ЦЕНТРАЛЬНЫЙ БАНК РОССИЙСКОЙ ФЕДЕРАЦИИ

(БАНК РОССИИ)

УТВЕРЖДЕН
ЦБРФ.62.0.39683.ТЗ.8

ЕДИНАЯ ПЛАТФОРМА ВНЕШНЕГО ВЗАИМОДЕЙСТВИЯ ВНЕШНИЙ ПОРТАЛ (ПОРТАЛ "БИВРЁСТ")

Технические условия внешнего обмена Часть 65

ЦБРФ.62.0.39683.ТУ-65

На 88 листах

2022

Настоящие Технические условия разработаны в рамках реализации проекта «Единая платформа внешнего взаимодействия (Внешний портал Банка России)» (ЕПВВ).

Данный документ входит в состав технорабочей документации на Портал «Биврёст».

В настоящем документе представлено описание взаимодействия Портала «Биврёст» с внешними системами УИО с использованием REST-сервисов.

Содержание

Обозначения и сокращения

Сокращение	**Определение**
API	Application Programming Interface, интерфейс прикладного программирования, набор методов взаимодействия между приложениями
HTTP	Hypertext Transfer Protocol, Гипертекстовый протокол передачи, описывается документом RFC2616
HTTPS	HTTP over TLS, Защищенная версия протокола HTTP с использованием SSL3/TLS, описывается документом RFC2818
JSON	JavaScript Object Notation, структурированный текстовый формат обмена данными, на основе синтексиса объектов JavaScript
REST	Representational State Transfer, «передача репрезентативного состояния», архитектурный стиль взаимодействия компонентов распределённых приложений в сети
XML	eXtensible Markup Language, расширяемый язык разметки
ГОСТ	Государственный стандарт
ЕПВВ	Единая система обмена данными с внешними абонентами
ИОД	Информация ограниченного доступа
ЛК	Личный кабинет
СОС	Список отозванных сертификатов
УИО	Участник информационного обмена
УЦ	Удостоверяющий центр
ЭП	Электронная подпись
ЭС	Электронное сообщение

1 Общие положения

1.1 Наименование документа

Полное наименование автоматизированной системы – «Внешний портал Единой платформы внешнего взаимодействия. Описание внешнего взаимодействия».

Условное обозначение – Портал «Биврёст».

1.2 Цель документа

Настоящие технические условия разработаны в целях описания технических аспектов информационного взаимодействия с внешними системами.

1.3 Сфера применения

Настоящие технические условия применяются для реализации информационного обмена между внешними системами и Порталом "Биврёст".

В рамках информационного обмена между участниками могут передаваться следующие типы информации:

- открытая информация, предназначенная для официальной передачи во внешние организации, средства массовой информации и т.п.;
- внутрибанковская информация, предназначенная для использования исключительно сотрудниками Банка России при выполнении ими своих служебных обязанностей;
- информация ограниченного доступа;
- персональные данные.

Не допускается передача информации, включающей ИОД и персональные данные в открытом виде. Шифрование соответствующей информации должно осуществляться абонентскими средствами криптозащиты на основе сертифицированных программных комплексов, использующих криптографические алгоритмы ГОСТ.

2 Общее описание системы

Архитектура портала "Биврёст" состоит из технологических подсистем (ТПС), которые реализуют функции, необходимые для обеспечения работы потоков взаимодействия, которые в свою очередь, реализуют функции прикладного характера. Также в рамках технологических подсистем реализуются сервисы, обеспечивающие выполнение технологических операций.

В настоящем документе представлено описание взаимодействия Портала «Биврёст» с внешними системами УИО с использованием REST-сервисов (REST API).

3 Описание внешнего взаимодействия

3.1 Взаимодействие с использованием ПП Универсальный REST-сервис приема отчетности

3.1.1 Сценарии использования универсального REST-сервиса

3.1.1.1 Определение задачи для отправки:

Для определения списка доступных задач можно использовать метод, описанный в разделе 3.1.6. Из указанного списка необходимо выбрать задачу на основе:

- анализа описания;
- требований Банка России;
- руководства Пользователя ЕПВВ.

В случае, если необходимая задача отсутствует в списке, рекомендуются 2 действия:

- Вызвать метод из раздела 3.1.6.2, убедиться, что в списке видов деятельностей есть необходимый (тот, для которого реализуется взаимодействие с Банком России);
- Зайти в ЛК УИО через web-интерфейс и проверить наличие задачи в разделе «Представление отчётности».

3.1.1.2 Отправка сообщения с отправкой файлов через HTTP

Любая отправка сообщения происходит из 4 этапов:

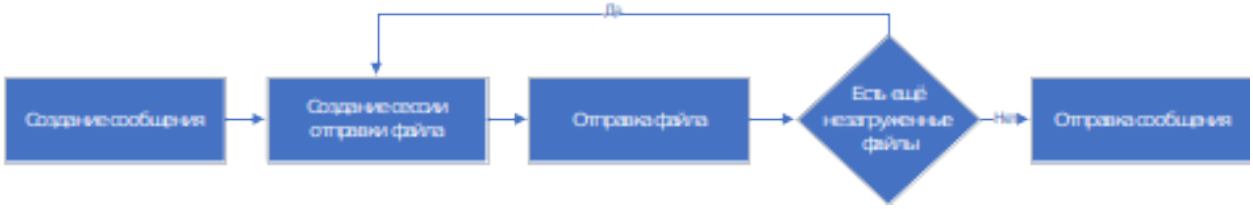

Рисунок 1 - Порядок отправки сообщения

- На этапе создания сообщения вызывается метод POST /messages (3.1.3.1), в котором указывается базовая информация о сообщении и список файлов (не сами файлы). В ответ универсальный REST создаёт новое сообщение (в web-интерфейсе оно отображается как «Черновик»), возвращает зарезервированные ID для сообщения и для всех файлов;

- На этапе создания сессии отправки файла вызывается метод POST /messages/{msgId}/files/{fileId}/createUploadSession (3.1.3.2), где msgId и fileId – сгенерированы универсальным REST и переданы клиенту на предыдущем этапе. В ответ универсальный REST возвращает полную ссылку, по которой необходимо осуществлять отправку;
- На этапе отправки клиент методом PUT /messages/{msgId}/files/{fileId} (3.1.3.3) отправляет части (чанки) сообщений. Передаваемый диапазон байт регулируется самим клиентом (позволяет передать одной частью или раздробить на несколько частей);
- После того как все файлы сообщения успешно загружены на сервер, клиент вызывает метод POST /messages/{msgId} (3.1.3.4), которым финализирует отправку. Сообщение переходит в статус «Отправлено» и передаётся на обработку в Банк России.

3.1.1.3 Отправка сообщения с отправкой файлов сообщения через FASP (ТПС ASPERA)

Отправка файлов через FASP (ТПС ASPERA) возможна только для задач, у которых установлен признак AllowAspera (для получения справочника задач см. п.п. 3.1.6.1). При этом, отправка сообщения состоит из следующих этапов:

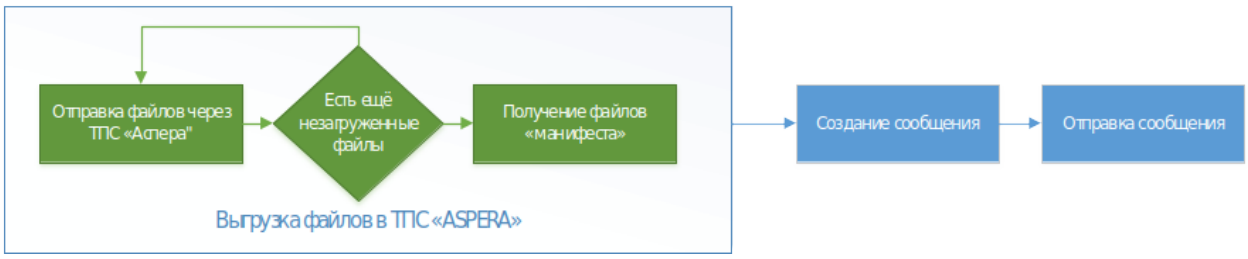

Рисунок 2 - Порядок отправки сообщения. Этапы.

- Клиент загружает файлы (имена файлов должны быть в виде GUID), используя ТПС ASPERA, и получает на них файлы «манифеста», подтверждающие загрузку;
- На этапе создания сообщения вызывается метод POST /messages (3.1.3.1), в котором указывается базовая информация о сообщении и список файлов (не сами файлы). В ответ универсальный REST создаёт новое сообщение (в web-

интерфейсе оно отображается как «Черновик»), возвращает зарезервированные ID для сообщения и для всех файлов. При этом пользователь должен обязательно указать, что репозиторием файлов сообщения будет Aspera, а также контрольную сумму, алгоритм ее формирования и путь к файлу, включая его имя, полученные из файла «манифеста» ТПС «Aspera»;

- В случае получения успешного ответа, клиент вызывает метод POST /messages/{msgId} (3.1.3.4), которым финализирует отправку. Сообщение переходит в статус «Отправлено» и передаётся на обработку в Банк России.

3.1.1.4 Поиск сообщения

Для получения информации о сообщении используется метод GET /messages (3.1.4). Есть возможность передачи в качестве параметров следующей фильтрующей информации:

- Фильтр по дате сообщения (с какой даты времени и по какую дату и время было отправлено сообщение);
- Фильтр по размеру сообщения (это может потребоваться для удаления «самых больших сообщений» для очистки квоты);
- Фильтр по типу сообщения (входящие, исходящие, новые, отвеченные).

Так же возможна реализация с поиском сообщения на стороне клиента. В этом случае по очереди с сервера запрашиваются все сообщения (без указания дополнительных фильтров), которые сохраняются у клиента дальнейший поиск информации осуществляется уже на локальном сервере/АРМе. Этот способ рекомендуется для клиентов, имеющих значительный объём взаимодействия с Банком России (получение/передача более 30 (значение уточняется) сообщений в день), чтобы предотвратить излишнюю загрузку серверов ЕПВВ обработкой и фильтрацией запросов на стороне сервера.

3.1.1.5 Получение актуального статуса сообщения

После того, как нужное сообщение было найдено, статус сообщения можно определить двумя способами:

1) Вызвать метод GET /messages/msgId (3.1.4.2), определять по значению поля Status самого сообщения.

Статус	**Статус в ЛК**	**Описание**
draft	Черновик	Сообщение с данным статусом создано, но ещё не отправлено.
sent	Отправлено	Сообщение получено сервером

Статус	Статус в ЛК	Описание
delivered	Загружено	Сообщение прошло первоначальную проверку
error	Ошибка	При обработке сообщения возникла ошибка
processing	Принято в обработку	Сообщение передано во внутреннюю систему Банка России
registered	Зарегистрировано	Сообщение зарегистрировано
rejected	Отклонено	Сообщение успешно дошло до получателя, но было отклонено
new	Новое	Только для входящих сообщений. Сообщение в данном статусе ещё не прочтено Пользователем УИО.
read	Прочитано	Только для входящих сообщений. Сообщение в данном статусе прочтено Пользователем УИО.
replied	Отправлен ответ	Только для входящих сообщений. На сообщение в данном статусе направлен ответ.
success	Доставлено	Сообщение успешно размещено в ЛК/Сообщение передано роутером во внутреннюю систему Банка России, от которой не ожидается ответ о регистрации

2) Вызвать метод GET /messages/msgId (3.1.4.2), определять по наличию/отсутствию нужных квитанций в блоке Receipts. Тип квитанции определять по значению поле Status внутри массива значений Receipts. Возможные значения поля status:

Статус	Статус в ЛК	Описание
sent	Отправлено/Отправлен ответ	Сообщение получено сервером
delivered	Загружено	Сообщение прошло первоначальную проверку В рамках 5361-У, квитанцией о загрузке считается эта квитанция.
error	Ошибка	При обработке сообщения возникла ошибка
processing	Принято в обработку	Сообщение передано во внутреннюю систему ЦБ
registered	Зарегистрировано	Сообщение зарегистрировано В рамках 5361-У, квитанцией о регистрации считается эта квитанция.
rejected	Отклонено	Сообщение успешно дошло до получателя, но было отклонено

Статус	Статус в ЛК	Описание
new	Новое	Только для входящих сообщений. Сообщение в данном статусе ещё не прочтено Пользователем УИО.
read	Прочитано	Только для входящих сообщений. Сообщение в данном статусе прочтено Пользователем УИО.
replied	Отправлен ответ	Только для входящих сообщений. На сообщение в данном статусе направлен ответ.
success	Доставлено	Сообщение успешно размещено в ЛК/Сообщение передано роутером во внутреннюю систему Банка России, от которой не ожидается ответ о регистрации

3.1.1.6 Обработка ошибок при работе с ПП Универсальный REST-сервис

В случае ошибок REST методов из класса HTTP 4XX, в теле ответа передается JSON-объект с описанием ошибки вида:

BODY

```
{
  "HTTPStatus": "integer",
  "ErrorCode": "string",
  "ErrorMessage": "string",
  "MoreInfo": "object"
}
```

Где:

HTTPStatus – HTTP статус класса 4xx согласно Hypertext Transfer Protocol (HTTP) Status Code Registry [7].

ErrorCode – внутренний код ошибки Портала. Служит клиенту для автоматизированной обработки ошибок;

ErrorMessage – расшифровка ошибки. Служит для человеко-читаемой обработки ошибок;

MoreInfo – объект с дополнительно информацией к ошибке, по-умолчанию пустой.

Подробный список возможных ошибок ПП Универсальный REST приведен в Таблица 2 п. 7

3.1.2 Техническое описание взаимодействия с универсальным REST-сервисом

3.1.2.1 Параметры подключения к внешнему порталу с использованием универсального REST-сервиса

Для подключения к Порталу "Биврёст" посредством универсального REST-сервиса необходимо в Клиентском ПО отправителя указать следующую ссылку (URL):

https://portal5.cbr.ru/back/rapi2/*

Вместо звездочки необходимо указать соответствующе методы (3.1.3-3.1.6).

3.1.2.2 Внешний портал одновременно с текущей версией API REST-сервиса поддерживает одну ранее реализованную (предыдущую) версию API. При этом:

- по ссылке https://portal5.cbr.ru/back/rapi2/* или по ссылке https://portal5.cbr.ru/back/rapi2/v1/* доступна доступна предыдущая версия;
- по ссылке https://portal5.cbr.ru/back/rapi2/v2/* доступна текущая версия.

3.1.2.3 Формат пакета с сообщением

Формат отправляемого пакета зависит от условий взаимодействия с Банком России и определяется отдельно для каждого потока.

3.1.2.4 Авторизация

Для осуществления информационного взаимодействия прежде всего необходимо пройти активацию Пользователя в ЛК ЕПВВ. После прохождения активации и изменении логина и пароля, их можно использовать для авторизации в универсальный REST. Авторизация осуществляется с помощью передачи HTTP заголовка «Authorization». Тип авторизации – «Basic».

В качестве логина и пароля передаются учетные записи, созданные во время активации ЛК.

3.1.2.5 Общие правила оформления сообщений, передаваемых по протоколу HTTP 1.1

СЛЕДУЕТ задавать абсолютный URL в параметрах методов POST и GET.

Поле Accept СЛЕДУЕТ заполнять значениями «application/json».

Поле User-Agent СЛЕДУЕТ заполнять строковым значением, идентифицирующим ПО, которое используется для взаимодействия с Внешним порталом.

Слово «СЛЕДУЕТ» используется для указания того, что данное требование спецификации должно быть обеспечено, если этому не препятствуют серьезные причины.

Данное правило соответствует требованию к реализации протокола, определенное в RFC 2119.

3.1.3 Отправка сообщений

Отправка сообщений на стороне ЕПВВ осуществляется с использованием универсального REST-сервиса.

Для сообщений потоков, по задачам которых установлен флаг AllowAspera, инициативным действием является отправка файлов с помощью протокола FASP, предварительно создавать сообщение через универсальный REST запрещено.

3.1.3.1 Для создания нового сообщения используется метод POST

POST: */messages

В котором передается следующие параметры:

REQUEST

HEADER

```
{
     "Content-Type": "string",
}
```

Где:

Content-Type – тип передаваемого контента (application/json);

BODY

```
{
     "Task": "string",
     "CorrelationId": "string($uuid)",
     "GroupId": "string($uuid)",
     "Title": "string",
     "Text": "string",
     "Files": [
     {
          "Name": "string",
          "Description": "string",
          "FileType": "string",
          "Encrypted": "boolean",
          "SignedFile": "string",
          "Size": "integer",
          "RepositoryType": "string",
          "RepositoryInfo": {
               "CheckSum": "string",
               "CheckSumType": "string",
```

```
                "Path": "string"
            }
        }],
    "Receivers": [
    {
         "Inn": "string",
         "Ogrn": "string",
         "Bik": "string",
         "Email": "string",
         "RegNum": "string",
         "DivisionCode": "string",
         "Activity": "string"
    }
   ]
}
```

Где:

а) Task – код задачи (по справочнику задач в формате «Zadacha_*", где Zadacha_ - неизменная часть, * - число/набор символов определяющий порядковый номер/обозначение задачи), используется для идентификации задачи;

б) CorrelationId - идентификатор корреляции сообщения в формате UUID [16] (необязательно, указывается для формирования ответного сообщения для потоков, поддерживаемых данную функциональность);

в) GroupId –идентификатор группы сообщений в формате UUID [16] (необязательно, указывается для передачи информации о том, что сообщение является частью группы сообщений для потоков, поддерживаемых данную функциональность);

г) Title – название сообщения, отображается в интерфейсе;

д) Text – текст сообщения, отображается в интерфейсе;

е) Files – файлы включенные в сообщение:

1) Name – имя файла;

2) Description – описание файла (необязательное поле, для запросов и предписаний из Банка России содержит имя файла с расширением, однако может содержать запрещённые символы Windows);

3) FileType – указывается тип файла. Допустимы следующие типы файлов:

- Document – любые данные, которые не проходят логический контроль непосредственно на ВП ЕПВВ (файлы документов, любые архивы, в т.ч. зашифрованные, неструктурированные данные и другие файлы);
- SerializedWebForm - xml файл определенной структуры, который может быть проверен ВП ЕПВВ на соответствие его схеме;

– Sign – файл УКЭП, проверка которой влияет на прием/отбраковку сообщения, применяется для основной подписи сообщения и подписи машиночитаемой доверенности;
– PowerOfAttorney – файл машиночитаемой доверенности.

4) Encrypted – признак зашифрованности файла;

5) SignedFile – имя и расширение другого приложенного файла сообщения, подписью для которого является данный файл (заполняется только для файлов подписи *.sig);

6) Size – размер отправляемого файла в байтах. Имеет формат int64 (т.е. signed 64 bits);

7) RepositoryType – указывается “http” или “aspera”. Необязательный параметр, указывающий тип репозитория, в который пользователь будет загружать файл. В случае если не установлен, то зависит от характеристик задачи (см. п. 3.1.6.1, параметр AllowAspera);

8) RepositoryInfo – информация о характеристиках репозитория, в который будет загружен файл. Заполняется в случае, если указан RepositoryType = aspera. Содержит следующие поля:

– CheckSum – контрольная сумма файла, необходимая для контроля его целостности. Берется пользователем из «манифеста», формируемого ТПС «Aspera» после загрузки файла;
– CheckSumType – алгоритм расчёта контрольной суммы файла, в зависимости от установок ТПС «Aspera». Берется пользователем из «манифеста», формируемого ТПС «Aspera» после загрузки файла;
– Path – путь к файлу относительно хранилища пользователя в ТПС «Аспера», включая имя файла. Имена файлов должны быть в виде GUID без расширения. Имя генерирует сам пользователь. Берется пользователем из «манифеста», формируемого ТПС «Aspera» после загрузки файла.

ж) Receivers – получатели сообщения (необязательно, указывается для потоков адресной рассылки);

1) Inn – индивидуальный номер налогоплательщика получателя;

2) Ogrn – основной государственный регистрационный номер получателя;

3) Bik – банковский идентификационный код получателя;

4) Email – адрес электронной почты получателя;

5) RegNum – регистрационный номер КО – получателя по КГРКО;

6) DivisionCode – номер филиала КО – получателя по КГРКО;

7) Activity – краткое наименование вида деятельности.

RESPONSE

HTTP 200 – Ok

```
[
{
          "Id": "string($uuid)",
          "CorrelationId": "string($uuid)",
          "GroupId": "string($uuid)",
          "Type": "string",
          "Title": "string",
          "Text": "string",
          "CreationDate": "string",
          "UpdatedDate": "string",
          "Status": "string",
          "TaskName": "string",
          "RegNumber": "string",
          "TotalSize": "integer",
          "Sender":{
               "Inn": "string",
               "Ogrn": "string",
               "Bik": "string",
               "RegNum": "string",
               "DivisionCode": "string"
          },
          "Files": [
          {
               "Id": "string($uuid)",
               "Name": "string",
               "Description": "string",
               "FileType": "string",
               "Encrypted": "boolean",
               "SignedFile": "string($uuid)",
               "Size": "integer",
               "RepositoryInfo": [
               {
                    "Path": "string",
                    "Host": "string",
                    "Port": "integer",
                    "CheckSum": "string",
                    "CheckSumType": "string",
                    "RepositoryType": "string"
               }
               ]
          }
```

```
          ],
          "Receipts": [
          {
               "ReceiveTime": "string",
               "StatusTime": "string",
               "Status": "string",
               "Message": "string",
               "Files": [
               {
                    "Id": "string($uuid)",
                    "Name": "string",
                    "Description": "string",
                    "FileType": "string",
                    "Encrypted": "boolean",
                    "SignedFile": "string($uuid)",
                    "Size": "integer",
                    "RepositoryInfo": [
                    {
                         "Path": "string",
                         "Host": "string",
                         "Port": "integer",
                         "RepositoryType": "string"
                    }
                    ]
               }
               ]
          }
          ]
}
]
```

Где:

а) Id – уникальный идентификатор сообщения в формате UUID [16];

б) CorrelationId - идентификатор корреляции сообщения в формате UUID [16];

в) GroupId – идентификатор группы сообщений в формате UUID [16];

г) Type – тип сообщения исходящее (значение: outbox) или входящее (значение: inbox);

д) Title – название сообщения;

е) Text – текст сообщения;

ж) CreationDate – дата создания сообщения (ГОСТ ISO 8601-2001 по маске «yyyy-MM-dd’T’HH:mm:ss’Z’»);

з) UpdatedDate – дата последнего изменения статуса сообщения (ГОСТ ISO 8601-2001 по маске «yyyy-MM-dd’T’HH:mm:ss’Z’»);

и) Status – статус сообщения (возможные значения и их описание находится в п.3.1.1.5);

к) TaskName – наименование задачи;

л) RegNumber – регистрационный номер;

м) TotalSize – общий размер сообщения в байтах. Имеет формат int64 (т.е. signed 64 bits);

н) Sender – информация об отправителе:

1) Inn – индивидуальный номер налогоплательщика отправителя;

2) Ogrn – основной государственный регистрационный номер отправителя;

3) Bik – банковский идентификационный код отправителя;

4) RegNum – регистрационный номер КО – отправителя по КГРКО;

5) DivisionCode – номер филиала КО – отправителя по КГРКО;

о) Files – файлы включенные в сообщение:

1) Id – уникальный идентификатор файла в формате UUID [16];

2) Name – имя файла;

3) Description – описание файла (необязательно поле, для запросов и предписаний из Банка России содержит имя файла с расширением, однако может содержать запрещённые символы Windows);

4) FileType –тип файла. Допустимы следующие типы файлов:

– Document – любые данные, которые не проходят логический контроль непосредственно на ВП ЕПВВ (файлы документов, любые архивы, в т.ч. зашифрованные, неструктурированные данные и другие файлы);

– SerializedWebForm - xml файл определенной структуры, который может быть проверен ВП ЕПВВ на соответствие его схеме;

– Sign – файл УКЭП, проверка которой влияет на прием/отбраковку сообщения, применяется для основной подписи сообщения и подписи машиночитаемой доверенности;

– PowerOfAttorney – файл машиночитаемой доверенности.

5) Encrypted – признак зашифрованности файла;

6) SignedFile – идентификатор файла сообщения, подписью для которого является данный файл (заполняется только для файлов подписи *.sig);

7) Size - общий размер файла в байтах. Имеет формат int64 (т.е. signed 64 bits);

8) RepositoryInfo – информация о репозиториях (описание репозитория в котором расположен файл. Данная информация используется как для загрузки

файла, так и при его выгрузке):

- Path – путь к файлу в репозитории;
- Host – IP адрес или имя узла репозитория;
- Port – порт для обращения к репозиторию;
- RepositoryType – тип репозитория (значения: aspera, http);
- CheckSum – контрольная сумма файла, для файлов переданных через ТПС «Aspera. Для файлов, отправленных через http, отсутствует;
- CheckSumType – алгоритм расчёта контрольной суммы файла для файлов переданных через ТПС «Aspera. Для файлов, отправленных через http, отсутствует;

п) Receipts – квитанции, полученные в ответ на сообщение:

1) ReceiveTime – время получения квитанции;
2) StatusTime – время из самой квитанции;
3) Status – состояние обработки сообщения (возможные значения и их описание находится в п.3.1.1.5);
4) Message – дополнительная информация из квитанции;
5) Files – файлы, включенные в квитанцию.

Возможные ошибки:

HTTP 400 – Bad Request

BODY

```
{
  "HTTPStatus": 400,
  "ErrorCode": "REQUEST_PLAYLOD_INCORRECT",
  "ErrorMessage": "Неправильное тело запроса",
  "MoreInfo": {}
}
```

HTTP 400 – Bad Request

BODY

```
{
  "HTTPStatus": 400,
  "ErrorCode": "TASK_CODE_MUST_BE_SENT",
  "ErrorMessage": "Должен быть передан параметр Task ",
  "MoreInfo": {}
}
```

HTTP 401 – Unauthorized

BODY

```
{
  "HTTPStatus": 401,
  "ErrorCode": "ACCOUNT_NOT_FOUND ",
  "ErrorMessage": "Аккаунт не найден",
  "MoreInfo": {}
}
```

HTTP 406 – Not Acceptable

BODY

```
{
  "HTTPStatus": 406,
  "ErrorCode": "MESSAGE_SENT_ERROR",
  "ErrorMessage": "Сообщение не может быть отправлено",
  "MoreInfo": {}
}
```

HTTP 406 – Not Acceptable

BODY

```
{
  "HTTPStatus": 406,
  "ErrorCode": "DUPLICATE_FILE_NAME",
  "ErrorMessage": "Имена файлов не должны повторяться",
  "MoreInfo": {}
}
```

HTTP 406 – Not Acceptable

BODY

```
{
  "HTTPStatus": 406,
  "ErrorCode": "FILE_SIZE_ERROR",
  "ErrorMessage": "Размер файла {file.name} Должен быть в
диапазоне от 1 до 9223372036854775807 байт",
  "MoreInfo": {}
}
```

HTTP 406 – Not Acceptable

BODY

```
{
  "HTTPStatus": 406,
  "ErrorCode": " FILE_ENCRYPTION_FLAG_MUST_BE_SET",
  "ErrorMessage": "Для файла {requestfile.name} должен быть указан
флаг шифрования ",
  "MoreInfo": {}
}
```

HTTP 406 – Not Acceptable

BODY

```
{
  "HTTPStatus": 406,
  "ErrorCode": "REQ_FILE_EXTENSION_ERROR",
  "ErrorMessage": "Файл {requestfile.name} с указанным флагом
шифрования должен иметь расширение .enc ",
  "MoreInfo": {}
}
```

HTTP 406 – Not Acceptable

BODY

```
{
  "HTTPStatus": 406,
  "ErrorCode": "SIGN_FILE_EXTENSION_ERROR",
  "ErrorMessage": "Файл подписи {sigfile.name} должен иметь
расширение \'.sig\’",
  "MoreInfo": {}
}
```

HTTP 406 – Not Acceptable

BODY

```
{
  "HTTPStatus": 406,
  "ErrorCode": "SIGN_FILE_NOT_FOUND",
  "ErrorMessage": "Не найден файл для подписи {sigfile.name}.
",
  "MoreInfo": {}
}
```

HTTP 406 – Not Acceptable

BODY

```
{
  "HTTPStatus": 406,
  "ErrorCode": "INCORRECT_RECEIVER",
  "ErrorMessage": "Получатель должен быть КО",
  "MoreInfo": {}
}
```

HTTP 406 – Not Acceptable

BODY

```
{
  "HTTPStatus": 406,
  "ErrorCode": "RECEIVER_NOT_SET",
  "ErrorMessage": "Не определен получатель",
  "MoreInfo": {}
```

```
}
```

HTTP 406 – Not Acceptable

BODY

```
{
  "HTTPStatus": 406,
  "ErrorCode": "SEND_BY_THIS_TASK_NOT_ALLOWED",
  "ErrorMessage": "Не доступна отправка сообщения по указанной
задаче",
  "MoreInfo": {}
}
```

HTTP 406 – Not Acceptable

BODY

```
{
  "HTTPStatus": 406,
  "ErrorCode": "INVALID_FILE_EXTENSION",
  "ErrorMessage": "Недопустимое расширение файла для данной
задачи",
  "MoreInfo": {
     "InvalidFiles": [{
         "FileName": "string
     }],
     "ValidExtesions": "string"
   }
}
```

Где

- InvalidFiles – массив файлов с невалидными расширениями;
- FileName – имена файлов в массиве:
- ValidExtesions – строка с перечислением расширений файлов, допустимых для Задачи сообщения.

HTTP 413 – Message size too large

BODY

```
{
  "HTTPStatus": 413,
  "ErrorCode": " MESSAGE_QUOTA_EXCIDED ",
  "ErrorMessage": " Сообщение не может быть отправлено, так как
размер ЭС превышает квоту (%размер квоты ЭС в МВ%)",
  "MoreInfo": {"AccountQuota": "integer",
               "RestOfQuota": "integer",
               "MessageQuota": "integer"
              }
```

```
}
```

Где

- AccountQuota - общая квота ЛК в Мб;
- RestOfQuota - остаток квоты ЛК в Мб:
- MessageQuota - квота на размер сообщения в Мб

HTTP 413 – Message size too large

BODY

```
{
  "HTTPStatus": 413,
  "ErrorCode": "ACCOUNT_QUOTA_EXCIDED",
  "ErrorMessage": "Сообщение не может быть отправлено, так как
<оставшаяся квота хранения истории обмена ЭС (%остаток квоты в MB
%) будет превышена> / <как размер ЭС превышает квоту (%размер
квоты ЭС в MB%)>",
  "MoreInfo": {"AccountQuota": "integer",
               "RestOfQuota": "integer",
               "MessageQuota": "integer"
               }
}
```

Где

- AccountQuota - общая квота ЛК в Мб;
- RestOfQuota - остаток квоты ЛК в Мб:
- MessageQuota - квота на размер сообщения в Мб

HTTP 422 – Unprocessable Entity

BODY

```
{
  "HTTPStatus": 422,
  "ErrorCode": "INCORRECT_BODY_PARAM",
  "ErrorMessage": "Неверные параметры в теле запроса. Проверьте
сообщение на соответствие параметрам задачи",
  "MoreInfo": {}
}
```

3.1.3.2 Для создания сессии отправки HTTP используется метод POST

Перед отправкой файла необходимо создать сессию отправки:

POST: */messages/{msgId}/files/{fileId}/createUploadSession

Данный метод выполняется только для файлов, отправляемых по HTTP (т.е. "RepositoryType" = "http"). Если "RepositoryType" = "Aspera", то метод вернет ошибку 405 (см. ниже в RESPONSE).

В метод передается следующие параметры:

REQUEST

PATH

```
{
"MsgId": "string($uuid)",
"FileId": "string($uuid)"
}
```

Где:

- MsgId – уникальный идентификатор сообщения в формате UUID [16], полученный в качестве ответа при вызове метода из 3.1.3.1,;
- FileId – уникальный идентификатор файла в формате UUID [16], полученный в качестве ответа при вызове метода из 3.1.3.1.

RESPONSE

HTTP 200 – Ok

```
{
     "UploadUrl": "string",
     "ExpirationDateTime": "string"
}
```

Где:

- UploadUrl – путь для загрузки файла;
- ExpirationDateTime – дата и время истечения сессии.

В случае ошибок:

HTTP 400 – Bad Request

BODY

```
{
  "HTTPStatus": 400,
  "ErrorCode": "FILE_ALLREADY_LOADED",
  "ErrorMessage": "Файл уже загружен",
  "MoreInfo": {}
}
```

HTTP 404 – Not found / Не найден.

BODY

```
{
  "HTTPStatus": 404,
  "ErrorCode": "MESSAGE_NOT_FOUND",
  "ErrorMessage": "Невозможно найти сообщение с указанным id",
  "MoreInfo": {}
}
```

HTTP 404 – Not found / Не найден.

BODY

```
{
  "HTTPStatus": 404,
  "ErrorCode":  "FILE_NOT_FOUND",
  "ErrorMessage": "Невозможно найти файл с указанным id",
  "MoreInfo": {}
}
```

HTTP 405 – Invalid input.

BODY

```
{
  "HTTPStatus": 405,
  "ErrorCode": "BASE_REQUEST_ADDRESS_NOT_FOUND",
  "ErrorMessage": "Не найден базовый адрес запроса",
  "MoreInfo": {}
}
```

HTTP 405 – Invalid input.

BODY

```
{
  "HTTPStatus": 405,
  "ErrorCode": "NOT_ALLOWED_FOR_ASPERA_REPO",
  "ErrorMessage": "Для указанного файла указано RepositoryType =
Aspera, он не может быть загружен через HTTP",
  "MoreInfo": {}
}
```

3.1.3.3 Для отправки файла по HTTP используется метод PUT

PUT: */messages/{msgId}/files/{fileId}

В котором передается следующий параметр:

REQUEST

PATH

```
{
     "MsgId": "string($uuid)",
     "FileId": "string($uuid)"
}
```

HEADER

```
{
     "Content-Type": "string",
     "Content-Length": "integer",
     "Content-Range": "string"
}
```

BODY

```
Content
```

Где:

а) MsgId – уникальный идентификатор сообщения в формате UUID [16], полученный в качестве ответа при вызове метода из п.3.1.3.1;

б) FileId – уникальный идентификатор файла в формате UUID [16], полученный в качестве ответа при вызове метода из п.3.1.3.1;

в) Content-Type - тип передаваемого контента (application/octet-stream);

г) Content-Length – размер чанка в байтах, передав размер в первый раз, для дальнейших отправок следует использовать тот же размер;

д) Content-Range - положение чанка и размер всего файла в байтах в виде ‘bytes 0-25/128’;

е) Content – массив байт.

Пример заполненного HEADER запроса:

```
PUT https://portal5test.cbr.ru/back/rapi2/messages/c84bc346-d57c-
482b-b19d-73234492034b/files/8dce3e13-ae01-4df0-ae69-87730a956bce
HTTP/1.1
Accept: application/json, application/soap+xml
Authorization: Basic S0xETkVPTEFOVFxPU3ljaGV2YTo4ODkxZHJlYW1mYWxs
Content-Type: application/octet-stream
Content-Range: bytes 0-3207/3208
Content-Length: 3208
```

Примечание: в HEADER могут присутствовать дополнительные поля заголовков, устанавливаемые клиентским ПО в соответствии с [7].

RESPONSE

HTTP 201 – Created

```
{
     "Id": "string($uuid)",
     "Name": "string",
     "Description": "string",
     "Encrypted": "boolean",
     "SignedFile": "string($uuid)",
     "Size": "integer",
     "RepositoryInfo": [
     {
          "Path": "string",
          "Host": "string",
          "Port": "integer",
          "RepositoryType": "string"
     }
     ]
}
```

Где:

а) Id – уникальный идентификатор файла в формате UUID [16];

б) Name – имя файла;

в) Description – описание файла (необязательно поле, для запросов и предписаний из Банка России содержит имя файла с расширением, однако может содержать запрещённые символы Windows);

г) Encrypted – признак зашифрованности файла;

д) SignedFile – идентификатор файла сообщения, подписью для которого является данный файл (заполняется только для файлов подписи *.sig);

е) Size - общий размер файла в байтах. Имеет формат int64 (т.е. signed 64 bits);

ж) RepositoryInfo – информация о репозиториях (описание репозитория в котором расположен файл. Данная информация используется как для загрузки файла, так и при его выгрузке):

1) Path – путь к файлу в репозитории;

2) Host – IP адрес или имя узла репозитория;

3) Port – порт для обращения к репозиторию;

4) RepositoryType – тип репозитория (значения: aspera, http).

HTTP 202 – Accepted

```
{
     "NextExpectedRange": "string"
}
```

Где:

NextExpectedRange – описывает следующий ожидаемый диапазон байтов.

HTTP 400 – Bad Request

BODY

```
{
  "HTTPStatus": 400,
  "ErrorCode":  "CONTENT_LENGTH_NOT_SET",
  "ErrorMessage": "Не указан параметр content-length",
  "MoreInfo": {}
}
```

HTTP 400 – Bad Request

BODY

```
{
  "HTTPStatus": 400,
  "ErrorCode":  "CONTENT_RANGE_INCORRECT",
  "ErrorMessage": "Не указан или указан неверно параметр content-
range",
  "MoreInfo": {}
}
```

HTTP 400 – Bad Request

BODY

```
{
  "HTTPStatus": 400,
  "ErrorCode": "DATA_UNREADABLE",
  "ErrorMessage": "Не удалось прочитать данные, убедитесь, что
контент задан корректно!",
  "MoreInfo": {}
}
```

HTTP 400 – Bad Request

BODY

```
{
  "HTTPStatus": 400,
  "ErrorCode":  "CONTENT_LENGTH_INCORRECT",
  "ErrorMessage": "Параметр content-length не соответствует
размеру входящих данных",
  "MoreInfo": {}
}
```

HTTP 400 – Bad Request

BODY

```
{
```

```
  "HTTPStatus": 400,
  "ErrorCode":  "DATA_RANGE_SAVE_ERROR",
  "ErrorMessage": "Ошибка сохранения участка данных",
  "MoreInfo": {}
}
```

HTTP 400 – Bad Request

BODY

```
{
  "HTTPStatus": 400,
  "ErrorCode":  "FILE_SIZE_NOT_MATCH_DB",
  "ErrorMessage": "Размер файла не соответствует записи из базы
данных",
  "MoreInfo": {}
}
```

HTTP 400 – Bad Request

BODY

```
{
  "HTTPStatus": 400,
  "ErrorCode":  "DATA_ALLREADY_WRITTEN",
  "ErrorMessage": "Данные уже записаны",
  "MoreInfo": {}
}
```

HTTP 404 – Not found.

BODY

```
{
  "HTTPStatus": 404,
  "ErrorCode":  "MESSAGE_NOT_FOUND",
  "ErrorMessage": "Невозможно найти сообщение с указанным id",
  "MoreInfo": {}
}
```

HTTP 404 – Not found.

BODY

```
{
  "HTTPStatus": 404,
  "ErrorCode":  "FILE_NOT_FOUND",
  "ErrorMessage": "Невозможно найти файл с указанным id",
  "MoreInfo": {}
}
```

HTTP 405 – Invalid input.

BODY

```
{
  "HTTPStatus": 405,
  "ErrorCode": "NOT_ALLOWED_FOR_ASPERA_REPO",
  "ErrorMessage": "Для указанного файла указано RepositoryType =
Aspera, он не может быть загружен через HTTP",
  "MoreInfo": {}
}
```

HTTP 405 – Invalid input.

BODY

```
{
  "HTTPStatus": 405,
  "ErrorCode":  "BASE_REQUEST_ADDRESS_NOT_FOUND",
  "ErrorMessage": "Не найден базовый адрес запроса",
  "MoreInfo": {}
}
```

3.1.3.4 Для подтверждения отправки сообщения используется метод POST:

POST: */messages/{msgId}

В котором передается следующие параметры:

REQUEST

PATH

```
{
    "MsgId": "string($uuid)"
}
```

Где:

– MsgId – уникальный идентификатор сообщения в формате UUID [16].

RESPONSE

HTTP 200 – Ok;

В случае ошибок:

HTTP 404 – Not found.

BODY

```
{
  "HTTPStatus": 404,
  "ErrorCode":  "MESSAGE_NOT_FOUND",
```

```
  "ErrorMessage": "Невозможно найти сообщение с указанным id",
  "MoreInfo": {}
}
```

HTTP 406 – Not Acceptable

BODY

```
{
  "HTTPStatus": 406,
  "ErrorCode":  "MESSAGE_SENT_ERROR",
  "ErrorMessage": "Сообщение не может быть отправлено",
  "MoreInfo": {}
}
```

HTTP 422 – Unprocessable Entity

BODY

```
{
  "HTTPStatus": 422,
  "ErrorCode":  "INCORRECT_REQUEST_PARAM",
  "ErrorMessage": "Неверные параметры в теле запроса. Проверьте
сообщение на соответствие параметрам задач",
  "MoreInfo": {}
}
```

3.1.4 Получение УИО сообщений, квитанций, файлов и информации

3.1.4.1 Для получения всех сообщений с учетом необязательного фильтра (не более 100 сообщений за один запрос) используется метод GET

GET: */messages

REQUEST

HEADER

```
{
"Content-Type": "string"
}
```

Где:

- Content-Type – тип передаваемого контента (application/json)

Метод может быть дополнен онлайн-запросом, содержащим следующие критерии:

QUERY

```
{
"Task": "string",
```

```
"MinDateTime": "string"($DateTime),
"MaxDateTime": "string"($DateTime),
"MinSize": "integer",
"MaxSize": "integer",
"Type": "string",
"Status": "string",
"Page": "integer"
}
```

Где:

а) Task – наименование задачи (если параметр будет указан, то будут возвращены только сообщения, полученные/отправленные в рамках указанной задачи); Пример: GET: */messages?Task=’Zadacha_23’;

б) MinDateTime – минимально возможная дата создания сообщения (ГОСТ ISO 8601-2001 по маске «yyyy-MM-dd’T’HH:mm:ss’Z’») (если параметр будет указан, то будут возвращены только сообщения, полученные/отправленные с указанной даты, включая ее);

в) MaxDateTime – максимально возможная дата создания сообщения (ГОСТ ISO 8601-2001 по маске «yyyy-MM-dd’T’HH:mm:ss’Z’») (если параметр будет указан, то будут возвращены только сообщения, полученные/отправленные ранее указанной даты, включая ее);

г) MinSize – минимально возможный размер сообщения в байтах (если параметр будет указан, то будут возвращены только сообщения больше и равно указанного размера);

д) MaxSize – максимально возможный размер сообщения в байтах (если параметр будет указан, то будут возвращены только сообщения меньше и равно указанного размера);

е) Type – тип сообщения исходящее (значение: outbox), входящее (значение: inbox) (если параметр будет указан, то будут возвращены только сообщения соответствующего типа);

ж) Status – статус сообщения Черновик (значение: draft), Отправлено (значение: sent), Загружено (значение: delivered), Ошибка (значение: error), Принято в обработку (значение: processing), Зарегистрировано (значение: registered), Отклонено (значение: rejected), Новое (значение: new), Прочитано (значение: read), Отправлен ответ (значение: replied), Доставлено (значение: success) (если параметр будет указан, то будут возвращены только сообщения с соответствующим статусом);

з) Page – номер страницы списка сообщений в разбивке по 100 сообщений (если не задан, то вернутся первые 100 сообщений). Пример: GET: */messages?page={n}, где {n} – номер страницы содержащей 100 сообщений (n-я сотня сообщений). Допустимые значения: n > 0 (положительные целые числа, больше 0). Если запрос страницы не указан, возвращается

первая страница сообщений. Если n за границами диапазона страниц, то вернется пустой массив сообщений. В случае некорректного номера страницы – ошибка.

Критерии запроса могут быть использованы как по одиночке, так и в любом сочетании, через ‘&’, например:

GET: */messages?Type=”inbox”&Status =”read”&Page=2

Т.е. выбрать входящие прочитанные сообщения со 101-го по 200-е.

RESPONSE

В случае корректного запроса

HTTP 200 – OK

```
[
{
          "Id": "string($uuid)",
          "CorrelationId": "string($uuid)",
          "GroupId": "string($uuid)",
          "Type": "string",
          "Title": "string",
          "Text": "string",
          "CreationDate": "string"($DateTime),
          "UpdatedDate": "string"($DateTime),
          "Status": "string",
          "TaskName": "string",
          "RegNumber": "string",
          "TotalSize": "integer",
          "Sender": [
          {
               "Inn":  string",
               "Ogrn": "string",
               "Bik": "string",
               "RegNum": "string",
               "DevisonCode": "string"
          },
          "Files": [
          {
               "Id": "string($uuid)",
               "Name": "string",
               "Description": "string",
               "FileType": "string",
               "Encrypted": "boolean",
               "SignedFile": "string($uuid)",
               "Size": "integer",
               "RepositoryInfo": [
               {
                    "Path": "string",
                    "Host": "string",
```

```
                    "Port": "integer",
                    "RepositoryType": "string",
                    "CheckSum": "string",
                    "CheckSumType": "string"
               }
               ]
          }
          ],
          "Receipts": [
          {
               "Id": "string($uuid)",
               "ReceiveTime": "string",
               "StatusTime": "string",
               "Status": "string",
               "Message": "string",
               "Files": [
               {
                    "Id": "string($uuid)",
                    "Name": "string",
                    "Description": "string",
                    "FileType": "string",
                    "Encrypted": "boolean",
                    "SignedFile": "string($uuid)",
                    "Size": "integer",
                    "RepositoryInfo": [
                    {
                         "Path": "string",
                         "Host": "string",
                         "Port": "integer",
                         "RepositoryType": "string",
                    }
                    ]
               }
               ]
          }
          ]
}
]
```

Где:

а) Id – уникальный идентификатор сообщения в формате UUID [16];

б) CorrelationId - идентификатор корреляции сообщения в формате UUID [16];

в) GroupId –идентификатор группы сообщений в формате UUID [16];

г) Type – тип сообщения исходящее (значение: outbox) или входящее (значение: inbox);

д) Title – название сообщения;

е) Text – текст сообщения;

ж) CreationDate – дата создания сообщения (ГОСТ ISO 8601-2001 по маске «yyyy-MM-dd’T’HH:mm:ss’Z’»);

з) UpdatedDate – дата последнего изменения статуса сообщения (ГОСТ ISO 8601-2001 по маске «yyyy-MM-dd’T’HH:mm:ss’Z’»);

и) Status – статус сообщения (возможные значения и их описание находится в п.3.1.1.5);

к) TaskName – наименование задачи;

л) RegNumber – регистрационный номер;

м) TotalSize – общий размер сообщения в байтах. Имеет формат int64 (т.е. signed 64 bits);

н) Sender – отправитель сообщения (необязательное поле, только для сообщений, отправляемых другими Пользователями):

1) Inn – ИНН отправителя сообщения
2) Ogrn – ОГРН отправителя сообщения
3) Bik – БИК отправителя сообщения
4) RegNum – рег. номер КО-отправителя сообщения по КГРКО
5) DevisonCode – номер филиала КО-отправителя сообщения по КГРКО

о) Files – файлы, включенные в сообщение:

1) Id – уникальный идентификатор файла в формате UUID [16];
2) Name – имя файла;
3) Description – описание файла (необязательное поле, для запросов и предписаний из Банка России содержит имя файла с расширением, однако может содержать запрещённые символы Windows);
4) FileType – тип файла. Допустимы следующие типы файлов:

- Document – любые данные, которые не проходят логический контроль непосредственно на ВП ЕПВВ (файлы документов, любые архивы, в т.ч. зашифрованные, неструктурированные данные и другие файлы);
- SerializedWebForm - xml файл определенной структуры, который может быть проверен ВП ЕПВВ на соответствие его схеме;
- Sign – файл УКЭП, проверка которой влияет на прием/отбраковку сообщения, применяется для основной подписи сообщения и подписи машиночитаемой доверенности;
- PowerOfAttorney – файл машиночитаемой доверенности.

5) Encrypted – признак зашифрованности файла;

6) SignedFile – идентификатор файла сообщения, подписью для которого является данный файл (заполняется только для файлов подписи *.sig);

7) Size - общий размер файла в байтах. Имеет формат int64 (т.е. signed 64 bits);

8) RepositoryInfo – информация о репозиториях (описание репозитория в котором расположен файл. Данная информация используется как для загрузки файла, так и при его выгрузке):

- Path – путь к файлу в репозитории;
- Host – IP адрес или имя узла репозитория;
- Port – порт для обращения к репозиторию;
- RepositoryType – тип репозитория (значения: aspera, http);
- CheckSum – контрольная сумма файла, для файлов переданных через ТПС «Aspera. Для файлов, отправленных через http, отсутствует;
- CheckSumType – алгоритм расчёта контрольной суммы файла для файлов переданных через ТПС «Aspera. Для файлов, отправленных через http, отсутствует;

п) Receipts – квитанции, полученные в ответ на сообщение:

1) Id – уникальный идентификатор файла в формате UUID [16];

2) ReceiveTime – время получения квитанции;

3) StatusTime – время из самой квитанции;

4) Status – состояние обработки сообщения (возможные значения и их описание находится в п.3.1.1.5);

5) Message – дополнительная информация из квитанции;

6) Files – файлы, включенные в квитанцию.

Примечание: Для потоков, по которым не сохраняются чанки файлов в ЭС (сохраняется только метаинформация о файлах), возвращается пустое значение Id файла, нулевой размер файла (Size = 0) и массив с пустыми значениями для блока RepositoryInfo:

```
"Files": [
…
    {
        "Id": "",
        …
        "Size": 0,
        "RepositoryInfo": [
```

```
            {
              "Path": "",
              "Host": "",
              "Port": 0,
              "RepositoryType": "null"
            }
          ]
    {
]
```

Что бы определить, сколько всего записей в списке сообщений в наличии, API возвращает 5 (пять) заголовков с каждым положительным ответом (HTTP 200).

HEADER

```
    "EPVV-Total": "integer"
    "EPVV-TotalPages": "integer"
    "EPVV-CurrentPage: "integer"
    "EPVV-PerCurrentPage: "integer"
    "EPVV-PerNextPage: "integer"
```

Где:

- EPVV-Total – общее количество сообщений в запросе;
- EPVV-TotalPages – общее количество страниц, охватывая все сообщения по 100 сообщений на странице;
- EPVV-CurrentPage – номер текущей страницы;
- EPVV-PerCurrentPage – количество сообщений на текущей странице;
- EPVV-PerNextPage – количество сообщений на следующей странице.

В случае ошибок:

HTTP 400 – Bad Request

В случае некорректного указания номера страницы в запросе (отрицательное число, строка текста), возвращается ошибка. В теле ответа объект с описанием ошибки.

BODY

```
{
  "HTTPStatus": 400,
  "ErrorCode": "INCORRECT_PAGE_NUM",
  "ErrorMessage": "Произошла ошибка. Некорректное значение
страницы: {page}",
  "MoreInfo": {
              "TotalItems": "integer",
              "TotalPages": "integer"
  }
}
```

Где:

- TotalItems - всего элементов в массиве;
- TotalPages - всего станиц.

HTTP 400 – Bad Request

В случае некорректного указания какого-либо из QUERY-параметров.

BODY

```
{
  "HTTPStatus": 400,
  "ErrorCode": "NCORRECT_REQUEST_PARAM",
  "ErrorMessage": " Некорректное значение параметра запроса ",
  "MoreInfo": {
              "BadParams": ["String"]
}}
```

Где:

- BadParams – массив имен некорректных QUERY-параметров в строке запроса.

3.1.4.2 Для получения данных о конкретном сообщении используется метод GET:

GET: */messages/{msgId}

REQUEST

PATH

```
{
     "MsgId": "string($uuid)"
}
```

Где:

- msgId – уникальный идентификатор сообщения.

RESPONSE

HTTP 200 – Ok

```
{
     "Id": "string($uuid)",
     "CorrelationId": "string($uuid)",
     "GroupId": "string($uuid)",
     "Type": "string",
     "Title": "string",
```

```
    "Text": "string",
    "CreationDate": "string",
    "UpdatedDate": "string",
    "Status": "string",
    "TaskName": "string",
    "RegNumber": "string",
    "TotalSize": "integer",
    "Sender": [
    {
        "Inn": "string",
        "Ogrn": "string",
        "Bik": "string",
        "RegNum": "string",
        "DevisonCode": "string"
    },
    "Files": [
    {
        "Id": "string($uuid)",
        "Name": "string",
        "Description": "string",
        "FileType": "string",
        "Encrypted": "boolean",
        "SignedFile": "string($uuid)",
        "Size": "integer",
        "RepositoryInfo": [
        {
            "Path": "string",
            "Host": "string",
            "Port": "integer",
            "RepositoryType": "string",
            "CheckSum": "string",
            "CheckSumType": "string"
        }
        ]
    }
    ],
    "Receipts": [
    {
        "Id": "string($uuid)",
        "ReceiveTime": "string",
        "StatusTime": "string",
        "Status": "string",
        "Message": "string",
        "Files": [
        {
            "Id": "string",
            "Name": "string",
            "Description": "string",
            "FileType": "string",
            "Encrypted": "boolean",
            "SignedFile": "string($uuid)",
```

```
                "Size": "integer",
                "RepositoryInfo": [
                {
                     "Path": "string",
                     "Host": "string",
                     "Port": "integer",
                     "RepositoryType": "string"
                }
                ]
           }
           ]
     }
     ]
}
```

Где:

а) Id – уникальный идентификатор сообщения в формате UUID [16];

б) CorrelationId - идентификатор корреляции сообщения в формате UUID [16];

в) GroupId –идентификатор группы сообщений в формате UUID [16];

г) Type – тип сообщения исходящее (значение: outbox) или входящее (значение: inbox);

д) Title – название сообщения;

е) Text – текст сообщения;

ж) CreationDate – дата создания сообщения (ГОСТ ISO 8601-2001 по маске «yyyy-MM-dd’T’HH:mm:ss’Z’»);

з) UpdatedDate – дата последнего изменения статуса сообщения (ГОСТ ISO 8601-2001 по маске «yyyy-MM-dd’T’HH:mm:ss’Z’»);

и) Status – статус сообщения (возможные значения и их описание находится в п.3.1.1.5);

к) TaskName – наименование задачи;

л) RegNumber – регистрационный номер;

м) TotalSize – общий размер сообщения в байтах. Имеет формат int64 (т.е. signed 64 bits);

н) Sender – отправитель сообщения (необязательное поле, только для сообщений, отправляемых другими Пользователями):

1) Inn – ИНН отправителя сообщения

2) Ogrn – ОГРН отправителя сообщения

3) Bik – БИК отправителя сообщения

4) RegNum – рег. номер КО-отправителя сообщения по КГРКО

5) DevisonCode – номер филиала КО-отправителя сообщения по КГРКО

o) Files – файлы, включенные в сообщение:

1) Id – уникальный идентификатор файла в формате UUID [16];

2) Name – имя файла;

3) Description – описание файла (необязательное поле, для запросов и предписаний из Банка России содержит имя файла с расширением, однако может содержать запрещённые символы Windows);

4) FileType – указывается тип файла. Допустимы следующие типы файлов:

- Document – любые данные, которые не проходят логический контроль непосредственно на ВП ЕПВВ (файлы документов, любые архивы, в т.ч. зашифрованные, неструктурированные данные и другие файлы);
- SerializedWebForm - xml файл определенной структуры, который может быть проверен ВП ЕПВВ на соответствие его схеме;
- Sign – файл УКЭП, проверка которой влияет на прием/отбраковку сообщения, применяется для основной подписи сообщения и подписи машиночитаемой доверенности;
- PowerOfAttorney – файл машиночитаемой доверенности.

5) Encrypted – признак зашифрованности файла;

6) SignedFile – идентификатор файла сообщения, подписью для которого является данный файл (заполняется только для файлов подписи *.sig);

7) Size - общий размер файла в байтах. Имеет формат int64 (т.е. signed 64 bits);

8) RepositoryInfo – информация о репозиториях (описание репозитория в котором расположен файл. Данная информация используется как для загрузки файла, так и при его выгрузке):

- Path – путь к файлу в репозитории;
- Host – IP адрес или имя узла репозитория;
- Port – порт для обращения к репозиторию;
- RepositoryType – тип репозитория (значения: aspera, http);
- CheckSum – контрольная сумма файла, для файлов переданных через ТПС «Aspera. Для файлов, отправленных через http, отсутствует;

– CheckSumType – алгоритм расчёта контрольной суммы файла для файлов, переданных через ТПС «Aspera. Для файлов, отправленных через http, отсутствует;

п) Receipts – квитанции, полученные в ответ на сообщение:

1) Id - уникальный идентификатор файла в формате UUID [16];

2) ReceiveTime – время размещения квитанции в Личном Кабинете;

3) StatusTime – время из самой квитанции;

4) Status – состояние обработки сообщения (возможные значения и их описание находится в п.3.1.1.5);

5) Message – дополнительная информация из квитанции;

6) Files – файлы, включенные в квитанцию.

Примечание: Для потоков, по которым не сохраняются чанки файлов в ЭС (сохраняется только метаинформация о файлах), возвращается пустое значение Id файла, нулевой размер файла (Size = 0) и массив с пустыми значениями для блока RepositoryInfo:

```
"Files": [
…
     {
          "Id": "",
          …

          "Size": 0,
          "RepositoryInfo": [
             {
               "Path": "",
               "Host": "",
               "Port": 0,
               "RepositoryType": "null"
             }
           ]
     {
…
]
```

В случае ошибки в Id сообщения

HTTP 404 – Not found.

BODY

```
{
  "HTTPStatus": 404,
  "ErrorCode":  "MESSAGE_NOT_FOUND",
  "ErrorMessage": "Невозможно найти сообщение с указанным id",
```

```
  "MoreInfo": {}
}
```

3.1.4.3 Для скачивания конкретного сообщения используется метод GET:

GET: */ messages/{msgId}/download

REQUEST

PATH

```
{
     "MsgId": "string($uuid)"
}
```

HEADER

```
"Range": "string"
```

Где:

а) MsgId – уникальный идентификатор сообщени в формате UUID [16];

б) Range – запрашиваемый диапазон байтов (необязательное поле). В случае указания имеет вид: Range: bytes = {диапазон байт}, где диапазон байт от 0 до Size-1. Указание множественных диапазонов не поддерживается.

Например:

- Range: bytes =1024-4095, что означает будет скачан диапазон с первого по четвертый килобайты;
- Range: bytes =4096- , означает будет скачан диапазон с четвертого килобайта до конца файла;
- Range: bytes = -4096, означает будут скачаны последние четыре килобайта файла.

Подробнее о заголовке Range см. документ «Hypertext Transfer Protocol (HTTP/1.1): Range Requests» [7].

RESPONSE

HTTP 200 – OK (для полного получения);

HEADER

- Accept-Ranges: bytes – Заголовок информирует клиента о том, что он может запрашивать у сервера фрагменты, указывая их смещения от начала файла в байтах;

– Content-Length: {полный размер загружаемого сообщения};

или

HTTP 206 – Partial content (для получения определённого диапазона, если был указан Range);

HEADER

– Accept-Ranges: bytes;
– Content-Range: bytes {начало фрагмента}-{конец фрагмента}/{полный размер сообщения}, например: Content-Range: bytes 1024-4095/8192, означает, что был предоставлен фрагмент с первого по четвертый килобайты из сообщения в 8 килобайт;
– Content-Length: {размер тела сообщения}, то есть передаваемого фрагмента, например: Content-Length: 1024, означает, что размер фрагмента один килобайт.

BODY – запрашиваемое сообщение целиком или указанный в Range диапазон байт.

В случае ошибок:

HTTP 404 – Not found

BODY

```
{
  "HTTPStatus": 404,
  "ErrorCode": "MESSAGE_NOT_FOUND",
  "ErrorMessage": "Невозможно найти сообщение с указанным id",
  "MoreInfo": {}
}
```

HTTP 404 – Not found

BODY

```
{
  "HTTPStatus": 404,
  "ErrorCode":  "FILE_TEMPORARY_NOT_AVAILABLE",
  "ErrorMessage": "Файлы сообщения временно недоступны",
  "MoreInfo": {
            "MissedFiles": [{
                          "Id": "string"
                          "FileName": "string"
                          "RepositoriInfo":
                                    "RepositoryInfo"{...}
```

```
                                    }]
                }
}
```

Где

- Id - уникальный идентификатор файла;
- FileName - имя временно недоступного файла;
- RepositoriInfo – информация о репозиториях (см. выше).

HTTP 410 – Gone.

BODY

```
{
  "HTTPStatus": 410,
  "ErrorCode": "FILE_PERMANENTLY_NOT_AVAILABLE",
  "ErrorMessage": "Файлы сообщения более недоступны или задача не
предусматривает их хранения",
  "MoreInfo": {
             "MissedFiles": [{
                             "Id": "string"
                             "FileName": "string"
                            }]
               }
}
```

Где

Id - уникальный идентификатор файла;

FileName - имя файла, без содержимого;

HTTP 416 – Range Not Satisfiable.

BODY

```
{
  "HTTPStatus": 416,
  "ErrorCode":  "INCORRECT_BYTE_RANGE",
  "ErrorMessage": "В запросе не верно указан диапазон байт",
  "MoreInfo": {}
}
```

3.1.4.4 Для получения данных о конкретном файле используется метод GET

GET: */messages/{msgId}/files/{fileId}

REQUEST

PATH

```
{
     "MsgId": "string($uuid)",
     "FileId": "string($uuid)"
}
```

Где:

– MsgId – уникальный идентификатор сообщения;

– FileId – уникальный идентификатор файла.

RESPONSE

HTTP 200 – Ok

```
{
     "Id": "string($uuid)",
     "Name": "string",
     "Description": "string",
     "FileType": "string",
     "Encrypted": "boolean",
     "SignedFile": "string($uuid)",
     "Size": "integer",
     "RepositoryInfo": [
     {
          "Path": "string",
          "Host": "string",
          "Port": "integer",
          "RepositoryType": "string",
          "CheckSum": "string",
          "CheckSumType": "string"
          }
          ]
     }
     ]
}
```

Где:

а) Id – уникальный идентификатор файла в формате UUID [16];

б) Name – имя файла;

в) Description – описание файла (необязательное поле, для запросов и предписаний из Банка России содержит имя файла с расширением, однако может содержать запрещённые символы Windows);

г) FileType – указывается тип файла. Допустимы следующие типы файлов:

1) Document – любые данные, которые не проходят логический контроль непосредственно на ВП ЕПВВ (файлы документов, любые архивы, в т.ч.

зашифрованные, неструктурированные данные и другие файлы);

2) SerializedWebForm - xml файл определенной структуры, который может быть проверен ВП ЕПВВ на соответствие его схеме;

3) Sign – файл УКЭП, проверка которой влияет на прием/отбраковку сообщения, применяется для основной подписи сообщения и подписи машиночитаемой доверенности;

4) PowerOfAttorney – файл машиночитаемой доверенности.

д) Encrypted – признак зашифрованности файла;

е) SignedFile – идентификатор файла сообщения, подписью для которого является данный файл (заполняется только для файлов подписи *.sig);

ж) Size - общий размер файла в байтах. Имеет формат int64 (т.е. signed 64 bits);

з) RepositoryInfo – информация о репозиториях (описание репозитория в котором расположен файл. Данная информация используется как для загрузки файла, так и при его выгрузке):

1) Path – путь к файлу в репозитории;

2) Host – IP адрес или имя узла репозитория;

3) Port – порт для обращения к репозиторию;

4) RepositoryType – тип репозитория (значения: aspera, http).

5) CheckSum – контрольная сумма файла, для файлов, переданных через ТПС «Aspera. Для файлов, отправленных через http, отсутствует;

6) CheckSumType – алгоритм расчёта контрольной суммы файла для файлов, переданных через ТПС «Aspera. Для файлов, отправленных через http, отсутствует.

Примечание: Для потоков, по которым не сохраняются чанки файлов в ЭС (сохраняется только метаинформация о файлах), возвращается пустое значение Id файла, нулевой размер файла (Size = 0) и массив с пустыми значениями для блока RepositoryInfo:

```
{
     "Id": "",
     …

     "Size": 0,
     "RepositoryInfo": [
        {
          "Path": "",
          "Host": "",
          "Port": 0,
          "RepositoryType": "null"
```

```
            }
        ]
    {
```

В случае ошибок:

HTTP 404 – Not found.

BODY

```
{
  "HTTPStatus": 404,
  "ErrorCode": "MESSAGE_NOT_FOUND",
  "ErrorMessage": "Невозможно найти сообщение с указанным id",
  "MoreInfo": {}
}
```

HTTP 404 – Not found

BODY

```
{
  "HTTPStatus": 404,
  "ErrorCode": "FILE_NOT_FOUND",
  "ErrorMessage": "Невозможно найти файл с указанным id",
  "MoreInfo": {}
}
```

3.1.4.5 Для скачивания конкретного файла из конкретного сообщения используется метод GET:

GET: */messages/{msgId}/files/{fileId}/download

REQUEST

PATH

```
{
    "MsgId": "string",
    "FileId": "string"
}
```

HEADER

```
"Range": "string"
```

Где:

a) MsgId – уникальный идентификатор сообщения в формате UUID [16];

б) FileId – уникальный идентификатор файла в формате UUID [16];

в) Range – запрашиваемый диапазон байтов (необязательное поле). В случае указания имеет вид: Range: bytes = {диапазон байт}, где диапазон байт от 0 до Size-1. Указание множественных диапазонов не поддерживается.

Например:

- Range: bytes =1024-4095, что означает будет скачан диапазон с первого по четвертый килобайты;
- Range: bytes =4096- , означает будет скачан диапазон с четвертого килобайта до конца файла;
- Range: bytes = -4096 , означает будут скачаны последние четыре килобайта файла.

Подробнее о заголовке Range см. документ «Hypertext Transfer Protocol (HTTP/1.1): Range Requests» [7].

RESPONSE

HTTP 200 – OK (для полного получения файла);

HEADER

- Accept-Ranges: bytes; – Заголовок информирует клиента о том, что он может запрашивать у сервера фрагменты, указывая их смещения от начала файла в байтах;
- Content-Length: {полный размер загружаемого сообщения};

или

HTTP 206 – Partial content (для получения определённого диапазона, если был указан Range);

HEADER

- Accept-Ranges: bytes;
- Content-Range: bytes {начало фрагмента}-{конец фрагмента}/{полный размер сообщения}, например: Content-Range: bytes 1024-4095/8192 , означает, что был предоставлен фрагмент с первого по четвертый килобайты из сообщения в 8 килобайт;
- Content-Length: {размер тела сообщения}, то есть передаваемого фрагмента, например: Content-Length: 1024 , означает, что размер фрагмента один килобайт.

В BODY – запрашиваемый файл или диапазон;

В случае ошибок:

HTTP 404 – Not found.

BODY

```
{
  "HTTPStatus": 404,
  "ErrorCode":  "MESSAGE_NOT_FOUND",
  "ErrorMessage": "Невозможно найти сообщение с указанным id",
  "MoreInfo": {}
}
```

HTTP 404 – Not found.

BODY

```
{
  "HTTPStatus": 404,
  "ErrorCode":  "FILE_NOT_FOUND",
  "ErrorMessage": "Невозможно найти файл с указанным id",
  "MoreInfo": {}
}
```

HTTP 404 – Not found

BODY

```
{
  "HTTPStatus": 404,
  "ErrorCode":  "FILE_TEMPORARY_NOT_AVAILABLE",
  "ErrorMessage": "Файлы сообщения временно недоступны",
  "MoreInfo": {
            "MissedFiles": [{
                            "Id": "string($uuid)"
                            "FileName": “string”
                            "RepositoriInfo":
                                       “RepositoryInfo”{...}
                             }]
              }
}
```

Где

- Id – уникальный идентификатор файла в формате UUID [16];
- FileName – имя временно недоступного файла;
- RepositoriInfo – информация о репозиториях (см. выше).

HTTP 410 – Gone.

BODY

```
{
  "HTTPStatus": 410,
  "ErrorCode":  "FILE_PERMANENTLY_NOT_AVAILABLE",
  "ErrorMessage": "Файлы сообщения более недоступны или задача не
предусматривает их хранения",
  "MoreInfo": {
             "MissedFiles": [{
                              "Id": "string($uuid)"
                              "FileName": “string”
                             }]
               }
}
```

Где

- Id – уникальный идентификатор файла в формате UUID [16];
- FileName – имя файла, без содержимого;

HTTP 416 – Range Not Satisfiable.

BODY

```
{
  "HTTPStatus": 416,
  "ErrorCode":  "INCORRECT_BYTE_RANGE",
  "ErrorMessage": "В запросе не верно указан диапазон байт",
  "MoreInfo": {}
}
```

3.1.4.6 Для получения данных о квитанциях на сообщение используется метод GET:

GET: */messages/{msgId}/receipts

REQUEST

PATH

```
{
     "MsgId": "string($uuid)"
}
```

Где:

- MsgId – уникальный идентификатор сообщения в формате UUID [16].

RESPONSE

HTTP 200 – Ok

```
[
{
     "Id": "string($uuid)",
     "ReceiveTime": "string",
     "StatusTime": "string",
     "Status": "string",
     "Message": "string",
     "Files": [
     {
          "Id": "string($uuid)",
          "Name": "string",
          "Description": "string",
          "FileType": "string",
          "Encrypted": "boolean",
          "SignedFile": "string($uuid)",
          "Size": "integer",
          "RepositoryInfo": [
          {
               "Path": "string",
               "Host": "string",
               "Port": "integer",
               "RepositoryType": "string"
          }
          ]
     }
     ]
}
]
```

Где:

а) Id – уникальный идентификатор файла в формате UUID [16];

б) ReceiveTime – время размещения квитанции в Личном Кабинете;

в) StatusTime – время из самой квитанции;

г) Status – состояние обработки сообщения (возможные значения и их описание находится в п.3.1.1.5);

д) Message – дополнительная информация из квитанции;

е) Files – файлы, включенные в квитанцию:

1) Id – уникальный идентификатор файла в формате UUID [16];

2) Name – имя файла;

3) Description – описание файла (необязательное поле, для запросов и предписаний из Банка России содержит имя файла с расширением, однако может содержать запрещённые символы Windows);

4) FileType – указывается тип файла. Допустимы следующие типы файлов:

- Document – любые данные (файлы документов, неструктурированные данные и другие файлы);
- Sign – файл УКЭП к файлам документов;

5) Encrypted – признак зашифрованности файла;

6) SignedFile – идентификатор файла сообщения, подписью для которого является данный файл (заполняется только для файлов подписи *.sig);

7) Size - общий размер файла в байтах. Имеет формат int64 (т.е. signed 64 bits);

8) RepositoryInfo – информация о репозиториях (описание репозитория в котором расположен файл. Данная информация используется как для загрузки файла, так и при его выгрузке):

- Path – путь к файлу в репозитории;
- Host – IP адрес или имя узла репозитория;
- Port – порт для обращения к репозиторию;
- RepositoryType – тип репозитория (значения: aspera, http).

В случае ошибок:

HTTP 404 – Not found.

BODY

```
{
  "HTTPStatus": 404,
  "ErrorCode":  "MESSAGE_NOT_FOUND",
  "ErrorMessage": "Невозможно найти сообщение с указанным id",
  "MoreInfo": {}
}
```

HTTP 404 – Not found

BODY

```
{
  "HTTPStatus": 404,
  "ErrorCode":  "RECEIPT_NOT_FOUND",
  "ErrorMessage": "Невозможно найти квитанцию с указанным id",
  "MoreInfo": {}
}
```

3.1.4.7 Для получения данных о квитанции на сообщение используется метод GET

GET: */messages/{msgId}/receipts/{rcptId}

REQUEST

PATH

```
{
     "MsgId": "string($uuid)",
     "rcptId": "string($uuid)"
}
```

Где:

- MsgId – уникальный идентификатор сообщения в формате UUID [16];
- rcptId – уникальный идентификатор квитанции в формате UUID [16].

RESPONSE

HTTP 200 – Ok

```
[
{
     "Id": "string($uuid)",
     "ReceiveTime": "string",
     "StatusTime": "string",
     "Status": "string",
     "Message": "string",
     "Files": [
     {
          "Id": "string($uuid)",
          "Name": "string",
          "Description": "string",
          "FileType": "string",
          "Encrypted": "boolean",
          "SignedFile": "string($uuid)",
          "Size": "integer",
          "RepositoryInfo": [
          {
               "Path": "string",
               "Host": "string",
               "Port": "integer",
               "RepositoryType": "string"
          }
          ]
     }
     ]
}
]
```

Где:

а) Id – уникальный идентификатор файла в формате UUID [16];

б) ReceiveTime – время размещения квитанции в Личном Кабинете;

в) StatusTime – время из самой квитанции;

г) Status – состояние обработки сообщения (возможные значения и их описание находится в п.3.1.1.5);

д) Message – дополнительная информация из квитанции;

е) Files – файлы, включенные в квитанцию:

1) Id – уникальный идентификатор файла в формате UUID [16];

2) Name – имя файла;

3) Description – описание файла (необязательное поле, для запросов и предписаний из Банка России содержит имя файла с расширением, однако может содержать запрещённые символы Windows);

4) FileType – указывается тип файла. Допустимы следующие типы файлов:

- Document – любые данные (файлы документов, неструктурированные данные и другие файлы);
- Sign – файл УКЭП к файлам документов;

5) Encrypted – признак зашифрованности файла;

6) SignedFile – идентификатор файла сообщения, подписью для которого является данный файл (заполняется только для файлов подписи *.sig);

7) Size - общий размер файла в байтах. Имеет формат int64 (т.е. signed 64 bits);

8) RepositoryInfo – информация о репозиториях (описание репозитория в котором расположен файл. Данная информация используется как для загрузки файла, так и при его выгрузке):

- Path – путь к файлу в репозитории;
- Host – IP адрес или имя узла репозитория;
- Port – порт для обращения к репозиторию;
- RepositoryType – тип репозитория (значения: aspera, http).

В случае ошибок:

HTTP 404 – Not found.

BODY

```
{
```

```
  "HTTPStatus": 404,
  "ErrorCode":  "MESSAGE_NOT_FOUND",
  "ErrorMessage": "Невозможно найти сообщение с указанным id",
  "MoreInfo": {}
}
```

HTTP 404 – Not found

BODY

```
{
  "HTTPStatus": 404,
  "ErrorCode":  "RECEIPT_NOT_FOUND",
  "ErrorMessage": "Невозможно найти квитанцию с указанным id",
  "MoreInfo": {}
}
```

3.1.4.8 Для получения данных о файле квитанции на сообщение используется метод GET

GET: */messages/{msgId}/receipts/{rcptId}/files/{fileId}

REQUEST

PATH

```
{
    "MsgId": "string($uuid)",
    "rcptId": "string($uuid)",
    "fileId": "string($uuid)"
}
```

Где:

- MsgId – уникальный идентификатор сообщения в формате UUID [16];
- rcptId – уникальный идентификатор квитанции в формате UUID [16];
- fileId – уникальный идентификатор файла в формате UUID [16].

RESPONSE

HTTP 200 – Ok

```
{
    "Id": "string($uuid)",
    "Name": "string",
    "Description": "string",
    "FileType": "string",
    "Encrypted": "boolean",
    "SignedFile": "string($uuid)",
    "Size": "integer",
```

```
    "RepositoryInfo": [
    {
         "Path": "string",
         "Host": "string",
         "Port": "integer",
         "RepositoryType": "string"
    }
    ]
}
```

Где:

а) Id – уникальный идентификатор файла в формате UUID [16];

б) Name – имя файла;

в) Description – описание файла (необязательное поле, для запросов и предписаний из Банка России содержит имя файла с расширением, однако может содержать запрещённые символы Windows);

г) FileType – указывается тип файла. Допустимы следующие типы файлов:

1) Document – любые данные (файлы документов, любые архивы, в т.ч. зашифрованные, неструктурированные данные и другие файлы);

2) Sign – файл УКЭП к файлам документов;

д) Encrypted – признак зашифрованности файла;

е) SignedFile – идентификатор файла сообщения, подписью для которого является данный файл (заполняется только для файлов подписи *.sig);

ж) Size - общий размер файла в байтах. Имеет формат int64 (т.е. signed 64 bits);

з) RepositoryInfo – информация о репозиториях (описание репозитория в котором расположен файл. Данная информация используется как для загрузки файла, так и при его выгрузке):

1) Path – путь к файлу в репозитории;

2) Host – IP адрес или имя узла репозитория;

3) Port – порт для обращения к репозиторию;

4) RepositoryType – тип репозитория (значения: aspera, http).

В случае ошибок:

HTTP 404 – Not found.

BODY

```
{
  "HTTPStatus": 404,
```

```
  "ErrorCode":  "MESSAGE_NOT_FOUND",
  "ErrorMessage": "Невозможно найти сообщение с указанным id",
  "MoreInfo": {}
}
```

HTTP 404 – Not found

BODY

```
{
  "HTTPStatus": 404,
  "ErrorCode": "RECEIPT_NOT_FOUND",
  "ErrorMessage": "Невозможно найти квитанцию с указанным id",
  "MoreInfo": {}
}
```

HTTP 404 – Not found.

BODY

```
{
  "HTTPStatus": 404,
  "ErrorCode": "FILE_NOT_FOUND",
  "ErrorMessage": "Невозможно найти файл с указанным id",
  "MoreInfo": {}
}
```

3.1.4.9 Для скачивания файла квитанции на сообщение используется метод GET

GET: */messages/{msgId}/receipts/{rcptId}/files/{fileId}/download

REQUEST

PATH

```
{
    "MsgId": "string($uuid)",
    "rcptId": "string($uuid)",
    "fileId": "string($uuid)"
}
```

HEADER

```
    "Range": "string"
```

Где:

а) MsgId – уникальный идентификатор сообщения в формате UUID [16];

б) rcptId – уникальный идентификатор квитанции в формате UUID [16];

в) fileId – уникальный идентификатор файл в формате UUID [16];

г) Range – запрашиваемый диапазон байтов (необязательное поле). В случае

указания имеет вид: Range: bytes = {диапазон байт}, где диапазон байт от 0 до Size-1.

Указание множественных диапазонов не поддерживается.

Например:

- Range: bytes =1024-4095, что означает будет скачан диапазон с первого по четвертый килобайты;
- Range: bytes =4096-, означает будет скачан диапазон с четвертого килобайта до конца файла;
- Range: bytes = -4096, означает будут скачаны последние четыре килобайта файла.

Подробнее о заголовке Range см. документ «Hypertext Transfer Protocol (HTTP/1.1): Range Requests» [7].

RESPONSE

HTTP 200 – OK (для полного получения файла);

HEADER

- Accept-Ranges: bytes; – Заголовок информирует клиента о том, что он может запрашивать у сервера фрагменты, указывая их смещения от начала файла в байтах;
- Content-Length: {полный размер загружаемого сообщения};

или

HTTP 206 – Partial content (для получения определённого диапазона, если был указан Range);

HEADER

- Accept-Ranges: bytes;
- Content-Range: bytes {начало фрагмента}-{конец фрагмента}/{полный размер сообщения}, например: Content-Range: bytes 1024-4095/8192, означает, что был предоставлен фрагмент с первого по четвертый килобайты из сообщения в 8 килобайт;
- Content-Length: {размер тела сообщения}, то есть передаваемого фрагмента, например: Content-Length: 1024, означает, что размер фрагмента один килобайт.

В BODY – запрашиваемый файл или диапазон.

В случае ошибок:

HTTP 404 – Not found.

BODY

```
{
  "HTTPStatus": 404,
  "ErrorCode":  "MESSAGE_NOT_FOUND",
  "ErrorMessage": "Невозможно найти сообщение с указанным id",
  "MoreInfo": {}
}
```

HTTP 404 – Not found

BODY

```
{
  "HTTPStatus": 404,
  "ErrorCode": "RECEIPT_NOT_FOUND",
  "ErrorMessage": "Невозможно найти квитанцию с указанным id",
  "MoreInfo": {}
}
```

HTTP 404 – Not found.

BODY

```
{
  "HTTPStatus": 404,
  "ErrorCode":  "FILE_NOT_FOUND",
  "ErrorMessage": "Невозможно найти файл с указанным id",
  "MoreInfo": {}
}
```

HTTP 404 – Not found

BODY

```
{
  "HTTPStatus": 404,
  "ErrorCode":  "FILE_TEMPORARY_NOT_AVAILABLE",
  "ErrorMessage": "Файлы сообщения временно недоступны",
  "MoreInfo": {
             "MissedFiles": [{
                              "Id": "string($uuid)"
                              "FileName": "string"
                              "RepositoriInfo":
                                         “RepositoryInfo”{...}
                               }]
               }
}
```

Где

- Id – уникальный идентификатор файла в формате UUID [16];
- FileName – имя временно недоступного файла;
- RepositoriInfo – информация о репозиториях (см. выше).

HTTP 410 – Gone.

BODY

```
{
  "HTTPStatus": 410,
  "ErrorCode":  "FILE_PERMANENTLY_NOT_AVAILABLE",
  "ErrorMessage": "Файлы сообщения более недоступны или задача не
предусматривает их хранения",
  "MoreInfo": {
             "MissedFiles": [{
                              "Id": "string($uuid)"
                              "FileName": "string"
                             }]
               }
}
```

Где

- Id – уникальный идентификатор файла в формате UUID [16];
- FileName – имя файла, без содержимого.

HTTP 416 – Range Not Satisfiable.

BODY

```
{
  "HTTPStatus": 416,
  "ErrorCode":  "INCORRECT_BYTE_RANGE",
  "ErrorMessage": "В запросе не верно указан диапазон байт",
  "MoreInfo": {}
}
```

3.1.5 Удаление сообщений

Удаление сообщений и файлов на стороне УИО осуществляется с использованием универсального REST-сервиса следующим образом:

3.1.5.1 Для удаления конкретного сообщения используется метод DELETE

DELETE: */messages/{msgId}

REQUEST

PATH

```
{
    "MsgId": "string($uuid)"
}
```

Где:

– MsgId – уникальный идентификатор сообщения в формате UUID [16].

RESPONSE

HTTP 200 – Ok;

В случае ошибок:

HTTP 403 – Forbidden

BODY

```
{
  "HTTPStatus": 403,
  "ErrorCode":  "MESSAGE_DELETE_ERROR",
  "ErrorMessage": "Удалить можно только исходящее ЭС в статусах Ошибка, Отклонено, Зарегистрировано, Доставлено, Черновик или прочитанное входящее ЭС, если Текущая дата - Дата создания ЭС => 1 год.",
  "MoreInfo": {}
}
```

HTTP 404 – Not found.

BODY

```
{
  "HTTPStatus": 404,
  "ErrorCode":  "MESSAGE_NOT_FOUND",
  "ErrorMessage": "Невозможно найти сообщение с указанным id",
  "MoreInfo": {}
}
```

3.1.5.2 Для удаления конкретного файла или отмены сессии отправки используется метод DELETE

DELETE: */messages/{msgId}/files/{fileId}

REQUEST

PATH

```
{
     "MsgId": "string($uuid)",
     "FileId": "string($uuid)"
}
```

Где:

– MsgId – уникальный идентификатор сообщения в формате UUID [16];

– FileId – уникальный идентификатор файла в формате UUID [16].

RESPONSE

HTTP 200 – Ok;

В случае ошибок:

HTTP 403 – Forbidden

BODY

```
{
  "HTTPStatus": 403,
  "ErrorCode": "MESSAGE_DELETE_ERROR",
  "ErrorMessage": "Удалить можно только исходящее ЭС в статусах
Ошибка, Отклонено, Зарегистрировано, Доставлено, Черновик или
прочитанное входящее ЭС, если Текущая дата - Дата создания ЭС => 1
год.",
  "MoreInfo": {}
}
```

HTTP 404 – Not found.

BODY

```
{
  "HTTPStatus": 404,
  "ErrorCode":  "MESSAGE_NOT_FOUND",
  "ErrorMessage": "Невозможно найти сообщение с указанным id",
  "MoreInfo": {}
}
```

HTTP 404 – Not found.

BODY

```
{
  "HTTPStatus": 404,
  "ErrorCode": "FILE_NOT_FOUND",
  "ErrorMessage": "Невозможно найти файл с указанным id",
```

```
  "MoreInfo": {}
}
```

3.1.6 Для получения справочной информации

3.1.6.1 Для получения справочника задач используется метод GET

GET: */tasks

Запрос может быть дополнен запросом на направление обмена по задачам - GET: */task?direction={d}, где {d} – направление обмена. Допустимые значения: 0/1/2. 0- входящие (БР->ЛК); 1-исходящие (ЛК->БР); 2- двунаправленные (ЛК->ЛК). Если параметр не указан, возвращается все задачи. В случае некорректного указания параметра – ошибка.

QUERY

```
{"Direction": "string"}
```

RESPONSE

HTTP 200 – Ok.

```
[
  {
    "Code": "string",
    "Name": "string",
    "Direction": "string",
    "AllowLinkedMessages": "boolean",
    "AllowAspera": "boolean",
    "Description": "string"
  }
]
```

Где:

а) Code – код задачи (по справочнику задач в формате "Zadacha_*", где Zadacha_ - неизменная часть, * - число/набор символов определяющий порядковый номер/обозначение задачи), используется для идентификации задачи;

б) Name – наименование задачи;

в) Direction – направление обмена. Может принимать значения:

1) inbox - входящее в ЛК;

2) outbox - исходящее из ЛК;

3) bidirectional - двунаправленное между ЛК.

г) AllowLinkedMessages – признак возможности отправки связанных сообщений;

д) AllowAspera – признак возможности отправки сообщений через Aspera;

е) Description – текстовое описание задачи, может быть не заполнено.

В случае ошибок:

HTTP 400 – Bad Request.

BODY

```
{
  "HTTPStatus": 400,
  "ErrorCode":  "DIRECTION_INCORRECT",
  "ErrorMessage": " Неверное значение параметра direction,
(допустимо 0,1,2)",
  "MoreInfo": {}
}
```

3.1.6.2 Для получения информации о своём профиле используется метод GET

GET: */profile

RESPONSE

HTTP 200 – Ok.

```
{
  "ShortName": "string",
  "FullName": "string",
  "Activities": [
    {
      "FullName": "string",
      "ShortName": "string",
      "SupervisionDevision": {
        "Name": "string"
      }
    }
  ],
  "Inn": "integer",
  "Ogrn": "integer",
  "InternationalId": "integer",
  "Opf": "string",
  "Email": "string",
  "Address": "string",
  "Phone": "string",
  "CreationDate": "string",
  "Status": "string"
}
```

Где:

- ShortName – краткое наименование компании;
- FullName – полное наименование компании;
- Activities – список видов деятельностей компании;
  - FullName – полное наименование вида деятельности;
  - ShortName – краткое наименование вида деятельности;
  - SupervisionDevision – поднадзорное подразделение:
  - Name – наименование поднадзорного подразделения Банка России;
- Inn – ИНН компании;
- Ogrn – ОГРН компании;
- InternationalId – международный идентификатор (необязательное, зарезервированное поле);
- Opf – организационно-правовая форма компании;
- Email – электронный адрес компании;
- Address – почтовый адрес компании;
- Phone – контактный телефон компании;
- CreationDate – дата создания ЛК компании;
- Status – текущий статус ЛК компании.

3.1.6.3 Для получения информации о квоте профиля используется метод GET:

GET: */profile/quota

RESPONSE

HTTP 200 – Ok.

```
{
    "TotalQuota": "integer",
    "UsedQuota": "integer",
    "MessageSize": "integer"
}
```

Где:

- TotalQuota – информация о доступной квоте в байтах. Имеет формат int64 (т.е. signed 64 bits);
- UsedQuota – информация об использованной квоте в байтах. Имеет формат int64 (т.е. signed 64 bits).
- MessageSize – информация о максимальном размере сообщения в байтах. Имеет формат int64 (т.е. signed 64 bits).

3.1.6.4 Для получения информации о технических оповещениях используется метод GET:

GET: */notifications

RESPONSE

HTTP 200 – Ok.

```
[
  {
    "Text": "string",
    "Date": "string"
  }
]
```

Где для каждого элемента массива:

- Text – текст технического оповещения;
- Date – Дата и время технического оповещения.

3.1.6.5 Для получения списка справочников используется метод GET

GET: */dictionaries

RESPONSE

HTTP 200 – Ok.

```
[
  {
    "Id": "string($uuid)",
    "Text": "string",
    "Date": "string"
  }
]
```

Где:

- Id – уникальный идентификатор справочника, используется для идентификации задачи в формате UUID [16];
- Text – текстовое наименование справочника;
- Date – дата последнего обновления справочника.

Массив содержит информацию только о справочниках, к которым у пользователя есть доступ.

3.1.6.6 Для получения записей конкретного справочника, но не более 100 записей за один запрос, используется метод GET

GET: */dictionaries/{dictId}

Запрос может быть дополнен запросом на страницу со списком массива записей справочника в виде GET:*/dictionaries/{dictId}?page={n}, где {n} – страница (n-я сотня записей запрошенного справочника). Допустимые значения: n>0 (положительные целые числа, больше 0). Если запрос страницы не указан, возвращается первая страница записей справочника. В случае некорректного {n} – ошибка.

REQUEST

PATH

```
{
     "dictId": "string($uuid)"
}
```

Где:

– dictId – уникальный идентификатор справочника в формате UUID [16].

PATH

```
{
     "page": "integer"
}
```

Где:

page – номер страницы справочника в разбивке по 100 записей (если не задан, от вернутся первые 100 записей).

Примечание: Запрещается запрашивать справочник, предварительно не убедившись в том, что он был обновлён (при запросе списка справочников методом, описанным в пункте 3.1.6.5 значение поля Date изменилось на актуальную дату).

RESPONSE

В случае корректного запроса:

HTTP 200 – OK. В случае корректного запроса в теле ответа возвращается объект, состоящий из массива записей запрашиваемого справочника и информации о пагинации (разбиении на страницы).

BODY

```
{
    Items: [<записи справочника>]
    PaginationInfo:
```

```
    {
         "TotalRecords" : "integer",
         "TotalPages" : "integer",
         "CurrentPage" : "integer",
         "PerCurrentPage" : "integer",
         "PerNextPage" : "integer",
         "MaxPerPage" : "integer"
    }
}
```

Где:

Items: [] – массив записей справочника, в зависимости от его структуры. В массиве возвращаются записи справочника со статусом не равным «удален», не более 100 за один запрос;

PaginationInfo - объект с информацией о пагинации, содержащий:

totalRecords - всего записей в справочнике;

totalPages - всего страниц с разбивкой не более 100 записей на странице;

currentPage - текущая страница (соответствует n из запроса page={n});

perCurrentPage - записей на текущей странице или null, если запрошенная страница не существует;

perNextPage - записей на следующей страница или null, если страница не существует;

maxPerPage- максимальное количество записей на странице (всегда 100).

Дополнительно информация о пагинации дублируется в заголовке ответного сообщения, с каждым положительным ответом (HTTP 200),

HEADER

```
"EPVV-Total": "integer"
"EPVV-TotalPages": "integer"
"EPVV-CurrentPage: "integer"
"EPVV-PerCurrentPage: "integer"
"EPVV-PerNextPage: "integer"
```

EPVV-Total – общее количество сообщений в запросе;

EPVV-TotalPages – общее количество страниц, охватывая все сообщения по 100 сообщений на странице;

EPVV-CurrentPage – номер текущей страницы;

EPVV-PerCurrentPage – количество сообщений на текущей странице;

EPVV-PerNextPage – количество сообщений на следующей странице.

В случае некорректного (ошибочного запроса):

HTTP 400 – Bad Request

```
{
  "HTTPStatus": 400,
  "ErrorCode":  "INCORRECT_PAGE_NUM",
  "ErrorMessage": "Произошла ошибка. Некорректное значение
страницы: {page}",
  "MoreInfo": {
               "TotalItems": “integer”,
               "TotalPages": “integer”
}}
```

Где:

TotalItems - всего элементов в массиве;

TotalPages - всего станиц.

HTTP 404 – Not found.

BODY

```
{
  "HTTPStatus": 404,
  "ErrorCode":  "DICTIONARY_NOT_FOUND",
  "ErrorMessage": "Справочник не найден",
  "MoreInfo": {}
}
```

В случае если запрошен некорректный (несуществующий) идентификатор справочника dictId;

HTTP 403 – Forbidden / Запрещено.

BODY

```
{
  "HTTPStatus": 403,
  "ErrorCode":  "DICTIONARY_FORBIDDEN",
  "ErrorMessage": "Доступ к справочнику запрещен",
  "MoreInfo": {}
}
```

В случае если у пользователя нет прав на просмотр справочника, с указанным идентификатором справочника dictId.

3.1.6.7 Для скачивания конкретного справочника в виде файла используется метод GET:

GET: */dictionaries/{dictId}/download

REQUEST

PATH

```
{
   "dictId": "string($uuid)"
}
```

Где:

– dictId – уникальный идентификатор справочника в формате UUID [16].

RESPONSE

В случае корректного запроса:

– HTTP 200 – Ok

В случае успешного ответа возвращается двоичный поток вида application/octet-stream, содержащий zip-архив с двумя файлами в формате xml. Один файл содержит описание структуры справочника, второй - данные запрошенного справочника. В файле данных возвращаются все записи справочника со статусом не равным «удален». Xsd-схемы xml-файлов справочников определены в Приложении И документа [].

В случае ошибок:

HTTP 404 – Not found.

BODY

```
{
  "HTTPStatus": 404,
  "ErrorCode":  "DICTIONARY_NOT_FOUND",
  "ErrorMessage": "Справочник не найден",
  "MoreInfo": {}
}
```

В случае если запрошен некорректный (несуществующий) идентификатор справочника dictId;

HTTP 403 – Forbidden / Запрещено

BODY

```
{
  "HTTPStatus": 403,
```

```
  "ErrorCode":  "DICTIONARY_FORBIDDEN",
  "ErrorMessage": "Доступ к справочнику запрещен",
  "MoreInfo": {}
}
```

В случае если у пользователя нет прав на просмотр справочника, с указанным идентификатором справочника dictId.

3.2 Взаимодействие с использованием сервиса REST-УТА

3.2.1 Описание сервиса REST-УТА

3.2.1.1 Принципы взаимодействия

Инициатором электронного обмена может быть как КО, так и Банк России. Прием информации от КО должен осуществляться Порталом "Биврёст" с использованием REST-сервиса. В качестве транспортного адаптера при этом должно использоваться специальное программное обеспечение файлового взаимодействия Банка России (СПО УТА).

3.2.1.2 Общие правила оформления сообщений, передаваемых по протоколу HTTP 1.1

СЛЕДУЕТ задавать абсолютный URL в параметрах методов POST и GET.

Поле Accept СЛЕДУЕТ заполнять значениями «application/soap+xml», «application/json».

Поле User-Agent СЛЕДУЕТ заполнять строковым значением, идентифицирующим ПО, которое используется для взаимодействия с Внешним порталом.

Слово «СЛЕДУЕТ» используется для указания того, что данное требование спецификации должно быть обеспечено, если этому не препятствуют серьезные причины. Данное правило соответствует требованию к реализации протокола, определенное в RFC 2119.

3.2.1.3 Авторизация

Для осуществления информационного взаимодействия необходимо пройти авторизацию в Портале "Биврёст". Авторизация осуществляется с помощью передачи HTTP заголовка «Authorization». Тип авторизации – «Basic».

В качестве логина передается доменная учетная запись, которая имеет вид: «DOMAIN\user». Логический адрес и права отправителя определяются согласно указанной доменной учетной записи абонента.

3.2.1.4 Отправка данных

Прием данных от КО на стороне Портала "Биврёст" осуществляется с использованием

POST: http://<host>:<port>/rapi2/outbox/data

REQUEST

```
{
  "to" : [ "sds-smtp", "sds-file" ],
  "fileName" : "super-puper-file.txt",
  "data" :
"0JXRgdC70Lgg0LLRiyDRjdGC0L4g0L/RgNC+0YfQuNGC0LDQu9C4LCDRgtC+INCy0
Ysg0LzQvtC70L7QtNGG0Ysh",
  "taskName" : "TestForCrypt"
}
```

где

- to - массив логических адресов получателей;
- fileName - имя передаваемого блока данных, если потребуется создать файл при передаче - это будет его имя;
- data - base64 закодированный блок передаваемых данных;
- taskName - название задачи в рамках которой будет производиться передача.

RESPONSE

```
{
  "messageId" : "86e1fc11-679a-4cdb-a990-0e435e0c5bad",
  "status" : "OK"
}
```

где

- messageId - идентификатор переданного на обработку сообщения в формате UUID [16].

3.2.2 Отправка файла

Отправка ЭС, содержащего файл, осуществляется в три этапа.

3.2.2.1 I этап

POST: http://<host>:<port>/rapi2/outbox/files

REQUEST

```
{
  "size" : 112839389
}
```

где

- size - размер передаваемого файла в байтах.

RESPONSE

```
{
  "messageId" : "79a274a8-4b11-40a2-be93-7fb839cc7a3d",
  "status" : "OK"
}
```

где

- messageId - идентификатор сессии отправки файла в формате UUID [16], его следует использовать в качестве {id} далее в методах 2 и 3.

3.2.2.2 II этап

PUT: http://<host>:<port>/rapi2/outbox/files/{id}

REQUEST

```
{
  "chunk" :
"0JAg0YPQtiDQtdGB0LvQuCDQuCDRjdGC0L4g0L/RgNC+0YfQuNGC0LDQu9C4LCDRg
tC+INGB0L7QstGB0LXQvCDQvNC+0LvQvtC00YbRiyE="
}
```

RESPONSE

```
{
  "status" : "OK"
}
```

3.2.2.3 III этап

POST: http://<host>:<port>/rapi2/outbox/files/{id}

REQUEST

```
{
  "to" : [ "sds-smtp", "sds-file" ],
  "fileName" : "super-puper-file.txt",
  "taskName" : "TestForCrypt"
}
```

где

- to - массив логических адресов получателей;
- fileName - имя передаваемого файла;

– taskName - название задачи в рамках которой будет производиться передача.

RESPONSE

```
{
  "messageId" : "86e1fc11-679a-4cdb-a990-0e435e0c5bad"
  "status" : "OK"
}
```

где

– messageId - уникальный идентификатор переданного на обработку сообщения.

3.2.3 Получение сообщений

3.2.3.1 Получение списка сообщений

GET: http://<host>:<port>/rapi2/inbox/messages

RESPONSE

```
{
  "ids" : [ "493451f7-5dc7-4004-a831-88b18139b8b9", "f730c6bb-
30e0-4bc6-b21e-d8fe82662c70" ],
  "status" : "OK"
}
```

где

– ids - уникальный идентификатор ожидающих приема сообщений в формате UUID [16], их следует принимать в качестве {id} в методах ниже.

3.2.3.2 Получение метаданных сообщения

GET: http://<host>:<port>/rapi2/inbox/messages/{id}

RESPONSE

```
{
  "from" : "sds-file",
  "to" : [ "ext-smtp", "ext-file" ],
  "fileName" : "super-puper-file.txt",
  "taskName" : "TestForCrypt",
  "status" : "OK"
}
```

где

- from - логический адрес отправителя сообщений;
- to - массив логических адресов получателей сообщения;
- fileName - имя передаваемого в сообщении файла или блока данных;
- taskName - имя задачи в рамках которой происходила передача сообщения.

3.2.3.3 Получение тела сообщения (всего сразу одним файлом)

GET: http://<host>:<port>/rapi2/inbox/messages/{id}/data

RESPONSE

```
File как application/octet-stream
```

3.2.4 Получение сообщения частями

3.2.4.1 Получение количества частей

GET: http://<host>:<port>/rapi2/inbox/messages/{id}/parts/{chunkSize}

где

- id - уникальный идентификатор сообщения;
- chunkSize - желаемый размер части на которые стоит разбивать файл.

RESPONSE

```
{
  "chunkCount" : 10,
  "chunkSize" : 12345
}
```

где

- chunkCount - количество частей в разбиении;
- chunkSize - размер части (повторяет переданный в запросе chunkSize).

ВАЖНО! Размер последней части может не равняться chunkSize и быть меньше, т.к. не всегда файл ровно разделяется по указанному размеру.

3.2.4.2 Получение части

GET: http://<host>:<port>/rapi2/inbox/messages/{id}/parts/{chunkSize}/{part}

где

- id - уникальный идентификатор сообщения в формате UUID [16];
- chunkSize - желаемый размер части на которые стоит разбивать файл;
- part - номер части от 0 до chunkCount - 1 включительно (нумерация не с 1, а с 0).

RESPONSE

```
Chunk как application/octet-stream
```

ВАЖНО! Если вы передали part >= chunkCount то ошибки не произойдет, вам будет просто возвращен chunk размером 0 байт. Это также можно использовать как индикатор окончания передачи, чтобы не вызывать метод в разделе I. Т.е. вы можете реализовать загрузку файла кусочками следующим образом:

```
while ( ( chunk = fetch(part++) ).size != 0 )
{
  // do something
}
```

3.2.5 Удаление сообщения

DELETE: http://<host>:<port>/rapi2/inbox/messages/{id}

RESPONSE

```
{
  "status" : "OK"
}
```

где

- id - уникальный идентификатор сообщения в формате UUID [16];

4 Защита передаваемой информации

4.1 Криптографическая обработка информации

Для обеспечения криптографической обработки информации, передаваемой из Клиентского ПО в Портал "Биврёст", на стороне Портала используется СКЗИ сертифицированное ФСБ РФ. Для передачи данных, прошедших криптографическую обработку, между внешними абонентами и Порталом "Биврёст" используется формат PKCS7#.

Для обеспечения криптографической обработки информации, передаваемой (получаемой) в (из) Банк(а) России посредством Портала "Биврёст" на стороне внешнего абонента должны присутствовать СКЗИ, удовлетворяющие следующим требованиям:

- требованиям Приказа ФСБ от 27 декабря 2011 г. № 796 «Об утверждении требований к средствам электронной подписи и требований к средствам удостоверяющего центра»;
- обладающие сертификатом ФСБ.

При криптографической защите информации на стороне абонент должны использоваться как блочные, так и поточные методы шифрования данных как при получении, так и направлении данных в Портал "Биврёст".

4.2 Условия взаимодействия с ВП ЕПВВ

В обработку принимаются сообщения, для которых выполняются все перечисленные критерии:

а) сообщение зашифровано потоковым методом или имеет размер менее 256 килобайт;

б) криптографические операции должны выполняться с учетом следующих требований:

- длина секретного ключа отправителя должна составлять 256 бит;
- длина открытого ключа получателя должна быть 512 бит;

в) для потоков, требующих подписание, сообщение подписано не менее чем одной верной УКЭП, одна из которых должна соответствовать ЛК по параметру ИНН. УКЭП считается верной, если она криптографически верна, сертификат подписавшего не истек и отсутствует в актуальном на момент проверки СОС;

г) в контейнере подписи содержится собственный сертификат подписавшего в

кодировке DER.

5 Параметры подключения внешних абонентов к Внешнему порталу

5.1 Подключение к внешнему порталу с использованием REST-сервиса

Для подключения к Порталу "Биврёст" посредством унаследованного REST-сервиса (не рекомендуется использовать) необходимо в Клиентском ПО отправителя указать следующую ссылку: https://portal5.cbr.ru/service-reporting-api/*

Для подключения к Порталу "Биврёст" посредством универсального REST-сервиса (рекомендуется использовать) необходимо в Клиентском ПО отправителя указать следующую ссылку: https://portal5.cbr.ru/back/rapi2/*

Вместо звездочки необходимо указать методы, описанные в приложении к настоящему документу (3.2.1).

Спецификация REST-сервиса может быть изменена, обратная совместимость (версионность) не гарантируется.

5.2 Подключение к внешнему порталу с использованием протокола FASP

Организация взаимодействия внешних абонентов с сервером данных по протоколу FASP возможна с использованием следующих методов:

- Графический клиент IBM Aspera Client;
- Утилита командной строки IBM Aspera CLI;
- Наборы библиотек API IBM FASP Stream SDK и IBM FASP Manager SDK (поддерживаемые Python, Java, C#, GO, C++ - Mac Intel, C++ - Windows x86 (VS-2015), C++ - Windows x64 (VS-2015)).

Описание API IBM FASP Stream SDK и IBM FASP Manager SDK и библиотеки можно скачать с официального сервера разработчика developer.asperasoft.com.

5.3 Условия взаимодействия с ВП ЕПВВ

В обработку Порталом "Биврёст" принимаются сообщения, для которых выполняются перечисленные в п.п. 4.2 критерии и выполняются требования к имени файлов: имена всех файлов не должны содержать символы, отличные от латиницы, цифр и символов "." и не должны быть длиннее 64 символов с учётом расширений, если иное не оговорено в документации на соответствующий поток.

6 Перечень кодов ошибок при обработке сообщений

Перечень кодов и описания обработки приведен в таблице (Таблица 1).

Таблица 1 – Перечень кодов обработки сообщений подсистемой «Внешний шлюз»

Код обработки	Описание
0000	Сообщение успешно обработано
1001	Сообщение не будет обработано, поскольку не соответствует требуемой структуре.
1003	Сообщение не будет обработано, поскольку его формат не поддерживается Транспортным шлюзом ЕПВВ.
1005	Сообщение не будет обработано, поскольку архив сформирован неверно.
1006	Сообщение не будет обработано, поскольку служебный конверт не соответствует требуемой структуре.
1007	Сообщение не будет обработано, поскольку служебный конверт заполнен некорректно.
3001	Сообщение не будет обработано, поскольку не соответствует регламенту.
3003	Сообщение не будет обработано, поскольку заражено вирусом.
3004	Сообщение не будет обработано, поскольку адресная информация указана некорректно.
3008	Выявлена угроза при проверке файла модулем контентного анализа. Файл перемещен в карантин.
3010	Сообщение не будет обработано, так как не соответствует требуемой схеме XML сообщения.
3011	Требуемый документ (Charge, ExportRequest, PacketUNIFO) в полученных данных не найден
4001	Произошла ошибка при расшифровании сообщения.
4002	Произошла ошибка при проверке подписи в сообщении.
4003	Произошла ошибка при зашифровании сообщения.
4004	Произошла ошибка при снятии электронной подписи в сообщении.

7 Коды ошибок ПП Универсальный REST-сервис ЕПВВ

Перечень кодов и описания обработки приведен в таблице (Таблица 2).

Таблица 2 – Перечень кодов ошибок ПП Универсальный REST

HTTP STATUS	EPVV_ERROR	TEXT	Запрос
400	COMMON_ERROR	Общая ошибка запроса	GET /messages
400	CONTENT_LENGTH_INCORRECT	Параметр content-length не соответствует размеру входящих данных	PUT /messages/{msgId}/files/{fileId}
400	CONTENT_LENGTH_NOT_SET	Не указан параметр content-length	PUT /messages/{msgId}/files/{fileId}
400	CONTENT_RANGE_INCORRECT	Не указан или указан неверно параметр content-range	PUT /messages/{msgId}/files/{fileId}
400	DATA_ALLREADY_WRITTEN	Данные уже записаны	PUT /messages/{msgId}/files/{fileId}
400	FILE_ALLREADY_LOADED	Файл уже загружен	PUT /messages/{msgId}/files/{fileId} POST /messages/{msgId}/files/{fileId}/createUploadSession
400	DATA_RANGE_SAVE_ERROR	Ошибка сохранения участка данных	PUT /messages/{msgId}/files/{fileId}
400	DATA_UNREADABLE	Не удалось прочитать данные, убедитесь что контент задан корректно!	PUT /messages/{msgId}/files/{fileId}
400	DIRECTION_INCORRECT	Неверное значение параметра direction (допустимо 0,1,2)	GET /tasks
400	DICTIONARY_DATA_ERROR	Ошибка при получении данных справочника {dictid}	GET /dictionaries/{dictId}/download
400	REQUEST_AUTHOR_NOT_SET	Не удалось извлечь автора запроса	DELETE /messages/{msgId}

HTTP STATUS	EPVV_ERROR	TEXT	Запрос
400	REQUEST_PLAYLOD_INCORRECT	Неправильное тело запроса	POST /messages
400	INCORRECT_PAGE_NUM	Произошла ошибка. Некорректное значение страницы: {page}	GET /dictionaries/{dictId} GET /messages
400	INCORRECT_REQUEST_PARAM	Некорректное значение параметра запроса	GET /messages
400	TASK_CODE_MUST_BE_SENT	Должен быть передан параметр Task	PUT: /messages/{msgId}/files/{fileId}
400	FILE_SIZE_NOT_MATCH_DB	Размер файла не соответствует записи из базы данных	POST /messages
401	ACCOUNT_NOT_FOUND	Аккаунт не найден	POST /messages GET /profile GET /quote
403	DICTIONARY_FORBIDDEN	Доступ к справочнику запрещен	GET /dictionaries/{dictId}
403	MESSAGE_DELETE_ERROR	Удалить можно только исходящее ЭС в статусах ошибка, отклонено, зарегистрировано, доставлено, черновик или прочитанное входящее ЭС, если текущая дата - дата создания ЭС => 1 год	DELETE /messages/{msgId} DELETE /messages/{msgId}}/files/{fileId}
404	DICTIONARY_NOT_FOUND	Справочник не найден	GET /dictionaries/{dictId} GET /dictionaries/{dictId}/download
404	FILE_NOT_FOUND	Невозможно найти файл с указанным id	POST /messages/{msgId}/files/{fileId}/createUploadSession GET /messages/{msgId}/files/{fileId} PUT /messages/{msgId}/files/{fileId} GET /messages/{msgId}/files/{fileId}/d

HTTP STATUS	EPVV_ERROR	TEXT	Запрос
			ownload GET /messages/{msgId}/receipts/{rcptId}/files/{fileId} GET /messages/{msgId}/receipts/{rcptId}/files/{fileId}/download
404	FILE_TEMPORARY_NOT_AVAILABLE	Файл сообщения временно недоступен	GET /messages/{msgId}/download (прим.: если для всех файлов в сообщении) GET /messages/{msgId}/files/{fileId}/download
404	MESSAGE_NOT_FOUND	Невозможно найти сообщение с указанным id	GET /messages/{msgId} DELETE /messages/{msgId} POST /messages/{msgId} GET /messages/{msgId}/download POST /messages/{msgId}/files/{fileId}/createUploadSession GET /messages/{msgId}/files/{fileId} PUT /messages/{msgId}/files/{fileId} GET /messages/{msgId}/files/{fileId}/download GET /messages/{msgId}/receipts GET /messages/{msgId}/receipts/{rcptId} GET /messages/{msgId}/receipts/{rcptId}/files/{fileId} GET /messages/{msgId}/receipts/{rcptId}/files/{fileId}/download
404	RECEIPT_NOT_FOUND	Невозможно найти квитанцию с указанным id	GET /messages/{msgId}/receipts/{rcptId} GET /messages/{msgId}/receipts/{rcptId}/files/{fileId} GET /messages/{msgId}/receipts/{rcptId

HTTP STATUS	EPVV_ERROR	TEXT	Запрос
			}/files/{fileId}/download
405	BASE_REQUEST_ADDRESS_NOT_FOUND	Не найден базовый адрес запроса	POST /messages/{msgId}/files/{fileId}/createUploadSession
405	NOT_ALLOWED_FOR_ASPERA_REPO	Для файла указано RepositoryType = Aspera, он не может быть загружен через HTTP	POST /messages/{msgId}/files/{fileId}/createUploadSession
406	MESSAGE_SENT_ERROR	Сообщение не может быть отправлено	POST /messages POST /messages/{msgId}
406	FILE_SIZE_ERROR	Размер файла {file.name} должен быть Должен быть в диапазоне от 1 до 9223372036854775807 байт	POST /messages
406	DUPLICATE_FILE_NAME	Имена файлов не должны повторяться	POST /messages
406	FILE_ENCRYPTION_FLAG_MUST_BE_SET	Для файла {requestfile.name} должен быть указан флаг шифрования.	POST /messages
406	REQ_FILE_EXTENSION_ERROR	Файл {requestfile.name} с указанным флагом шифрования должен иметь расширение .enc.	POST /messages
406	SIGN_FILE_EXTENSION_ERROR	Файл подписи {sigfile.name} должен иметь расширение \'.sig\’	POST /messages
406	INCORRECT_RECEIVER	Получатель должен быть КО	POST /messages
406	RECEIVER_NOT_SET	Не определен получатель	POST /messages
406	SEND_BY_THIS_TASK_NOT_ALLOWED	Не доступна отправка сообщения по указанной задаче.	POST /messages
406	SIGN_FILE_NOT_FOUND	Не найден файл для подписи {sigfile.name}.	POST /messages

HTTP STATUS	EPVV_ERROR	TEXT	Запрос
406	INVALID_FILE_EXTENSION	Недопустимое расширение файла для данной задачи	POST /messages
410	FILE_PERMANENTLY_NOT_AVAILABLE	Файл сообщения более недоступен или задача не предусматривает его хранения	GET /messages/{msgId}/download (прим.: если для всех файлов в сообщении) GET /messages/{msgId}/files/{fileId}/download
413	ACCOUNT_QUOTA_EXCIDED	Сообщение не может быть отправлено, так как оставшаяся квота хранения истории обмена ЭС будет превышена.	POST /messages
413	MESSAGE_QUOTA_EXCIDED	Сообщение не может быть отправлено, так как размер ЭС превышает квоту	POST /messages
416	INCORRECT_BYTE_RANGE	В запросе не верно указан диапазон байт	GET /messages/{msgId}/download GET /messages/{msgId}/files/{fileId}/download GET /messages/{msgId}/receipts/{rcptId}/files/{fileId}/download
422	INCORRECT_BODY_PARAM	Неверные параметры в теле запроса. Проверьте сообщение на соответствие параметрам задачи	POST /messages POST /messages/{msgId}
422	INCORRECT_CORRELATION_ID	Не найдено сообщение, которое должно соответствовать переданному CorrelationId	POST /messages

Ссылочные документы

1 RFC2119: Key words for use in RFCs to Indicate Requirement Levels. (http://www.ietf.org/rfc/rfc2119.txt).

2 W3C SOAP Version 1.2, W3C Candidate Recommendation 19 December 2002 (http://www.w3.org/TR/soap12-part0; http://www.w3.org/TR/soap12-part1; http://www.w3.org/TR/soap12-part2).

3 Унифицированные форматы электронных банковских сообщений. Структура и правила заполнения заголовков служебного конверта.

4 RFC2821: Simple Mail Transfer Protocol (http://www.ietf.org/rfc/rfc2821.txt).

5 RFC2822 : Internet Message Format (http://www.ietf.org/rfc/rfc2822.txt).

6 RFC3023: XML Media Types (http://www.ietf.org/rfc/rfc3023.txt).

7 RFC2616: Hypertext Transfer Protocol – HTTP/1.1 (http://www.ietf.org/rfc/rfc2616.txt).

8 RFC1939: Post Office Protocol – Version 3 (http://www.ietf.org/rfc/rfc2616.txt).

9 WebSphere MQ Information Center V1.2. WebSphere MQ Application Programming Guide, © Copyright International Business Machines Corporation 1994, 2002. All rights reserved.

10 «Direct Internet Message Encapsulation» (http://msdn.microsoft.com/library/en-us/dnglobspec/html/draft-nielsen-dime-02.txt).

11 GZIP file format specification version 4.3, May 1996.

12 RFC3501: INTERNET MESSAGE ACCESS PROTOCOL – VERSION 4rev1 (www.ietf.org/rfc/rfc3501.txt).

13 RFC1731: IMAP4 Authentication Mechanisms (www.ietf.org/rfc/rfc1731.txt).

14 RFC2618: HTTP Over TLS (http://www.ietf.org/rfc/rfc2618.txt).

15 RFC1321: The MD5 Message-Digest Algorithm (http://www.ietf.org/rfc/rfc1321.txt).

16 RFC4122: A Universally Unique IDentifier (UUID) URN Namespace (https://tools.ietf.org/rfc/rfc4122.txt).

17 ЦБРФ.62.0.39683.ТУ.02-1 Технические условия внутреннего обмена