СТАНДАРТ БАНКА РОССИИ

ОТКРЫТЫЕ ПРОГРАММНЫЕ ИНТЕРФЕЙСЫ

Общие требования к прикладным стандартам.

Технический стандарт

Дата введения: 2026-10-01
Версия: 1.0.0

Москва
2025

Оглавление

1 Предисловие

Настоящий стандарт разработан Ассоциацией развития финансовых технологий (Ассоциацией ФинТех) при участии Центрального банка Российской Федерации (Банка России).

ПРИНЯТ И ВВЕДЕН в действие приказом Банка России от 19 декабря 2025 года № ОД-2889 «О введении в действие приказа Банка России СТО БР «Открытые программные интерфейсы. Общие требования к прикладным стандартам. Технический стандарт».

2 Введение

Настоящий стандарт содержит принципы и подходы, общие для всех спецификаций Открытых API, применяемых к обмену данными при взаимодействии через Открытые программные интерфейсы.

3 Область применения

Настоящий стандарт рекомендован к использованию организациями при обмене финансовыми сообщениями в среде Открытых программных интерфейсов. Настоящий стандарт предназначен для:

- участников обмена информацией о финансовых продуктах, счетах и других финансовых инструментах Пользователя, а также связанной с ними информацией;
- разработчиков информационного и программного обеспечения.

Положения настоящего стандарта носят рекомендательный характер и применяются совместно со следующими документами:

- Методические рекомендации МР.26.2.002-2024 ТК 26 «Информационная технология. Криптографическая защита информации. Использование российских криптографических алгоритмов в протоколах OpenID Connect (далее МР - OIDC).
- Стандарт Банка России СТО БР ФАПИ.СЕК-1.6-2024 «Безопасность финансовых (банковских) операций. Прикладные программные интерфейсы обеспечения безопасности финансовых сервисов на основе протокола OpenID» (далее - ФАПИ.СЕК).
- Стандарт Банка России СТО БР ФАПИ.ПАОК-1.0-2024 «Безопасность финансовых (банковских) операций. Обеспечение безопасности финансовых сервисов при инициации OpenID Connect клиентом потока аутентификации по отдельному каналу» (далее - ФАПИ.ПАОК).
- Стандарт Банка России СТО БР «Открытые программные интерфейсы. Общие положения».
- Стандарт Банка России СТО БР «Открытые программные интерфейсы. Глоссарий».
- Спецификация OpenAPI (Открытый стандарт OPENAPI Iniative. Подробнее на https://spec.openapis.org/)
- Другими документами комплекса стандартов Открытых API, размещенными официальном сайте Банка России в информационно-телекоммуникационной сети «Интернет».

4 Термины и определения

В настоящем стандарте применяются термины и определения в соответствии с ФАПИ.СЕК, ФАПИ.ПАОК, «Открытые программные интерфейсы. Глоссарий».

5 Общие положения

5.1 Средства передачи информации

Информационный обмен между участниками осуществляется с помощью электронных сообщений, формируемых как на стороне ПУ так и на стороне СПУ и передаваемых посредством Открытых программных интерфейсов.

5.2 Физический уровень описания сообщений

На физическом уровне при проектировании сообщений используется Спецификация OpenAPI 3.0 в формате YAML.

5.3 Использование международного стандарта ISO 20022

При проектировании сообщений реквизитный состав и типы данных полезной нагрузки определяются на основании реквизитов и определений финансовых сообщений стандартов Банка России для национальной платежной системы, сформированных в соответствии с методологией международного стандарта ISO 20022.

Принципы, которые применялись при использовании элементов и компонентов сообщений из стандарта ISO 20022:

- следование семантике сообщений из ISO 20022;
- уменьшение иерархии схем сообщений API для удобства участников среды Открытых программных интерфейсов;
- адаптация наименований и состава элементов схем сообщений API для удобства участников среды Открытых программных интерфейсов;
- использование типов данных элементов сообщений из ISO20022.

6 Безопасность и контроль доступа

6.1 Токен доступа и область действия

Для получения доступа к защищенным ресурсам в среде Открытых программных интерфейсов вызов к каждой конечной точке API должен производиться с токеном доступа. Токен доступа должен передаваться в заголовке Authorization в виде: Authorization: Bearer согласно раздела 6.7.5.1 MP OIDC.

Требуемая область действия токена доступа (параметр scope, подробное применение представлено в ФАПИ.СЕК) и тип предоставления доступа должны быть определены в каждой спецификации API.

6.2 Типы предоставления доступа

6.2.1 Тип доступа по учетным данным OIDC клиента (client credentials)

Тип доступа client credenials - используется для идентификации СПУ в контексте, где отсутствует конечный Пользователь и где не требуется его согласие.

Поддержка данного типа доступа осуществляется при аутентификации приложения клиента СПУ на сервере авторизации ПУ с использованием механизма аутентификации в соответствии с требуемым прикладным стандартом уровнем профиля безопасности API (требования профилей определены в ФАПИ.СЕК).

6.2.2 Тип доступа с кодом авторизации (authorization code)

Тип доступа authorization code используется в рамках сценария Гибридный поток или поток с кодом авторизации в соответствии с требуемым прикладным стандартом уровнем профиля безопасности API (требования профилей определены в ФАПИ.СЕК).

6.2.3 Поток аутентификации по отдельному каналу

В случае, когда Пользователь использует для аутентификации на стороне ПУ устройство, отличное от того, где он взаимодействовал с СПУ, то такая ситуация в среде Открытых программных интерфейсах регламентируется ФАПИ.ПАОК.

6.3 Защита взаимодействия на транспортном уровне

Все взаимодействия клиента с сервером авторизации и ресурсным сервером должны осуществляться по TLS-соединению с реализацией российских криптографических алгоритмов. Допустимы версии протокола: TLS 1.2 (ГОСТ Р 1323565.1.020–2020) и TLS 1.3 (ГОСТ Р 1323565.1.030–2020), согласно требованиям, зафиксированным в MP - OIDC и разделе 5.8.3 ФАПИ.СЕК.

6.4 Подпись и шифрование сообщений

Подпись и шифрование сообщений при необходимости реализуются в формате JWS/JWE, как определено в подразделе 5.7 ФАПИ.СЕК. Решение о включении данных механизмов определяется в прикладном стандарте. Подпись должна формироваться как отделяемая (detached) и передаваться в заголовке HTTP x-jws-signature.

6.5 Авторизация согласия

Процесс определения Пользователем разрешений для СПУ является авторизацией согласия.

СПУ запрашивает у ПУ создание ресурса согласия на предоставление сервиса с помощью доступа client credentials, тем самым подтверждает намерение Пользователя получить сервис. В зависимости от предоставляемого сервиса запрос может содержать дополнительную информацию о контексте взаимодействия с Пользователем в разделе Data или Risk полезной нагрузки, что позволяет ПУ оценивать риски, связанные с запросом и принимать решение об необходимости проводить усиленную аутентификацию Пользователя. ПУ отвечает сообщением, которое содержит идентификатор ресурса согласия. Пользователь проходит

аутентификацию на стороне ПУ и авторизует согласие, чей идентификатор был получен на предыдущем шаге, что приводит к выдаче СПУ от ПУ токена доступа, связанного с авторизованным согласием.

7 Требования к параметрам взаимодействия

7.1 Кодировка символов

Запросы и ответы API используют кодировку UTF-8. Это кодировка символов по умолчанию для JSON (RFC 7158 - **раздел 8.1**).

Однако ПУ может не принимать некоторые символы UTF-8, такие как символы emoji («идеограммы и «смайлики» не могут быть частью приемлемой ссылкой на платеж»). Если ПУ отклоняет сообщение с символом UTF-8, которое не может быть обработано, то ПУ отвечает кодом состояния HTTP 400 (неверный запрос).

7.2 Формат даты и времени

Все даты и даты с указанием времени в полезных нагрузках JSON указываются в формате JSON схемы date-time с указанием временной зоны в виде смещения относительно UTC.

Пример:

2019-07-08T11:23:03+03:00

Все даты и даты с указанием времени в параметрах query указываются в формате date-time с указанием временной зоны в виде смещения относительно UTC.

Пример:

GET /transactions?fromBookingDateTime=2024-10-01T00:00:00+03:00&toBookingDateTime=2024-10-31T23:59:59+03:00

Все даты в заголовках HTTP представлены как полные даты **RFC 7231.**

Пример:

Mon, 26 Aug 2019 14:23:51 GMT

Все даты в параметрах claims JWT имеют формат number JSON, как целое число, представляющий количество секунд с 1970-01-01T0:0:0 Z UTC.

Пример:

//Mon, 26 Aug 2019 14:23:51 UTC

1987646700

7.3 Структура пути URI ресурса

Путь URI соответствует следующей структуре:

- \[participant-path-prefix\]/open-banking/\[version\]/\[resource-group\]/\[resource\]/\[resource-id\]/\[sub-resource\]

Структура URI пути состоит из следующих элементов:

- [participant-path-prefix]
  Необязательный префикс ПУ.
- open-banking
  Постоянное значение, дополняемое к префиксу, для определения единого URI для всех API. Имеет значение “open-banking”, если не переопределено на уровне прикладных стандартов.
- [version]
  Версия API, выраженная в виде /v[major-version].[minor-version]/.
- [resource-group]

Идентификатор группы конечных точек, в соответствии с ролью участника.

- [resource]/[resource-id]
  Наименование ресурса и его идентификатор.
- [sub-resource]
  Наименование подресурса (ресурса 2-го уровня).

ПУ использует один и тот же participant-path-prefix и host name для всех своих ресурсов.

Примеры:

<https://bank.ru/oapi-channel/open-banking/v2.0/pisp/payments>
<https://bank.ru/oapi-channel/open-banking/v1.3/aisp-pe/accounts>

7.4 HTTP заголовки в сообщениях (headers)

7.4.1 Заголовки в запросах

Настоящий стандарт определяет следующие стандартные и пользовательские заголовки в запросах HTTP.

Параметр header	Комментарий	POST	GET	DELETE	PUT
Accept	Стандартный HTTP заголовок, определяющий тип контента, который требуется от сервера. Если СПУ ожидает незашифрованный ответ, то он указывает явно, что только ответ в формате JSON принимается (передавая значение application/json) в	Опционально	Опционально	Не используется	Опционально

Параметр header	Комментарий	POST	GET	DELETE	PUT
	качестве заголовка контента для всех конечных точек, которые отвечают в формате JSON. Если СПУ ожидает зашифрованный ответ, то он указывает явно, что принимается только ответ JWT (передавая значение application/jose+jwe) в качестве заголовка контента для всех конечных точек, которые отвечают JSON. Для конечных точек, которые не отвечают в формате JSON (например, GET../statements/ {statementId}/file), ПУ указывает доступные параметры на своем портале для разработчиков. СПУ может предоставлять дополнительную информацию. Если установлено недопустимое значение, то ПУ отвечает: 406 (Not Acceptable). Если значение не указано, по умолчанию используется application/json.				

Параметр header	Комментарий	POST	GET	DELETE	PUT
Authorization	Стандартный заголовок HTTP. Позволяет предоставлять учетные данные серверу авторизации и/или серверу ресурсов в зависимости от типа запрашиваемого ресурса. Для OAuth 2.0 / OIDC включает в себя Basic или Bearer схемы аутентификации.	Обязательно	Обязательно	Обязательно	Обязательно
Content-Type	Стандартный заголовок HTTP. Представляет формат полезной нагрузки в запросе. Устанавливается значение application/json, за исключением конечных точек, которые поддерживают Content-Type, отличный от application/json (например, POST /file-payment-consents/{consentId}/file). Устанавливается значение application/jose+jwe для зашифрованных запросов. СПУ может предоставлять	Обязательно	Не используется	Не используется	Обязательно

Параметр header	Комментарий	POST	GET	DELETE	PUT
	дополнительную информацию. Если установлено другое значение, то ПУ присылает ответ: 415 Unsupported Media Type.				
If-Modified-Since	Фильтрация по времени модификации. Возвращает только те ресурсы, которые были изменены с указанного времени обращения.	Не используется	Опционально	Не используется	Не используется
If-None-Match	Фильтрация по идентификатору данных. Возвращает ресурс только в случае, если представленное значение Etag не совпадает с текущим.	Не используется	Опционально	Не используется	Не используется
x-fapi-auth-date	Время последнего входа Пользователя в приложение СПУ. Значение предоставляется в виде HTTP-date, как в разделе **7.1.1.1** [RFC 7231]. Например, x-fapi-auth-date: Mon, 26 Aug 2019 12:23:11 GMT	Опционально	Опционально	Опционально	Не используется
x-fapi-customer-ip-address	IP-адрес Пользователя, если Пользователь в данный момент подключен к СПУ (авторизован в	Опционально	Опционально	Опционально	Не используется

Параметр header	Комментарий	POST	GET	DELETE	PUT
	приложении СПУ). Если заголовок в запросе указан и содержит корректное значение, то предполагается, что Пользователь присутствует во время взаимодействия.				
x-fapi-interaction-id	**RFC 4122** UID, используемый в качестве идентификатора корреляции. Если необходимо, то ПУ передает обратно значение идентификатора корреляции в заголовке ответа x-fapi-interaction-id.	Обязательно	Обязательно	Обязательно	Обязательно
x-idempotency-key	Не стандартный HTTP заголовок. Уникальный идентификатор запроса для поддержки идемпотентности. Обязательно для запросов POST к конечным точкам идемпотентного ресурса. Для других запросов не указывается.	Опционально	Не используется	Не используется	Не используется
x-jws-signature	Указывает, что тело запроса подписано. В документации на ресурсы отдельно определяется, когда это поле в	Условно (зависит от API)	Не используется	Не используется	Обязательно

Параметр header	Комментарий	POST	GET	DELETE	PUT
	заголовке указывается.				
x-customer-user-agent	В заголовке указывается тип устройства (user-agent), который использует Пользователь. СПУ может заполнить это поле значением типа устройства (user-agent), указанным Пользователем. Если Пользователь использует мобильное приложение СПУ, СПУ проверяет, что строка типа устройства (user-agent) отличается от строки типа устройства (user-agent) на основе браузера.	Опционально	Опционально	Опционально	Опционально

7.4.2 Обработка заголовков запросов

Сервер ресурсов должен корректно обрабатывать указанные заголовки, а также все стандартные заголовки HTTP в запросах, для обеспечения полной поддержки взаимодействий с клиентами и обеспечения работы протокола HTTP.

Если сервер ресурсов получает не зарегистрированные пользовательские заголовки (заголовки с префиксом «x-»), то его поведение должно быть следующим:

1. Игнорирование: если заголовок не оказывает влияния на обработку запроса и не нарушает безопасность, сервер может игнорировать его и продолжить обработку запроса в штатном режиме.
2. Логирование: В целях аудита и отладки сервер может записывать информацию о пользовательских заголовках в лог-файлы, чтобы отслеживать потенциальные необычные или вредоносные запросы.
3. Ошибка или предупреждение: если пользовательский заголовок нарушает требования безопасности или противоречит стандартам обработки, сервер может вернуть ответ с кодом ошибки 400 Bad Request и пояснением, что заголовок недопустим.

4. Дополнительные проверки безопасности: если пользовательские заголовки потенциально могут нести риск (например, заголовки, изменяющие параметры авторизации или безопасности), сервер ресурсов может включать блокирование запросов с неизвестными заголовками, которые могут представлять угрозу (попытки взлома с использованием нестандартных методов).

Пример:

POST /api/secure/operation HTTP/1.1
Host: api.example.com
x-Override-Authorization: fakeToken123

7.4.3 Заголовки в ответах

Настоящий стандарт определяет следующие стандартные и пользовательские заголовки в ответах HTTP.

Параметр header	Комментарий	Обязательность
Content-Type	Стандартный параметр заголовка HTTP. Представляет формат полезной нагрузки, возвращаемой в ответе. ПУ возвращает значение Content-Type, равное application/json, в качестве заголовка для всех незашифрованных конечных точек. ПУ возвращает значение Content-Type, равное application/jwe, для всех зашифрованных конечных точек.	Обязательно
Retry-After	Параметр заголовка, указывающий время (в секундах), в течение которого СПУ ожидает перед повторением операции. ПУ следует включать этот заголовок вместе с ответами с кодом состояния HTTP 429 (Too Many Requests)	Опционально
x-fapi-interaction-id	**RFC 4122** UID, используемый в качестве идентификатора корреляции. ПУ заполняет параметр заголовка ответа x-fapi-interaction-id значением, полученным в соответствующем параметре заголовка запроса или значением UID **RFC 4122**, если значение не было предоставлено в запросе для отслеживания взаимодействия	Обязательно
x-jws-signature	Указывает, что тело ответа подписано. Конкретные условия применения заголовка описаны в документации на ресурсы.	Условно (зависит от API)
Etag	Уникальный идентификатор, который определяет, изменился ли ресурс	Опционально

7.5 Идентификатор изменения ресурса

ПУ может возвращать в заголовке HTTP уникальный идентификатор ресурса Etag, с помощью которого СПУ может определять, изменился ли ресурс с момента последнего запроса.

СПУ может использовать HTTP метод HEAD в запросе на доступ к ресурсу с заголовком HTTP If-None-Match, указав в нем значение идентификатора полученного ранее ресурса Etag. ПУ возвращает только те ресурсы, в которых представленное значение Etag не является актуальным.

СПУ может использовать HTTP метод HEAD в запросе на доступ к ресурсу с заголовком HTTP If-Modified-Since, указав в нем дату и время полученного ранее ресурса. ПУ возвращает только те ресурсы, которые были изменены с указанного времени.

7.6 Коды статусов HTTP

Ниже приведены коды ответов HTTP для различных методов HTTP, для всех конечных точек API. Параметры ответа и значения кодов должны соответствовать требованиям разделов 5.4.2.8.и 5.4.2.9 ФАПИ.СЕК.

Ситуация	Статус HTTP	Комментарий	POST	GET	DELETE	PUT
Запрос успешно выполнен	200 OK		Нет	Да	Нет	Да
Операция создания выполнена успешно	201 Created	Результатом операции является создание нового ресурса.	Да	Нет	Нет	Нет
Операция удаления успешно завершена	204 No Content		Нет	Нет	Да	Нет
Запрос имеет неверный формат, отсутствующее или несовместимое тело JSON, параметры URL или поля заголовка	400 Bad Request	Запрошенная операция не будет выполнена.	Да	Да	Да	Да
Заголовок авторизации отсутствует или неверный токен	401 Unauthorized	Операции было отказано в доступе. аутентификация Пользователя может привести к созданию соответствующего	Да	Да	Да	Да

		токена, который может быть использован.				
Токен имеет неверную область действия или была нарушена политика безопасности	403 Forbidden	Операции было отказано в доступе. Повторная аутентификация Пользователя может привести к созданию соответствующего токена, который может быть использован.	Да	Да	Да	Да
СПУ пытается получить ресурс, который не указан в спецификации и не реализован на стороне ППИ	404 Not Found		Да	Да	Да	Да
СПУ попытался получить доступ к ресурсу с помощью метода, который не поддерживается	405 Method Not Allowed		Да	Да	Да	да
Запрос содержал параметр заголовка Accept, отличный от разрешенных media types, и набор символов, отличный от UTF-8	406 Not Acceptable		Да	Да	Да	Да
Операция была отклонена, поскольку полезная нагрузка находится в формате, не поддерживаемом этим методом на целевом ресурсе	415 Unsupported Media Type		Да	Нет	Нет	Да
Операция была отклонена, так как слишком много запросов было	429 Too Many Requests	ПУ ограничивают запросы, если они сделаны сверх их политики	Да	Да	Да	Да

сделано в течение определенного периода времени		добросовестного использования. ПУ документируют свои политики добросовестного использования на своих порталах для разработчиков. ПУ отвечают этим статусом, если количество запросов в единицу времени было превышено. ПУ следует включать заголовок Retry-After в ответ, указывающий, как долго СПУ ожидает перед повторением операции.				
Что-то пошло не так на стороне ПУ	500 Internal Server Error	Операция не удалась	Да	Да	Да	Да
Сервис не реализован на стороне ПУ	501 Not Implemented	СПУ пытается получить ресурс, который указан в спецификации, но не реализован на стороне ПУ (например, ПУ решил не реализовывать конечную точку API статуса для внутренних запланированных платежей)	Нет	Да	Нет	Нет
Устаревшая версия сервиса	503 Service Unavailable	Если API устарел и больше не поддерживается ПУ, его путь URI все еще может быть активным и принимать запросы. В этом контексте рекомендуется вернуть 503 Service Unavailable, чтобы TPP знал, что версия	Да	Да	Да	Да

		API находится в офлайн режиме.				

ПУ может возвращать другие стандартные коды состояния HTTP (например, от шлюзов и других периферийных устройств), как описано в **RFC 7231 - Раздел 6**.

ПУ отвечают на некорректные прикладные запросы общей структурой ошибок Открытых программных интерфейсов, если ошибка предусматривает передачу полезной нагрузки в ответе.

7.6.1 400 (Bad Request) или 404 (Not Found)

Если СПУ пытается запросить URL ресурса с идентификатором ресурса, который не существует, то ПУ отвечает 400 (неверный запрос), а не 404 (не найдено).

Например, если СПУ пытается выполнить запрос GET/payment/22289, где 22289 не является действительным paymentId, ПУ отвечает 400.

Если СПУ пытается получить доступ к URL-адресу ресурса, который не определен этими спецификациями (например, GET /bulk), то ПУ отвечает 404 (Not Found).

Если ПУ не реализовал конечную точку API, то он отвечает 501 (Not Implemented) для запросов к этому URL.

Таблица ниже иллюстрирует некоторые примеры предсказуемого поведения:

Ситуация	Запрос	Ответ
СПУ пытается получить платеж с несуществующим идентификатором paymentId	GET /payments/1001	400 (Bad Request)
СПУ пытается получить ресурс, который указан в спецификации, но не реализован на стороне ПУ. Например, ПУ решил не реализовывать конечную точку API для получения транзакций по счету	GET /accounts/{accountId}/transactions	404 (Not Found)
СПУ пытается получить ресурс, который не определен	GET /bulk	404 (Not Found)

7.6.2 403 (Forbidden)

Когда СПУ пытается получить доступ к ресурсу, к которому у него нет разрешения, ПУ возвращает 403 (Forbidden) с необязательным указанием дополнительной информации об ошибке в теле ответа на запрос.

Ситуация возможна в следующих случаях:

- СПУ использует токен доступа (access token), который не имеет соответствующей области действия (scope) для доступа к запрошенному ресурсу.
- СПУ попытался получить доступ к ресурсу с идентификатором, к которому у него нет доступа. Например, попытка получить доступ к GET /payments/1001, где платежный ресурс с идентификатором 1001 принадлежит другому СПУ.
- СПУ пытается получить доступ к ресурсу транзакции, а у СПУ нет согласия на авторизацию с правами доступа к запрашиваемому ресурсу.
- СПУ пытается получить доступ к ресурсу транзакции, а у СПУ нет согласия на авторизацию для accountId. Например, попытка получить доступ к GET /accounts/2001 или GET /accounts/2001/transactions, когда Пользователь не выбрал accountId 2001 для авторизации.
- СПУ пытается получить доступ к ресурсу, а ПУ решает повторно аутентифицировать Пользователя. ПУ отвечает соответствующим кодом ошибки, чтобы указать, что требуется повторная аутентификация.

7.6.3 401 (Unauthorized)

Когда СПУ использует токен доступа с истекшим сроком, ПУ возвращает 401 (Unauthorized) без какого-либо сообщения об ошибке.

Ситуация возникает, если ПУ завершил срок действия токена доступа по любой из следующих причин:

- Истек срок действия согласия.
- Подозрительное использование токена доступа или подозрение в мошенничестве.
- Плановая реализация усиленной аутентификации.

Эта ошибка потенциально может быть исправлена, если Пользователь повторно пройдет аутентификацию или аутентифицируется с использованием усиленной аутентификации.

7.6.4 429 (Too Many Requests)

Если СПУ пытается получить доступ к ресурсу слишком часто, то ПУ может вернуть 429 (Too Many Requests). Это нефункциональное требование, и отдельные ПУ определяют метрику запросов в единицу времени.

Ситуация возникает, когда:

- СПУ решает реализовать функцию “Статус платежа в реальном времени” для своих Пользователей и делает это некорректно, опрашивая конечную точку методом GET.
- СПУ решает использовать конечную точку для разового единичного платежа, как если бы она была конечной точкой пакетной оплаты, и отправляет большое количество запросов на оплату в очень короткий промежуток времени, так что это превышает политику использования ПУ.

7.7 Идемпотентность

Ключ идемпотентности используется для защиты от создания дубликатов ресурсов при использовании метода POST для конечных точек API.

Если для конечной точки API требуется ключ идемпотентности:

- Параметр заголовка x-idempotency-key содержит не более 40 символов. Если длина поля превышает 40 символов, то ПУ отклоняет запрос с кодом состояния 400 (Bad Request).
- СПУ не меняет тело запроса при использовании одинакового ключа x-idempotency-key. Если СПУ изменяет тело запроса, то ПУ не меняет конечный ресурс. ПУ рассматривает это как мошенническое действие.
- ПУ обрабатывает запрос как идемпотентный, если он получил запрос с существующим параметром x-idempotency-key от того же СПУ в течение 24 часов.
- ПУ не создает новый ресурс для запроса POST, если он определен как идемпотентный запрос.
- ПУ отвечает на запрос текущим статусом ресурса (или статусом, максимально близким к текущему, который можно получить в данный момент времени на существующем онлайн канале) и кодом статуса HTTP 201 (Created).
- СПУ не использует ключ идемпотентности при опросе состояния ресурсов.
- ПУ использует подпись сообщения вместе с ключом идемпотентности, чтобы гарантировать, что тело запроса не изменилось.

Если ключ идемпотентности не требуется для конечной точки API, но содержится в запросе, то ПУ игнорирует ключ идемпотентности.

7.8 Подписание сообщений

Для подписания сообщений в среде Открытых программных интерфейсов используется цифровая подпись в формате JSON, как определено в разделе 5.7.1 ФАПИ.СЕК. Обязательность подписания полезной нагрузки сообщения документируется в спецификациях API для прикладных стандартов.

7.9 Фильтрация

ПУ обеспечивает ограниченную поддержку фильтрации для операций GET, которые возвращают множественные записи.

Параметры фильтра всегда разные для конкретного поля (полей) ресурса и следуют правилам/форматам, определенным в справочниках для ресурса.

7.10 Нумерация страниц

ПУ предоставляет постраничный ответ для операций GET, которые возвращают множественные записи.

В такой ситуации ПУ:

- Если существует следующая страница записей ресурсов, то ПУ предоставляет ссылку на следующую страницу ресурсов в поле Links.next ответа. Отсутствие следующей ссылки будет означать, что текущая страница является последней страницей результатов.

- Если предыдущая страница записей ресурсов существует, то ПУ предоставляет ссылку на предыдущую страницу ресурсов в поле Links.prev ответа. Отсутствие предыдущей ссылки указывает на то, что текущая страница является первой страницей результатов.

Для разбитых на страницы ответов ПУ гарантирует, что количество записей на странице находится в разумных пределах и максимальное значение ограничивается размером массива возвращаемого ресурса.

Дополнительно ПУ предоставляет:

- Ссылку на первую страницу результатов в поле Links.first.
- Ссылку на последнюю страницу результатов в поле Links.last.
- Общее количество страниц в поле Meta.totalPages.

ПУ включает «self» ссылку на ресурс в поле Links.self, как описано в разделе «Ссылки».

Этот стандарт не определяет, каким образом параметры перелистывания страниц передаются ПУ, каждый ПУ может использовать свои собственные механизмы для разбивки ответа.

Если исходный запрос от СПУ включал параметры фильтра, то в ответе возвращаются только те результаты (разбитые на страницы), которые соответствуют фильтру.

7.11 Архивирование

Архивация ресурсов, которые были удалены или у них истек срок действия определятся ПУ на основе их внутренних требований.

8 Общая структура полезной нагрузки

В этом разделе приводится обзор структуры верхнего уровня для полезных нагрузок Открытых программных интерфейсов.

Данные, которые содержатся в разделе «Data», документируются для каждой отдельно взятой конечной точки API.

8.1 Соглашение о наименованиях

Простые элементы (атрибуты, поля данных) должны следовать соглашению о наименовании lowerCamelCase:

- Первая буква первого слова всегда в нижнем регистре.
- Все последующие слова должны начинаться с заглавной буквы.
- Пробелы, знаки препинания и спецсимволы не допускаются между словами.

Пример: areaCode, accountNumber.

Объекты и структуры данных должны использовать соглашение CamelCase:

- Первая буква каждого слова, включая первое, должна быть заглавной.
- Пробелы, знаки препинания и спецсимволы не допускаются между словами.

Пример: AccountDetails, TransactionList.

8.2 Идентификатор модели данных ресурса

Идентификатор модели данных ресурса REST API должен соответствовать следующим требованиям.

Уникальность: Идентификатор ресурса должен быть уникальным в контексте сервиса Поставщика услуг. Это означает, что никакие два ресурса не могут иметь одинаковый идентификатор.

Неизменяемость: Идентификатор ресурса должен быть неизменным. Это означает, что он не должен изменяться в течение жизненного цикла ресурса.

Согласованность: Идентификатор ресурса должен быть согласованным в формате и стиле.

Машиночитаемость: Идентификатор ресурса должен быть машинно-читаемым, то есть его должно быть легко обрабатывать и понимать компьютеру.

URL-дружелюбность: Идентификатор ресурса должен быть дружелюбным для URL. Это означает, что он не должен содержать никаких символов, которые не допускаются в URL, таких как пробелы, слеши или двоеточия. Это облегчает включение идентификатора в URL, не прибегая к кодированию или экранированию специальных символов.

8.3 Структура запроса

Структура верхнего уровня для запросов Открытых программных интерфейсов имеет следующий вид:

```
{
   "Data": {
      ...
   },
   "Risk": {
      ...
   }
}
```

Data

Раздел «Data» – (обязательный для каждого стандарта), раздел полезной нагрузки, содержащий прикладные данные.

Структура этого элемента отличается для каждой конечной точки API.

Risk

Раздел «Risk» – (опциональный) содержит индикаторы риска для конкретного запроса API, предоставленного ССПУ. Требования к наличию раздела в сообщениях определяется для каждого прикладного стандарта.

Индикаторы риска, содержащиеся в этом элементе, могут отличаться для каждой конечной точки API.

8.4 Структура ответа

В соответствии с принципом API RESTful полный ресурс воспроизводится как часть ответа.

В ответ включаются следующие дополнительные разделы высокого уровня:

- Links
- Meta

Структура верхнего уровня для ответов Открытых программных интерфейсов имеет следующий вид:

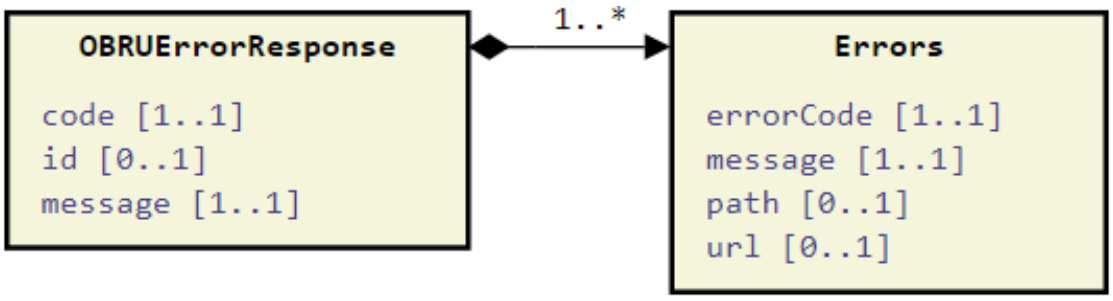
```
{
   "Data": {
     ...
   },
   "Risk": {
     ...
   },
   "Links": {
     ...
   },
   "Meta": {
     ...
   }
}
```

8.5 Структура ответа с ошибками

Структура для ответов с ошибками Открытых программных интерфейсов имеет следующий вид:

```
{
   "code": "...",
   "id": "...",
   "message": "...",
   "Errors": [
     {
       "errorCode": "...",
       "message": "...",
       "path": "...",
       "url": "..."
     }
   ]
}
```

UML -диаграмма для структуры ответов с ошибками:



Наименование	Тип	Описание	Кратность
code	Max40Text	Высокоуровневый текстовый код ошибки, необходимый для классификации	1..1
id	Max40Text	Уникальный идентификатор ошибки, для целей аудита, в случае неизвестных / не классифицированных ошибок	0..1
message	Max500Text	Краткое сообщение об ошибке. Например, «что-то не так с предоставленными параметрами запроса»	1..1
Errors	OBRUError	Список кодов ошибок	1..n

Состав данных объекта OBRUErrorResponse:

Наименование	Тип	Описание	Кратность
errorCode	OBRUErrorResponseErrorCode	Низкоуровневое текстовое описание ошибки в виде кода. Например, RU.SBRF.Field.Missing.	1..1
message	Max500Text	Описание ошибки. Например, ‘Обязательное поле не указано’.	1..1
path	Max500Text	Путь к элементу с ошибкой в JSON объекте. Рекомендуемое, но не обязательное поле.	0..1
url	URI	URL для помощи в устранении проблемы, также через URL можно предоставлять дополнительную информацию.	0..1

8.6 Необязательные поля

В объектах, где значение для необязательного поля не указано, поле исключается из полезной нагрузки JSON.

В объектах, где поле массива определено как имеющее значения кратности 0..n, поле массива включается в полезную нагрузку с пустым массивом. Примеры передачи необязательных полей:

```
{

   "name": "", //Неправильно. Поле "name" нужно исключить из полезной нагрузки.
   "age": 0, // Неправильно. Значение "0" не используется для указания неопределенного значе
ния.
   "creditorAccount": {}, // Неправильно. Поле "creditorAccount" нужно исключить.
   "Balance": [] // Правильно. Таким образом передается пустой массив.
}
```

8.7 Ссылки

Раздел «Links» является опциональным и должен содержать элементы, указывающие на абсолютные URI для связанных ресурсов.

Елемент «self» (ссылка на текущую страницу) является обязательной, если раздел «Links» присутствуется.

Пример передачи одинарной ссылки «self»:

```
"Links": {
   "self": "<https://api.bank.ru/open-banking/v3.1/payments/58923>"
}
```

При передаче большого количества данных раздел «Links» может также содержать элементы «first», «prev», «next» и «last».

Пример передачи всех элементов раздела «Links»:

```
"Links": {
   "self": "<http://rocks.ru/articles?page\[number\]=3&page\[size\]=25>",
   "first": "<http://rocks.ru/articles?page\[number\]=1&page\[size\]=25>",
   "prev": "<http://rocks.ru/articles?page\[number\]=2&page\[size\]=25>",
   "next": "<http://rocks.ru/articles?page\[number\]=4&page\[size\]=25>",
   "last": "<http://rocks.ru/articles?page\[number\]=6&page\[size\]=25>"
}
```

8.8 Метаданные

Раздел «Meta» обязателен, но может быть пустым. Необязательный элемент — «totalPages», указывает на количество передаваемых страниц. Если передается более одной страницы, то элемент «totalPages» обязательно присутствует.

Пример передачи раздела «Meta»:

```
"Meta": {
   "totalPages": 6
}
```

8.9 Пример использования

Пример структуры полезной нагрузки для многостраничного ответа ПУ:

HTTP/1.1 200 OK
x-fapi-interaction-id: 11bac543-d5de-3446-b687-880a5018434d
Content-Type: application/json

```json { “Data”: { … }, “Links”: { “self”: “https://bank.ru/open-banking/v2.0/aisp/accounts/98765/transactions/”, “last”: “https://bank.ru/open-banking/v2.0/aisp/accounts/98765/transactions?pg=6”, “first”: “https://bank.ru/open-banking/v2.0/aisp/accounts/98765/transactions/”, “next”: “https://bank.ru/open-banking/v2.0/aisp/accounts/98765/transactions?pg=2” }, “Meta”: { “totalPages”: 6, “firstAvailableDateTime”: “2019-05-03T00:00:00+00:00”, “lastAvailableDateTime”: “2019-12-03T00:00:00+00:00” } }
```