе

Документами, которые определяют операции, параметры операций и типы параметров являются:

- Request.xsd – описание формата запроса к XML-шлюзу
- Response.xsd – описание формата ответа от XML-шлюза

На основании этих документов представлена нижеследующая информация.

Запрос к XML-шлюзу

Все запросы к XML-шлюзу следует отправлять на https://xs2.x-plat.ru/
Формат запроса описывается с помощью XSD-схемы Request.xsd. Найти схему можно по следующему адресу: https://xs2.x-plat.ru/Request.xsd

signType

Является инструкцией, которая указывает на то, каким способом формируется и передается подпись сообщения.
Состоит из:

- **xml_gate_auth**
  Является перечислением, которое указывает на то, какой тип подписи используется клиентом для аутентификации.
  Возможные значения перечисления:

Значение	Описание
rsa_sha512	Аутентификация происходит по электронно-цифровой подписи на базе RSA ключа (4096 байт) с использованием hash-функции SHA512.
sha512	Аутентификация происходит по hash, где в качестве hash-функции будет использоваться алгоритм SHA512.

- контейнера **[base64|hex]**
  Является перечислением, которое указывает на то, какой способ передачи байт-массива используется клиентом.
  Возможные значения перечисления:

Значение	Описание
base64	Информация представлена в виде 64-разрядного кода, на базе алгоритма Base64.
hex	Информация представлена в шестнадцатеричном виде.

- направления байт-массива
  rev - указывает на обратный порядок байт-массива полученной подписи. В случае отсутствия направление будет прямым.

Пример:
запрос подписан RSA ключом (4096 байт) с использованием hash-функции SHA512 и передан в hex-представлении:

```
<signature type="rsa_sha512_hex">4aec...3d3e3</signature>
```

запрос подписан RSA ключом (4096 байт) с использованием hash-функции SHA512 и передан в hex-представлении в обратном порядке:

```
<signature type="rsa_sha512_hex_rev">5aad...3cb8</signature>
```

authInfo

Содержит информацию для аутентификации и авторизации пользователя:

Параметр	Тип	Описание
authInfo.point	int	Точка приема.
authInfo.login	string	Логин оператора на указанной точке приема.
authInfo.password	string	SHA1-отпечаток от пароля оператора на указанной

		точке приема.
authInfo.signature.type	signType	Тип подписи, используемый XML-клиентом.
authInfo.signature	string	Содержимое подписи в Base64 Encoding.
authInfo.disposablecode	int	Код подтверждения. Необязательное поле, передается при необходимости.

Пример:

```
<header>
    <point>3392</point>
    <login>login</login>
    <password>fEqNCco3Yq9h5ZUg1D3CZJT41Bs=</password>
    <signature
type="sha512_hex">C2600AF711FED04A6…F843155B4ED3DDB741EC38</signature>
</header>
```

receipt

Описывает информацию в квитанции:

Параметр	Тип	Описание
receipt.date	dateTime	Дата приема денежных средств от плательщика. Если не указана, считается, что она равна дате регистрации платежа в процессинге.
receipt.point	string	Название точки, с которой пришел платеж. Используется в связке с предварительной регистрацией таких имен через метод Points.
receipt.number	long	Номер платежа на квитанции. Если не указан, считается, что он равен идентификатору платежа id в payment.

Пример:

```
<receipt date="2011-08-24T19:56:47" point="PointName" number="1256" />
```

field

Содержит информацию об одном платежном поле платежа:

Параметр	Тип	Описание
field	string	Значение платежного поля.
field.name	string	Идентификатор платежного поля.

Пример:

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

pointType

Является перечислением, которое указывает на то, какой тип точки используется клиентом.
Возможные значения перечисления pointType:

Значение	Описание
Cashin	Терминал самообслуживания
Client	Точка с кассиром (представителем агента)
InternetCashin	Терминал самообслуживания с удаленным доступом клиента.

Location

Содержит информацию о местоположении точки приема платежей агента:

Параметр	Тип	Описание
location.locality	string	КЛАДР-код.
location.type	location.Type	Тип расположения, перечисление (см. «Приложение 1»).
location.street	string	Название улицы.

location.building	string	Номер строения.

Пример:

```
<location locality="7700000000000" type="002001" street="StreetName"
building="1a" />
```

pointInformation

Содержит информацию о точке приема платежей агента:

Параметр	Тип	Описание
pointsCommand.name	string	Название регистрируемой точки
pointsCommand.type	pointType	Тип регистрируемой точки
pointsCommand.location	Location	Местоположение регистрируемой точки

Пример:

```
<register name="PointName" type="Cashin">
  <location locality="7700000000000" type="002001" street="StreetName"
building="1a" />
</register>
```

paymentInfo

Содержит информацию о платеже, посылаемом на обработку XML-шлюзу.

Параметр	Тип	Описание
paymentInfo.id	long	Уникальный идентификатор платежа на стороне XML-клиента.
paymentInfo.provider	char(4)	Идентификатор провайдера (до 4 символов).
paymentInfo.amount	decimal	Сумма к зачислению.
paymentInfo.user_amount	decimal	Сумма с клиента (сумма к зачислению + комиссия). Поле необязательное.
paymentInfo.receipt	receipt	Информация о чеке, выданном на платеж. Поле необязательное.
paymentInfo.field	field	Список платежных полей платежа. Каждое платежное поле представляет собой название и значение платежного поля.

Пример платежа с одним платежным полем:

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

Пример платежа с двумя платежными полями, суммой с клиента и информацией по чеку:

```
<payment id="4354" provider="test" amount="15.00" user_amount="20.00">
    <receipt date="2016-01-01T11:57:58" point="PointName" number="5"/>
    <field name="firstfield">983513424722</field>
    <field name="secondfield">12</field>
</payment>
```

registeredPaymentInfo

Содержит информацию о платеже, который уже был зарегистрирован в системе X-plat.

Параметр	Тип	Описание
registeredPaymentInfo.id	Long	Уникальный идентификатор платежа на стороне XML-клиента.

Пример:

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

checkCommand

Описывает команду на проверку возможности оплаты указанного платежа.

Параметр	Тип	Описание
checkCommand.timeout	int	Максимальное время ожидания прекращения обработки платежа проводящим сервером X-plat. Не указывается, если надо получить ответ сразу.
checkCommand.payment	paymentInfo	Описание платежа, возможность проведения которого необходимо проверить.

Пример асинхронного запроса на проверку:

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

Пример синхронного запроса на проверку:

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

payCommand

Описывает команду на проведение платежа, возможность проведения которого была проверена с помощью команды checkCommand.

Параметр	Тип	Описание
payCommand.timeout	Int	Максимальное время ожидания прекращения обработки платежа проводящим сервером X-plat.
payCommand.payment	registeredPaymentInfo	Описание платежа, который необходимо провести.

Пример асинхронного запроса:

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

Пример синхронного запроса:

```
<pay timeout="1000">
    <payment id="6437282" />
</pay>
```

cashinCommand

Описывает команду однофазного проведения платежа.

Параметр	Тип	Описание
cashinCommand.payment	paymentInfo	Описание платежа, который был послан на однофазное проведение.

Пример:

```
<cashin>
```

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

statusCommand

Описывает команду запроса текущего статуса зарегистрированного платежа.

Параметр	Тип	Описание
statusCommand.payment	registeredPaymentInfo	Описание зарегистрированного платежа, статус которого необходимо запросить.

Пример:

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

balanceCommand

Описывает команду запроса баланса агента.

Пример:

```
<balance />
```

provlistCommand

Описывает команду запроса списка доступных провайдеров.

Пример:

```
<provlist />
```

pointsCommand

Описывает команду регистрации точки приема платежей.

Параметр	Тип	Описание
pointsCommand.register	pointInformation	Описание точки, которую необходимо создать.

Пример:

```
<points>
  <register name="PointName" type="Cashin">
   <location locality="7700000000000" type="002001"
street="StreetName" building="1a" />
 </register>
</points>
```

request

Описывает запрос к XML-шлюзу.

Параметр	Тип	Описание
request.guid	Guid	GUID запроса
request.header	authInfo	Информация для аутентификации запроса
request.check или request.pay или request.cashin или request.status или	checkCommand payCommand cashinCommand statusCommand	Одна из возможных команд к XML-шлюзу.

request.balance или request.provlist или request.points	balanceCommand provlistCommand pointsCommand	

Ответ XML-шлюза

Формат ответа от XML-шлюза описывается с помощью XSD-схемы Response.xsd. Найти схему можно по адресу: https://xs2.x-plat.ru/Response.xsd

requestResultCode

Является перечислением, которое указывает результат обработки запроса XML-шлюзом.
Возможные значения перечисления:

Значение	Описание
Success	Запрос был успешно обработан XML-шлюзом.
NotPostRequest	Был выполнен не POST-запрос, XML-шлюз проигнорировал его.
XmlParseError	Ошибка при получении XML-документа из POST-содержимого. Текст ошибки содержится в поле Description.
XmlSchemaError	Ошибка в схеме XML-документа. Текст ошибки содержится в поле Description.
AuthError	Ошибка аутентификации оператора. Неправильно указана тройка point, login, password.
UserLock	Оператор заблокирован.
EdsError	Ошибка ЭЦП.
Denied	Вызов данного метода для авторизованного пользователя недоступен.
DisposableCode	Для успешной обработки запроса необходимо вызвать его с указанием кода подтверждения.
InternalError	Внутренняя ошибка XML-шлюза. Необходимо обратиться на integration@x-plat.ru за разъяснением.

paymentResultCode

Является перечислением, которое указывает результат обработки платежа XML-шлюзом.
Возможные значения перечисления:

Код ошибки	Check	Pay	Cashin	Status	Описание	Состояние платежа в системе	Действие
Success	+	+	+	+	Платеж успешно обработан и добавлен в процессинг	Платеж принят системой к обработке	Переходить к анализу PaymentState
ProviderNotExistsOrLock	+		+		Провайдер не существует или недоступен агенту	Платеж не принят системой	Обратиться к менеджеру за уточнением с указанием передаваемого идентификатора провайдера
AmountMinError	+		+		Указанная сумма платежа не попадает в допустимый диапазон	Платеж не принят системой	Обновить описание поставщика услуг (метод Provlist)

DealerBalanceLimit	+	+	+		У агента недостаточно средств для проведения платежа	Платеж не может быть проведен в сторону провайдера	Необходимо убедиться, что достаточно средств для проведения платежа. В противном случае обратиться к менеджеру
FieldsError	+		+		Неверно указаны платежные поля	Платеж не принят системой	Обновить описание поставщика услуг (метод Provlist)
RequiredFieldsError	+		+		Указаны не все обязательные платежные поля	Платеж не принят системой	Обновить описание поставщика услуг (метод Provlist)
PaymentNotFound		+		+	На момент обработки запроса сервису не удалось запросить статус платежа	Неизвестно, так как во время обработки запроса и формирования ответа платеж может быть вставлен другим запросом	Убедиться, что по команде Check или Cashin ранее был получен ответ, который был успешно обработан. Если это так, то требуется повторить запрос и при повторе ошибки обратиться в службу технической поддержки (support@x-plat.ru) для уточнения состояния по платежу и причины ошибки
PaymentNotCheck		+			Для данного платежа невозможна оплата. Не выполнен шаг проверки, либо проверка завершилась неудачно	Платеж был принят системой, но на момент обработки запроса первая фаза, возможно, не была завершена	Убедиться в том, что первая фаза была отработана корректно. Если это так, то требуется повторить запрос и при повторе ошибки обратиться в службу технической поддержки (support@x-plat.ru) для уточнения состояния по платежу и причины ошибки

PointNotFound	+		+		В теге receipt указана точка, которую сервису не удалось сопоставить с текущим пользователем	Платеж не принят системой	Убедиться, что указанная точка была зарегистрирована ранее с использованием метода Points. Если это так, то необходимо обратиться в службу технической поддержки (support@x-plat.ru) для уточнения состояния указанной точки
InternalError	+	+	+	+	Прочая внутренняя ошибка на стороне XML-сервиса	Неизвестно (может быть как принят так и не принят системой)	Необходимо обратиться на integration@x-plat.ru за разъяснением

paymentStateCode

Является перечислением, которое указывает текущее состояние платежа на сервере X-plat.
Возможные значения перечисления:

Значение	Финальное	Описание
ServerOk	Нет	Платеж принят к обработке сервером X-plat.
PsChecking	Нет	Проверяется возможность оплаты платежа.
PsCheckError	Да	Ошибка при проверке возможности проведения платежа.
PsChecked	Да	Платеж проверен, возможна оплата.
PsPaying	Нет	Платеж посылается на оплату.
PsStatus	Нет	Запрос статуса по платежу от внешнего платежного сервиса
PsPayError	Да	Платеж не прошел оплату.
PsOk	Да	Платеж прошел оплату.
Canceled	Да	Проведение платежа отменено

paymentStateType

Является перечислением, которое указывает тип состояния платежа.
Возможные значения:

Значение	Описание
NotFinal	Состояние не финальное – обработка платежа продолжается.
FinalFatal	Состояние финальное. Повторять платеж не имеет смысл, т.к. это приведет к такому же результату. Например, ошибка типа – «указанный номер не существует»

paymentParameter

Описывает список выходных параметров платежа.

Параметр	Тип	Описание
paymentParameter	string	Словесное описание результата обработки платежа.
paymentParameter.name	string	Название возвращаемого провайдером параметра

Пример:

```
<parameter name="ProviderPaymentId">167304946-ntUW</parameter>
```

paymentResult

Описывает результат обработки платежа XML-шлюзом.

Параметр	Тип	Описание
paymentResult	string	Словесное описание результата обработки платежа. Может быть пустым.
paymentResult.code	paymentResultCode	Код результата обработки платежа.
paymentResult.fatal	Boolean	Не использовать при обработке ответа. Ориентироваться на значение атрибута code.

Пример:

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

paymentState

Описывает состояние, в котором находится платеж на сервере X-plat.

Параметр	Тип	Описание
paymentState	string	Словесное описание состояния платежа. Содержит подробное описание ошибки, если оно есть.
paymentState.code	paymentStateCode	Текущее состояние платежа.
paymentState.type	paymentStateType	Тип состояния платежа.
paymentState.date	DateTime	Время обновления состояния платежа.

Пример без описания состояния:

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

Пример с описанием состояния:

```
<state code="PsChecking" type="NotFinal" date="2009-03-23T09:55:57.723">Проверка на сервере провайдера продолжается</state>
```

paymentStatus

Описывает результат обработки платежа XML-шлюзом.

Параметр	Тип	Описание
paymentStatus.result	paymentResult	Результат обработки платежа.
paymentStatus.id	Long	Идентификатор платежа со стороны XML-клиента.
paymentStatus.pt_id	Int	Идентификатор платежа со стороны X-plat.
paymentStatus.post_date	DateTime	Время регистрации платежа на сервере X-plat.
paymentStatus.state	paymentState	Текущее состояние платежа.
paymentStatus.parameters	paymentParameter	Выходные параметры по платежу. Может отсутствовать.

Пример:

```
<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>
```

Пример с параметрами:

```
<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

Представляет собой список объектов paymentStatus.

requestResult

Описывает результат обработки запроса XML-шлюзом.

Параметр	Тип	Описание
requestResult	String	Словесное описание результата обработки запроса XML-шлюзом.
requestResult.code	requestResultCode	Код результата обработки запроса XML-шлюзом.
requestResult.fatal	Boolean	Указывает на то, фатальная ли ошибка. Если да, то необходимо прекратить работу с XML-шлюзом.

response

Описывает ответ XML-шлюза.

Параметр	Тип	Описание
response.guid	Guid	GUID запроса, на который пришел этот ответ.
response.result	requestResult	Результат обработки запроса.
response.payment	paymentStatus	Результат выполнения команды. Присутствует, если код результата обработки запроса равен Success.
response.signature	String	Подпись сообщения в Base64 Encoding.

Для сообщения ответа дополнительно формируется подпись, соответствующая rsa_sha512 или sha512, которая затем включается в само сообщение.
Строка подписи ответа формируется путем конкатенации следующих параметров:
«Строка для подписи» + «GUID ответа»
«Строка для подписи» – все, что внутри тега «**response**». Сначала значения атрибутов, затем рекурсивно значение подэлементов и, если их нет, то значение элемента (значение между тегами). Если в рекурсивном поиске встретится элемент **state**, то значение его атрибута **date** в строку для подписи не включается.

Для следующего ответа:

```
<?xml version="1.0" encoding="utf-8"?>
<response                    guid="10a17dc3-1f64-43c6-9fc2-1faa0c5487a8"
xmlns="http://xs2.x-plat.ru/Response.xsd">
  <result code="Success" fatal="false" />
  <payment id="100000">
    <result code="Success" fatal="false" />
    <pt_id>395046716</pt_id>
    <post_date>2016-09-09T13:22:55</post_date>
    <state    code="PsChecked"    type="FinalFatal"    date="2016-09-
09T13:22:55" />
```

```
  </payment>
<signature>117851312A8871F2C08AEB7AF14AC3D4C020F55B2B515AA3E4CF09540C3
F709A1EB555D8D47E7558075F5D21E8DF7FC3080240E29ABD9C3E1A5155F5E9CEC3E1<
/signature>
</response>
```

строка для подписи будет выглядеть следующим образом:
«Successfalse100000Successfalse3950467162016-09-09T13:22:55PsCheckedFinalFatal10a17dc3-1f64-43c6-9fc2-1faa0c5487a8».

Аутентификация на XML-шлюзе

Аутентификация производится исходя из информации, представленной в теге header.
Сначала проверяется существование оператора с указанным логином на указанной точке приема, затем, если оператор был найден, происходит сверка SHA1-отпечатка пароля. На этом этапе XML-шлюз может вернуть следующую ошибку:

- AuthError – ошибка аутентификации, т.е. неверно указана тройка point, login, password.

После этого выбирается информация об агенте, которому принадлежит точка приема, и об операторе. Исходя из этой информации, проверяется следующее: не заблокирован ли агент, не заблокирован ли пользователь, разрешена ли работа с XML-шлюзом. На этом этапе XML-шлюз может вернуть следующие ошибки:

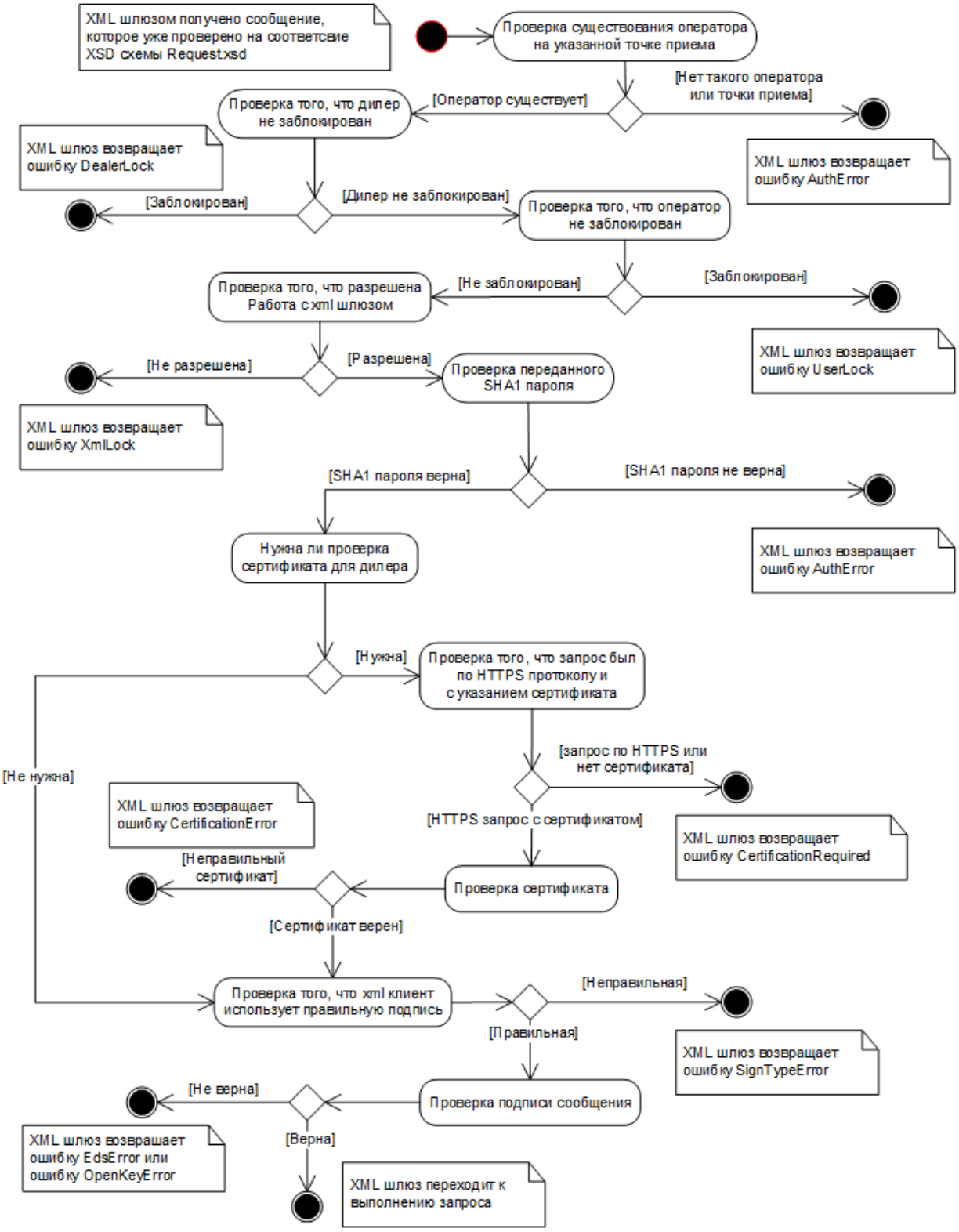
- DealerLock – агент, которому принадлежит указанная точка приема, заблокирован.
- UserLock – указанный оператор заблокирован.
- XmlLock – указанному оператору запрещена работа с XML-шлюзом.

После этого выбирается открытый ключ и тип подписи оператора. На основе этой информации проверяется, что XML-клиент использует правильный тип подписи, а так же то, что сообщение верно подписано. На этом этапе XML-шлюз может вернуть следующие ошибки:

- SignTypeError – XML-клиентом используется неправильный тип подписи сообщения.
- OpenKeyError – ошибка при получении открытого ключа оператора.
- EdsError – неверная подпись.

После успешного прохождения этих операций, считается, что сообщение прошло аутентификацию.

Общая схема проверки изображена на следующей диаграмме:



Подпись сообщения

Для сообщения дополнительно формируется подпись, соответствующая rsa_sha512 или sha512, которая затем включается в само сообщение (тег request.header.signature).

В общем случае этот процесс формирования подписи состоит из двух частей: формирование строки для подписи и непосредственно подпись полученной строки.

Формирование строки для подписи

Строка подписи формируется путем конкатенации следующих параметров: «Название метода» + «Строка для подписи от параметров» + «GUID запроса».
Примечание:
Все буквы в GUID запроса при подписи переводятся в нижний регистр, то есть вместо A, B, C, D, E и F используются a, b, c, d, e и f.

Название метода

Название метода зависит от вызываемого метода, а так же от параметра async (если он есть).

Метод XML-шлюза	Название метода
checkCommand	Check
payCommand	Pay
cashinCommand	Cashin
statusCommand	Status
balanceCommand	Balance
provlistCommand	Provlist
pointsCommand	Points

Строка для подписи от параметров

Для команды Check и Cashin формируется строка подписи от параметра paymentInfo. О том, как это сделать, написано в разделе «Формирование строки подписи для объекта PaymentInfo».
Для команд Pay и Status формируется строка подписи от параметра registeredPaymentInfo. О том, как это сделать, написано в разделе «Формирование строки подписи для объекта RegisteredPaymentInfo».

Формирование строки подписи для объекта PaymentInfo

Строка для подписи элемента **paymentInfo** формируется путем склеивания значения всех свойств, приведенных к строке, в следующем порядке: **PaymentId** + **Provider** + **Amount** + **UserAmount** *(если передается)* + **Field1.Name** + **Field1.Value** + … + **FieldN.Name** + **FieldN.Value**.
Обратите внимание, что строковое представление суммы платежа выглядит, как дробное число с 2 знаками после разделителя, где в качестве разделителя используется точка. Т.е., например, для суммы 5.5 строковое представление будет иметь следующий вид – «5.50», для суммы 95.34 – «95.34», для 90 – «90.00».

Пример:

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

Строка для подписи: «127823mega5.50phone9225498599».

Формирование строки подписи для объекта RegisteredPaymentInfo

Строка для подписи элемента **registeredPaymentInfo** формируется путем склеивания значения всех свойств, приведенных к строке, в следующем порядке: **PaymentId** + **«0»**.

Пример:

```
<payment id="127823" />
```

Строка для подписи: «1278230».

Подпись с помощью алгоритма с открытым ключом

В тег **signature** помещается электронно-цифровая подпись результата выполнения hash-функции от строки в кодировке WIN-1251, подписанная с помощью закрытого ключа.

Подпись с помощью hash-функции

В тег **signature** помещается результат выполнения hash-функции от строки в кодировке WIN-1251, получаемой в результате конкатенации строки для подписи с секретной фразой оператора.

Создание закрытых ключей

Для получения аутентификационных данных, используемых в заголовке запроса (точка, логин, пароль), необходимо воспользоваться приложением **Point Manager** https://www.x-plat.ru/soft/point/. При помощи данного приложения необходимо создать точку нужного типа, затем пользователя с предпочтительным способом аутентификации.

Если выбран тип с использованием hash-функции, то приложением будет предложено задать секретную фразу, которая в автоматическом режиме будет отправлена на сервер. Эту фразу необходимо использовать для формирования подписи сообщения.

В случае если выбран тип с использованием алгоритма с открытым ключом, приложение сгенерирует пару ключей. При этом публичный ключ в автоматическом режиме загрузится на сервер, а приватный необходимо использовать для формирования подписи запроса. Данный тип аутентификации является наиболее предпочтительным и криптостойким.

Операции поддерживаемые XML-шлюзом

XML-шлюз поддерживает следующие операции:

Операция	Назначение	Параметры	Возвращаемое значение
checkCommand	Проверка возможности проведения платежа. При этом платеж регистрируется в системе X-plat.	Описание платежа в виде XML-объекта PaymentInfo.	Ответ шлюза, содержащий объект PaymentStatus в виде XML.
payCommand	Проведение зарегистрированного платежа в системе X-plat.	Описание платежа в виде XML-объекта RegisteredPaymentInfo.	Ответ шлюза, содержащий объект PaymentStatus в виде XML.
cashinCommand	Однофазное проведение платежа в системе X-plat.	Описание платежа в виде XML объекта PaymentInfo	Ответ шлюза, содержащий объект PaymentStatus в виде XML
statusCommand	Получение текущего статуса по платежу, зарегистрированного в системе X-plat.	Описание платежа в виде XML-объекта RegisteredPaymentInfo.	Ответ шлюза, содержащий объект PaymentStatus в виде XML.
balanceCommand	Получение баланса агента.	Отсутствуют.	Ответ шлюза, содержащий объект BalanceResult в виде XML.
provlistCommand	Получение информации о доступных операторах.	Нужно ли получение картинок.	Ответ шлюза, содержащий объект ProvlistResult в виде XML.
pointsCommand	Регистрация точек приема платежей	Описание платежа в виде XML-объекта pointInformation	Ответ шлюза, содержащий объект PointsResult в виде XML.

Запрос баланса

Для получения текущего баланса агента, а так же размера доступного овердрафта, используется метод balanceCommand. Доступные средства, которые можно израсходовать, получаются путем складывания значения баланса и овердрафта.

Формат запроса

Стандартный запрос к XML-шлюзу, содержащий команду balanceCommand.

Пример:

```
<?xml version="1.0" encoding="utf-8"?>
<request xmlns="http://xs2.x-plat.ru/Request.xsd"
         guid="c17d8aae-ba95-46eb-911d-0b7d649c9a6b">
  <header>
    <point>3312</point>
    <login>login</login>
    <password>fEqNCco3Yq9h5ZUglD3CZJT41Bs=</password>
    <signature
type="sha512_base64">wmAK9xH+0Exzvx6f6mGBQyv…bTtPdt0HsOA==</signature>
  </header>
  <balance />
</request>
```

Подписываемая строка для сообщения формируется путем склеивания строки «Balance» и значения атрибута GUID корневого тега request.
Для примера, приведенного выше, подписываемая строка будет следующей:
«Balancec17d8aae-ba95-46eb-911d-0b7d649c9a6b»

Формат ответа

Ответ от XML-шлюза приходит в стандартном виде и содержит объект balanceResult в виде XML.

Пример успешного ответа:

```
<?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>g0S8Omi2D8g6pdoSd8t8Lbg4o8ykf/XvvfI5zrhFveK4LU1uaW+ryGnFH8N
DCEZmNsV/m0mo3Ker7B+eYiO4Jw==</signature>
</response>
```

Получение списка провайдеров

Для получения списка провайдеров, которые доступны агенту, используется метод provlistCommand. Ответом на данную команду является список провайдеров, разделенный на группы.
Метод имеет необязательный параметр logos (атрибут принимает значения «normal» или «small»), который может быть опущен. Если он указан, сервис пытается найти логотипы для групп и провайдеров нужного типа и отдать их в ответе.

Формат запроса

Стандартный запрос к XML-шлюзу, содержащий команду provlistCommand.

Пример запроса к XML-шлюзу:

```
<?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="sha512_hex">C2600AF711FED04C73BF1E9F…B4ED3DDB741EC38</signature>
  </header>
  <provlist logos="normal"/>
</request>
```

Строка для подписи формируется как название метода "Provlist", значение атрибута logos, если он передается, и значения атрибута GUID корневого тега request.
Для примера, приведенного выше, подписываемая строка будет следующей:
«Provlistnormal99ce944a-5660-45a2-a6c5-9138e5ea64a8»

Формат ответа

Ответ от XML-шлюза приходит в стандартном виде и содержит объект provlistResult в виде XML.

Пример удачного ответа на запрос:

```
<?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="Сотовая связь">
      <logo url="http://services.x-plat.ru/logos/group/1"
hash="3af1a2a3243814c4b154270c9be5feaf" />
    </group>
    <group id="4" title="Прочие услуги">
      <logo url="http://services.x-plat.ru/logos/group/4"
hash="4722c93af7b078c1d87af901040afeb1" />
    </group>
    <group id="5" title="Платежные системы">
      <logo url="http://services.x-plat.ru/logos/group/5"
hash="02c896a58e4d3235bb758263da73fe94" />
    </group>
```

```
    <group id="33" title="Банки">
      <logo url="http://services.x-plat.ru/logos/group/33"
hash="cef05bd5c6e57bd86bad64c8d59bd5a2" />
    </group>
    <provider id="mts" title="МТС" 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="Номер телефона" min="10" max="10"
regex="" format="8 (000) 000-0000;0;." />
    </provider>
    <provider id="hkp" title="Погашение кредита любого банка"
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="Мобильный телефон" min="10" max="10"
format="8 (000) 000-0000;0;." />
      <text id="lname" title="Фамилия" min="2" max="30" />
      <text id="fname" title="Имя" min="2" max="30" />
      <text id="mname" title="Отчество" min="1" max="30" />
      <number id="bik" title="БИК" min="9" max="9" />
      <number id="account" title="Номер счета" min="20" max="20"
regex="^\d{20}$" />
      <text id="additional" title="Номер договора\карты
(дополнительно)" optional="true" min="0" max="200" />
      <number id="passport" title="Серия и номер паспорта" min="10"
max="255" regex="" />
    </provider>
    <provider id="tur" 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/tur?type=normal"
hash="854a32dcae45accf2671d5355866c784"/>
      <number id="dogovor" title="Номер договора" min="1" max="50" />
      <text id="dogovor_surname" title="Фамилия туриста" min="1"
max="255" />
    </provider>
  </provlist>
<signature>117851312A8871F2C08AEB…ABD9C3E1A5155F5E9CEC3E1</signature>
</response>
```

Внутри provlist могут быть теги group и provider, оба эти тега имеют следующие параметры:

- id - идентификатор объекта, уникальный в рамках типа (у группы может быть идентификатор такой же как и у провайдера, но у двух групп/провайдеров не может быть одного и того же идентификатора)
- title - название объекта для отображения на клиенте
- group - идентификатор группы или групп, в которую\ые вложен объект. У элементов group может отсутствовать. Список групп перечисляется через пробел.
- logo - описание логотипа, если он есть для данной группы\провайдера. Логотип содержит следующие поля:
  - url - откуда можно загрузить сам логотип
  - hash - hash файла с логотипом (по нему можно проверять, стоит ли обновлять логотип)

Пример вложенности групп и провайдеров:

```
<provlist>
  <group id="1" title="Сотовая связь" />
  <group id="3" title="Интернет" />
  <group id="24" title="Дальсвязь" group="1" />
  <provider id="mts" title="МТС" group="1" ...
  <provider id="bee" title="Билайн" groups="1 3" ...
  <provider id="d001" title="Дальсвязь Центр" group="24" ...
  <provider id="d002" title="Дальсвязь Восток" group="24" ...
  <provider id="d003" title="Дальсвязь Запад" group="24" ...
</provlist>
```

На клиенте должно отображаться следующим образом:

```
- g.Сотовая связь
-- g.Дальсвязь
---- p.Дальсвязь Центр
---- p.Дальсвязь Восток
---- p.Дальсвязь Запад
-- p.МТС
-- p.Билайн
- g.Интернет
-- p.Билайн
```

У объекта provider так же есть дополнительные свойства:

- currency - идентификатор валюты провайдера;
- min - минимальная сумма платежа по провайдеру;
- max - максимальная сумма платежа по провайдеру;
- schema - указывает на то, по какой схеме необходимо принимать платежи в пользу данного провайдера. По умолчанию (если в ответе нет этого атрибута) считается, что провайдер работает по схеме default. В противном случае указывается название схемы, по которой работает провайдер. Если клиент не поддерживает работу по указанной схеме, ему рекомендуется игнорировать данного провайдера. Схем может быть перечислено несколько (разделяются через пробел) в порядке приоритета использования, если клиент не поддерживает первую, он должен проверить вторую и так далее;
- tags - специальные флаги, которые могут быть у провайдера. На данный момент подразумевается использование двух:
  - default:check - для default-схемы необходимо сделать онлайн проверку;
  - default:online - для default-схемы необходимо сделать онлайн оплату.
- limit - лимиты, установленные для этого провайдера. Может содержать атрибуты. Если их нет, то по этому провайдеру нет установленных лимитов:
  - max_amount_of_payment - максимальная сумма платежа, после которой он попадет в ручную проверку;
  - provider_daily_limit - максимальная сумма платежей в сутки по провайдеру;
  - masterkey_daily_limit - максимальная сумма платежей в сутки по ключевому полю (телефон, номер счета и т.д.).

Кроме этого внутри тега provider идут теги, описывающие платежные поля провайдера. На данный момент поддерживается три типа:

- number - платежное поле, для ввода которого необходимо отобразить цифровую клавиатуру;
- text - платежное поле, для ввода которого необходимо отобразить полную клавиатуру;
- list - платежное поле, значение которого выбирается из определенного списка.

Все три типа поля имеют следующие свойства:

- id - идентификатор платежного поля, используется как значение name платежного поля платежа при отправке;
- title - название платежного поля для отображения на клиенте;
- optional - указывает на необязательность поля. Если не передается, считается что поле обязательное, т.е. значение этого поля false.

Дальше каждый тип платежного поля имеет дополнительные свойства

- number и text:
  - min - минимальная длина платежного поля;
  - max - максимальная длина платежного поля;
  - regex - проверочное регулярное выражение для значения платежного поля. Может отсутствовать, в таком случае не передается;
  - format - формат для отображения, используется текущий;
- list - внутри перечисляется список возможных значений в тегах item, каждый их которых содержит:
  - key - значение платежного поля, которое надо передавать в value при отправке платежа;
  - text() - название выбора платежного поля для отображения на клиенте.

Регистрация ТПП.

Для регистрации новых ТПП используется метод pointsCommand.
Метод содержит тег register с обязательными параметрами: name (название ТПП, должно быть уникальным для каждой точки, которая регистрируется методом Points), type (тип регистрируемой ТПП), в который вложен тег location с обязательными параметрами locality (КЛАДР код из официального справочника), type (тип расположения, полный список возможных значений которых см. в приложении 1), street (название улицы), building (номер строения).
Тип регистрируемой ТПП type может принимать значения:

- Cashin - Терминал самообслуживания. Клиент имеет непосредственный доступ к ТПП.
- Client - ТПП с кассиром (представителем агента). Работа с программным обеспечением производит сотрудник, а не непосредственно клиент.
- InternetCashin - Терминал самообслуживания с удаленным доступом клиента. Интернет-кошелек, интернет-банкинг. У ТПП нет фактического адреса расположения, вместо него используется некий юридический.

Формат запроса

Стандартный запрос к XML-шлюзу, содержащий команду pointsCommand.

Пример:

```
<?xml version="1.0" encoding="utf-8"?>
<request xmlns="http://xs2.x-plat.ru/Request.xsd"
         guid="8e5f4f35-0cc9-4563-b928-7c4ac264cb4b">
 <header>
  <point>45515</point>
  <login>gate</login>
  <password>fEqNCco3Yq9h5ZUglD3CZJT4lBs=</password>
  <signature
type="sha512_base64">wmAK9xH+0Exzvx6f6…CX4QxVbTtPdt0HsOA==</signature>
 </header>
 <points>
  <register name="PointName" type="Cashin">
   <location locality="5600000100000" type="001" street="Street"
building="1a" />
 </register>
</points>
</request>
```

Подписываемая строка для сообщения формируется путем склеивания строки «Points» и строка подписи для каждого тега register в той последовательности, в которой они идут.
Строка для подписи тега register формируется путем склеивания значений атрибутов: **name** + **type** + **location.locality** + **location.type** + **location.street** + **location.building**.
Для примера, приведенного выше, подписываемая строка будет следующей:
«PointsPointNameCashin5600000100000001Street1a8e5f4f35-0cc9-4563-b928-7c4ac264cb4b»

Формат ответа

Ответ от XML-шлюза приходит в стандартном виде и содержит объект pointsCommand в виде XML.

Пример успешного ответа:

```
<?xml version="1.0" encoding="utf-8"?>
<response guid="8e5f4f35-0cc9-4563-b928-7c4ac264cb4b"
```

```
          xmlns="http://xs2.x-plat.ru/Response.xsd">
  <result code="Success" fatal="false" />
  <points>
    <register name="PointName" status="Registered" />
  </points>
<signature>g0S8Omi2D8g6pdoSd8t8Lbg…sV/m0mo3Ker7B+eYiO4Jw==</signature>
</response>
```

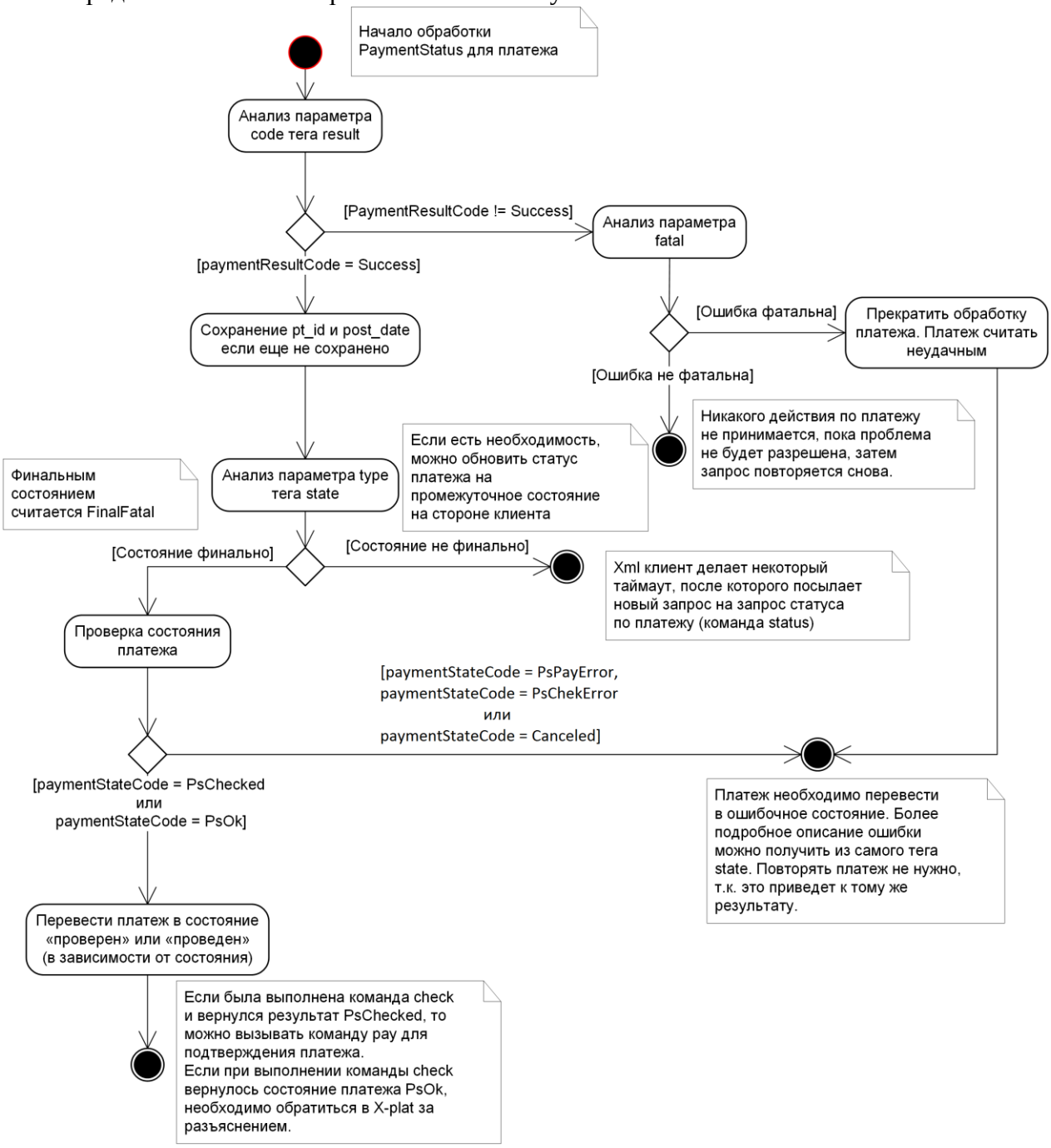
- name - название регистрируемой ТПП;
- status - означает статус регистрации указанной ТПП:
  - Registered - ТПП успешно зарегистрирована в системе, можно указывать ее в receipt;
  - LocalityNotFound - не удалось зарегистрировать ТПП, т.к. по значению location.locality ничего не удалось найти;
  - ManyLocalitiesFound - не удалось зарегистрировать ТПП, т.к. по значению location.locality нашлось более одного населенного пункта. В случае этого кода, в ответе так же будут присутствовать теги locality, что бы на стороне клиента можно было выбрать нужный населенный пункт;
- locality - присутствуют только в случае кода статуса ManyLocalitiesFound. Перечисляют найденные населенные пункты по location.locality из запроса, среди которых сервис не смог выбрать однозначное соответствие:
  - code - если послать этот код в качестве location.locality, будет выбран именно этот населенный пункт;
  - region - название региона для населенного пункта;
  - name - название самого населенного пункта.

Проведение платежей

Обработка объекта PaymentStatus

Все ответы на запросы по проведению платежей содержат объект PaymentStatus, который содержит информацию о текущем состоянии платежа на сервере X-plat. Именно поэтому понимание обработки этого объекта XML-клиентом является важным для понимания всего процесса проведения платежа.

Ниже представлена схема обработки объекта PaymentStatus:

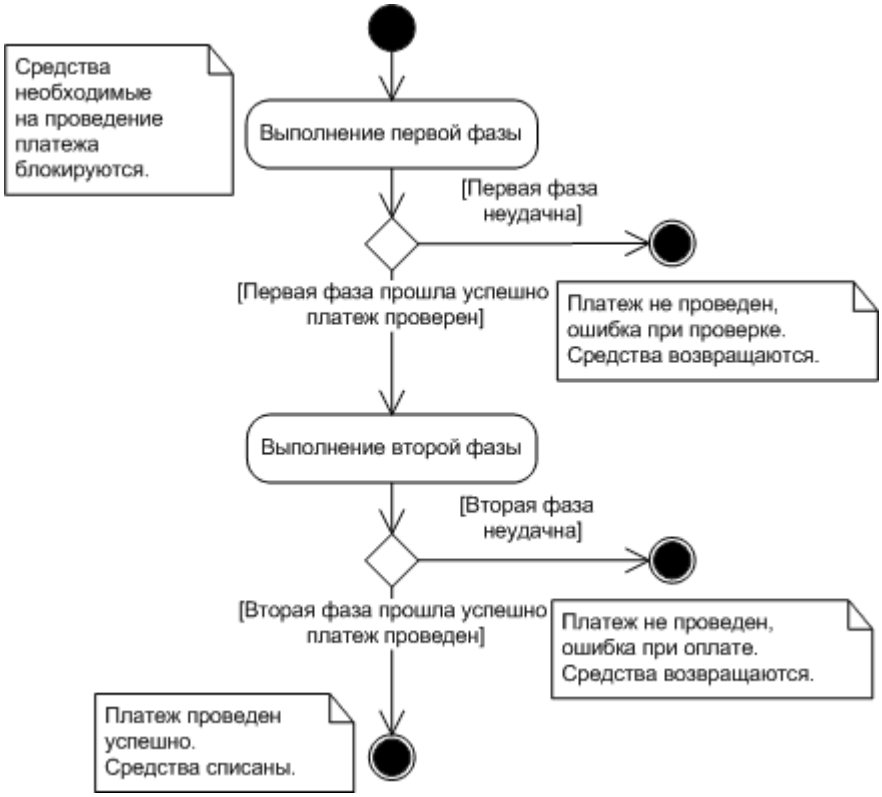


Двухфазное проведение платежа

Двухфазная обработка платежа состоит из двух фаз. На первой фазе проверяется возможность проведения платежа. Если она прошла успешно (платеж корректен и может быть проведен), то выполняется вторая фаза, после которой платеж непосредственно выполняется.

При выполнении первой фазы, на балансе агента блокируются средства, необходимые для выполнения платежа. Если первая фаза прошла неуспешно, то средства возвращаются агенту. После успешного проведения второй фазы, средства списываются окончательно. При неуспешном выполнении второй фазы, средства возвращаются агенту.

Общая схема проведения платежа



Первая фаза. Проверка возможности проведения платежа

Формируется запрос для отсылки команды checkCommand с описанием проверяемого платежа. После этого запрос посылается на XML-шлюз. В случае получения не финального состояния платежа, необходимо через некоторый промежуток запросить состояние зарегистрированного платежа и повторять это до тех пор, пока состояние платежа не будет финальным. Затем полученный ответ обрабатывается XML-клиентом.

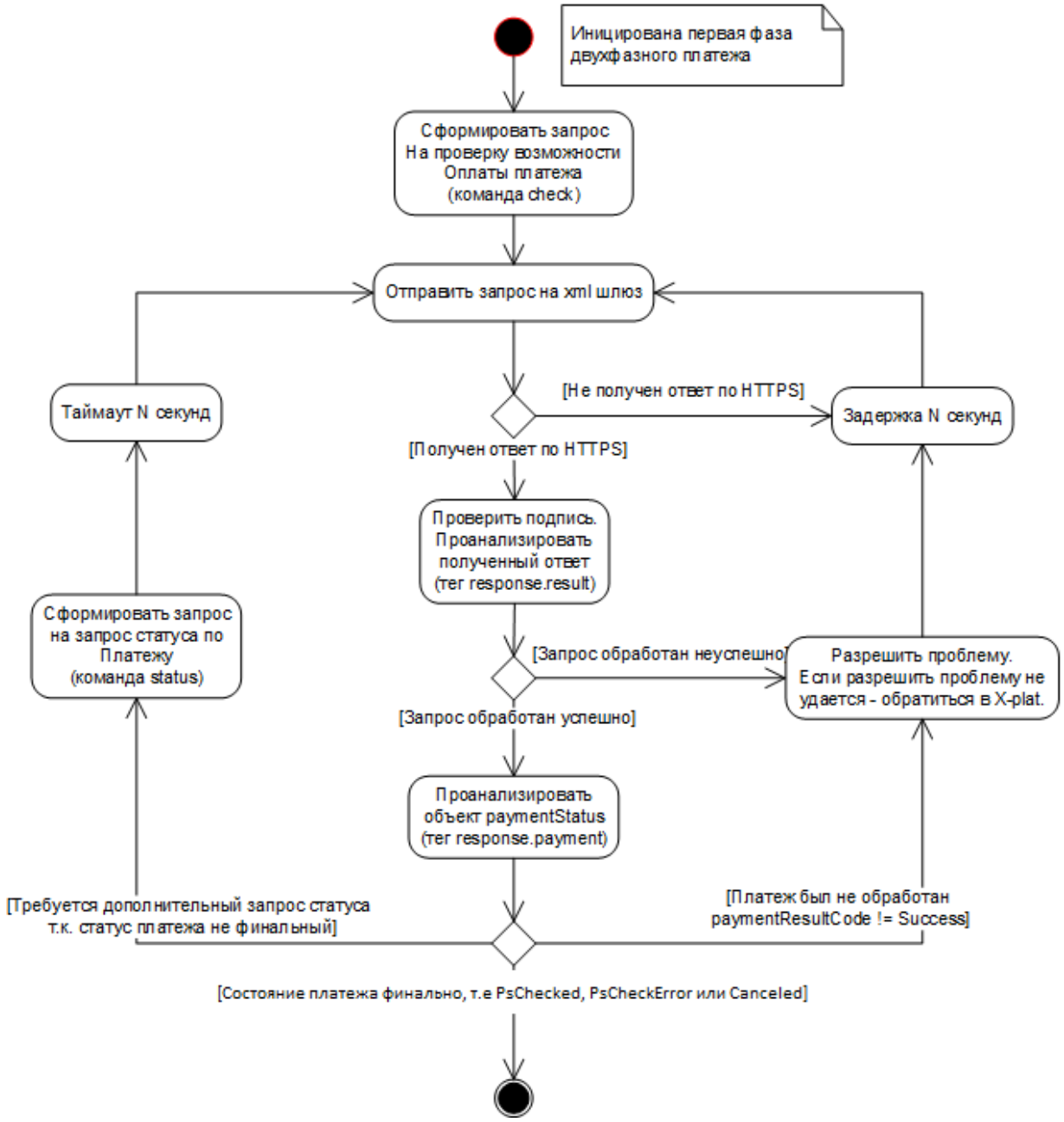
Пример запроса для проведения первой фазы:

```
<?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="sha512_hex">C2600AF711FED04C73BF1…155B4ED3DDB741EC38</signature>
  </header>
  <check timeout="100">
    <payment id="6437282" provider="bee" amount="1.00">
      <field name="phone">9035174909</field>
    </payment>
  </check>
</request>
```

Пример успешного ответа (состояние конечное):

```
<?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>117851312A8871F2C08AEB7AF1…9C3E1A5155F5E9CEC3E1</signature>
</response>
```

Схема проведения первой фазы двухфазного платежа:



Вторая фаза. Проведение платежа

Если первая фаза прошла удачно (состояние платежа PsChecked), то для подтверждения проведения платежа необходимо выполнить вторую фазу.
Для этого формируется запрос для отсылки команды payCommand с описанием зарегистрированного платежа. После этого запрос посылается на XML-шлюз. В случае получения не финального состояния платежа, необходимо через некоторый промежуток запросить состояние зарегистрированного платежа и повторять это до тех пор, пока состояние платежа не будет финальным. Затем полученный ответ обрабатывается XML-клиентом.

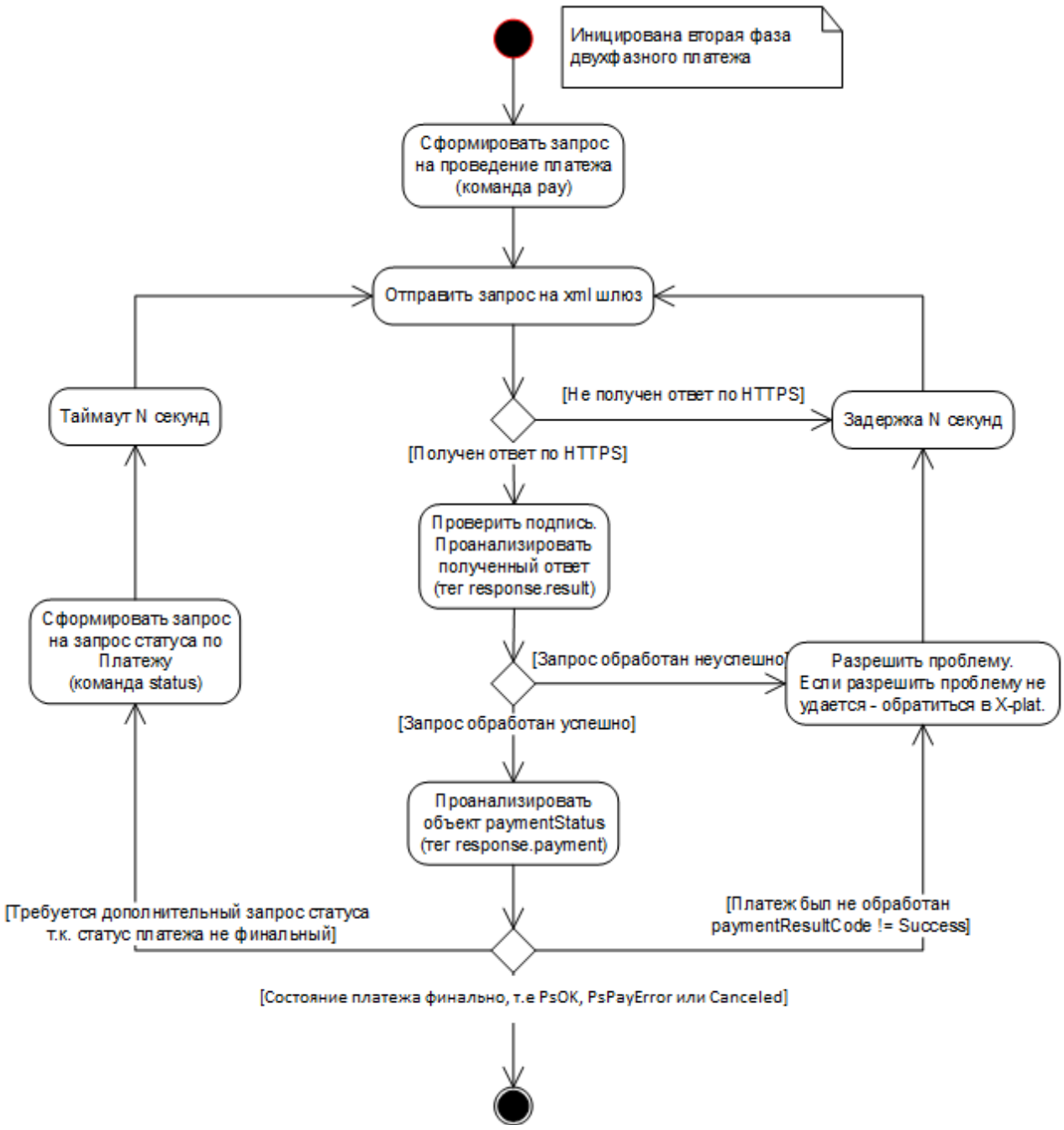
Пример запроса для проведения второй фазы:

```
<?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>fEqNCco3Yq9h5ZUglD3CZJT41Bs=</password>
    <signature
type="sha512_hex">C2600AF711FED04C73BF1…155B4ED3DDB741EC38</signature>
  </header>
  <pay timeout="100">
    <payment id="290361" />
  </pay>
</request>
```

Пример успешного ответа (состояние конечное):

```
<?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>117851312A8871F2C08AEB7AF1…9C3E1A5155F5E9CEC3E1</signature>
</response>
```

Схема проведения второй фазы двухфазного платежа:



Обработка ответа и ошибок

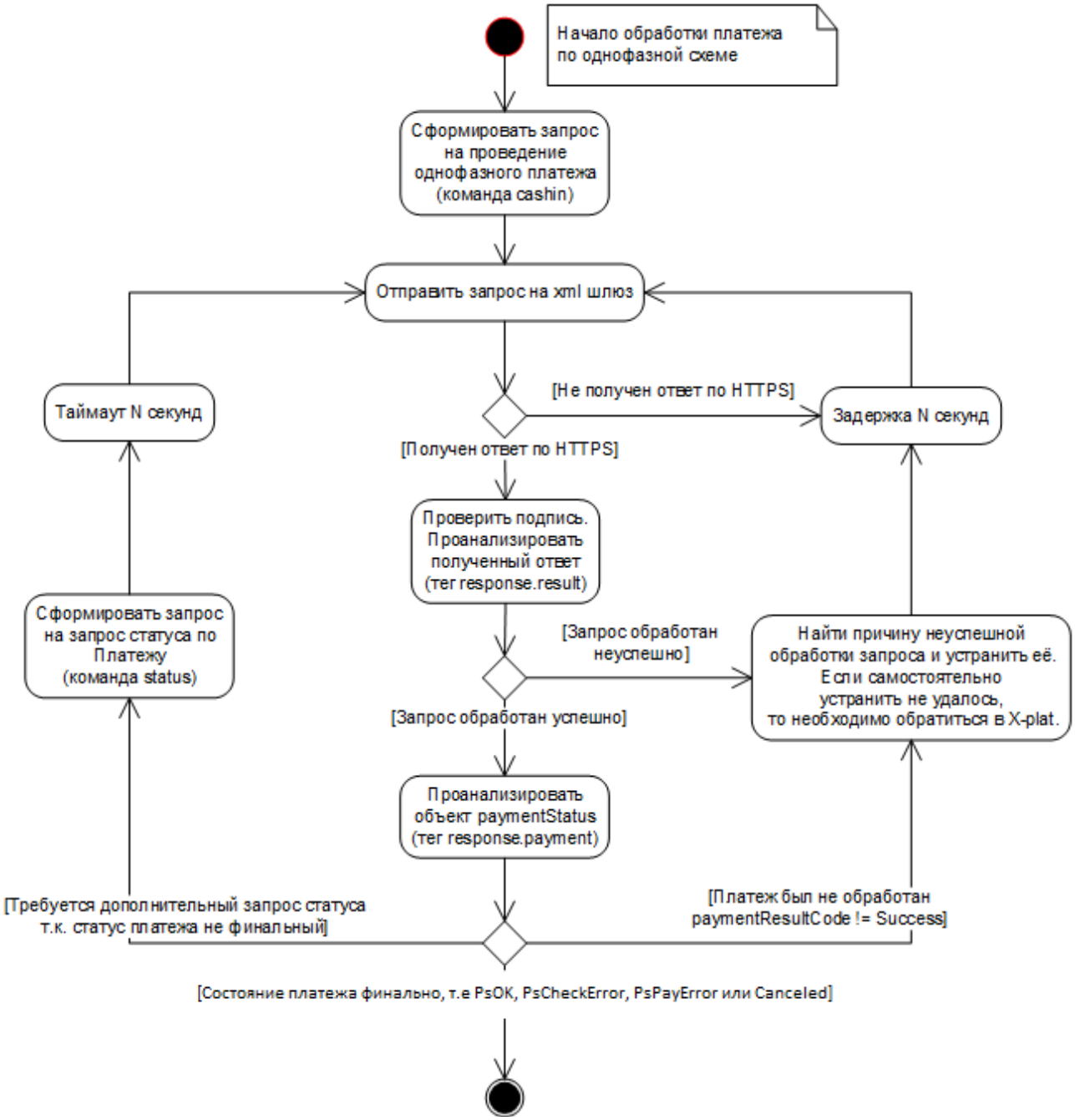
Все используемые для двухфазного проведения платежа команды (check, pay, status) возвращают один тип ответа, который содержит объект PaymentStatus.
В случае возвращения ошибки обработки запроса (тег response.result), необходимо разрешить возникшую проблему. После разрешения проблем можно повторить запрос снова.
В случае возвращения ошибки обработки платежа, действовать в соответствии с типом ошибки.

Однофазное проведение платежа

Однофазное выполнение платежа проходит в одну фазу. В отличие от двухфазного проведения платежа, от XML-клиента не требуется подтверждения платежа. После того, как возможность оплаты будет проверена проводящим сервером X-plat, платеж автоматически будет послан на отправку.

При однофазном проведении платежа средства, необходимые на проведение платежа, блокируются на счете агента. Если проведение платежа завершилось ошибкой, то средства возвращаются на счет агента. При успешном проведении платежа средства списываются окончательно.

Общая схема проведения однофазного платежа



Посылка запроса

Формируется запрос для отсылки команды cashinCommand с описанием платежа на оплату. После этого запрос посылается на XML-шлюз. В случае получения не финального состояния платежа, необходимо через некоторый промежуток времени запросить состояние зарегистрированного платежа и повторять это до тех пор, пока состояние платежа не будет финальным. Затем полученный ответ обрабатывается XML-клиентом.

В случае возвращения ошибки обработки запроса (тег response.result), необходимо разрешить возникшую проблему. После разрешения проблем можно повторить запрос снова.

В случае возвращения ошибки обработки платежа, действовать в соответствии с типом ошибки.

В случае возвращения неуспешного финального состояния платежа, платеж можно попытаться перепровести с новым идентификатором со стороны клиента, если тип состояния был FinalNotFatal (означает, что платеж не удалось провести, но при повторной посылке он может и провестись).

Пример запроса для однофазного проведения платежа:

```
<?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>fEqNCco3Yq9h5ZUglD3CZJT41Bs=</password>
    <signature
type="sha512_hex">C2600AF711FED04C73BF1E9FE…4ED3DDB741EC38</signature>
  </header>
  <cashin>
    <payment id="6437282" provider="bee" amount="1.00">
      <field name="phone">9035174909</field>
    </payment>
  </cashin>
</request>
```

Пример успешного ответа (состояние не конечное):

```
<?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="NotFinal" date="2008-09-
16T00:27:19.95" />
  </payment>

<signature>117851312A8871F2C08AEB7AF14A…3E1A5155F5E9CEC3E1</signature>
</response>
```

Комбинирование методов проведения платежа

Если сумма платежа заранее неизвестна, но есть необходимость выполнить проверку платежа перед его проведением и избежать холдирования большого объема денежных средств на счете агента, возможно комбинирование двух методов Check и Cashin.
Алгоритм комбинирования следующий:

- формируется запрос для отсылки команды checkCommand с описанием проверяемого платежа на минимально возможную сумму по текущему провайдеру. После этого запрос посылается на XML-шлюз. В случае получения не финального состояния платежа, необходимо через некоторый промежуток запросить состояние зарегистрированного платежа и повторять это до тех пор, пока состояние платежа не будет финальным;
- при получении успешного финального состояния платежа «PsChecked» формируется запрос для отсылки команды cashinCommand с описанием платежа (с новым номером транзакции на стороне XML-клиента) на полную сумму. После этого запрос посылается на XML-шлюз.

CheckCommand и cashinCommand отличаются друг от друга суммой платежа и номером транзакции. Вызов cashin с тем же номером транзакции, что и check, вернет текущее состояние платежа в системе. Так как данный сценарий не предполагает отправку команды pay, то созданная при check транзакция переведется в финальное ошибочное состояние и захолдированные денежные средства вернутся агенту.

Контактная информация

По всем возникающим вопросам, связанным с работой шлюза, необходимо обращаться по следующей электронной почте:
integration@x-plat.ru

Приложение 1. Типы расположения ТПП

location.type	Значение
001	Жилое здание
Учебно-воспитат. и научно-исслед. учреждения	
002001	Детский сад
002002	Школа
002003	Гимназия
002004	Лицей
002005	Колледж
002006	Институт
002007	Университет
002008	Академия
002009	Аэроклуб
002010	Автошкола
002011	Оборонное учебное заведение
002012	Научно-исследовательское учреждение
002013	Другое
Здравоохранение и социальное обслуживание населения	
003001	Больница
003002	Госпиталь
003003	Медицинский центр
003004	Поликлиника
003005	Аптека
Предприятия розничной торговли	
005001	Магазины: Торговый центр
005002	Магазины: Гипермаркет (не менее 25 касс)
005003	Магазины: Супермаркет (не менее 10 касс)
005004	Магазины: Минимаркет (от 2 до 9 касс)
005005	Магазины: Магазин
005006	Павильон/киоск
005007	Автосалон
005008	Салон связи
005009	Рыночный комплекс
005010	АЗС

Предприятия общественного питания	
006001	Ресторан
006002	Кафе
006003	Пиццерия
006004	Столовая
006005	Бар
Бытовое обслуживание населения	
007001	Баня
007002	Парикмахерская
007003	Ателье
007004	Салон красоты
007005	Автомойка
007006	Автосервис
007007	Химчистка
Коммунальное хозяйство	
008001	ЖЭК
008002	РЭУ
Предприятия связи	
009001	Отделение почтовой связи
009002	Переговорный пункт
Транспорт	
010001	Автобусный вокзал
010002	Аэровокзал
010003	Железнодорожный вокзал
010004	Речной вокзал
010005	Морской вокзал
010006	Аэропорт
010007	Речной порт
010008	Морской порт
010009	Таксопарк
010010	Автобусный парк
010011	Трамвайное депо
010012	Троллейбусный парк
010013	Ж/д и авиакассы
Досуг	

011001	Спортивный комплекс
011002	Спортивный клуб
011003	Бассейн
011004	Библиотека
011005	Музей
011006	Выставка
011007	Клуб (центр досуга, казино, игровой зал)
011008	Дом культуры
011009	Театр
011010	Концертный зал
011011	Кинотеатр
011012	Цирк
011013	Комплекс аттракционов
011014	Интернет клуб/кафе
011015	Аквапарк
Здания и помещения для временного пребывания	
012001	Гостиница
012002	Мотель
012003	Санатории
012004	Пансионат
012005	Дом отдыха
012006	Турбаза
012007	Общежитие учебного заведения
Административное здание	
013001	Здание института власти
013002	Административно-производственное здание
013003	Административно-хозяйственное здание
013004	Бизнес центр (Деловой центр)
013005	Банк (офис, доп. офис, отделение)
013006	Посольство
Производственное здание	
014001	Завод
014002	Фабрика
014003	База
014004	Склад

014005	ТЭЦ
015	Уличные терминалы
016000	Метро (Вестибюль станции)
017	Виртуальные точки (не имеющие географического адреса)