**FAC Cash Exchange Notes on Integration with AMQP**

For an overview of Cash Exchange workflows and integration requirements please refer to the separate Cash Exchange Operating Model document.

The FAC Exchange and Third Party (TTP) API uses amqp v1.0 protocol. ActiveMQ Artemis is used for the broker. **Queues:**

The exchange has a sender for queue called “main-queue” and a receiver for queue called “response-queue”.Accordingly, the TTP should have a receiver for queue called “main-queue” and a sender for queue called “response-queue”.

**Messages:**

Note: Each message should adhere to the json format, and contain [username] of the connected user in the [userId] field. Otherwise, the broker plug-in will reject the message.

For vertx:



**Tokens request process:**

1. The exchange creates and sends issuance/redemption requests to the TTP

Note: the Exchange keeps all the requests in a PENDING status on its side until a confirmation message from the TTP is received with Approved or Rejected status in response.

1. To send a request to the TTP, the exchange creates the TTPTransferIssue model for the request, and fills parameters:

* messageId - random identifier (must be different for each message),
* transactionId - request identifier (= requestId from state),
* previousMessageId - null for the first message,
* timestamp - current time,
* exchangeId - exchange identifier (exchange node X500name)
* ttpId - ttp identifier (null at the moment),
* requestorId - issuer identifier (issuer node X500name at the moment),
* requestAction - ISSUANCE or REDEMPTION,
* quantity - tokens quantity,
* tokenIdentifier - token identifier (“GBP” for example),
* paymentReference - payment reference (random identifier),
* null in other fields;

1. The exchange sends this message to the broker and saves the message;
2. The TTP receives message from the broker

*(*

*In the current Web app which is emulating the TTP:*

* *The TTP saves received message and determines its type;*
* *For TTPTransferIssue, the TTP creates a transaction in the database and waits for user decision;*
* *If user rejects or approves transactions, the TTP changes transaction status in the database, creates a copy of the request and changes some fields (see below)*

*)*

1. The TTP creates a copy of the request and changes some fields, where:

* messageId - new random identifier,
* previousMessageId - request message’s identifier,
* timestamp - current time,
* responseAction - user’s decision;

1. TTP sends the message to the broker, saves it;
2. The exchange receives the message, saves it, determines its type;
3. The exchange updates the request state depending on the user’s decision.

**Balance request process (exchange from ttp):**

1. The exchange creates the TTPBalanceIssue model for the request, and fills parameters:

* messageId - random identifier (must be different for each message),
* previousMessageId - null for the first message,
* timestamp - current time,
* exchangeId - exchange identifier (exchange node X500name),
* ttpId - ttp identifier (null at the moment),
* requestorId - issuer identifier (exchange node X500name at the moment),
* startTime – start point of the time period for calculating summary (null/not implemented at the moment),
* endTime – end point of the time period for calculating summary (null/not implemented at the moment),
* subType – parameter to filter transactions by sub type (issuance or redemption),
* requestStatus – parameter to filter transactions by status (completed, rejected or pending)
* tokenIdentifier - token identifier (“GBP” for example),
* null in other fields;

1. The exchange creates new receiver for queue with name “TTPBalanceIssue[messageId]”;
2. The exchange sends this message to the broker, saves message;
3. The TTP receives message from the broker, saves it, determines its type;

*(*

*In the current Web app which is emulating the TTP:*

* *The TTP saves received message and determines its type;*
* *For TTPBalanceIssue the TTP counts summary (of issued and redeemed tokens);*

*)*

1. The TTP creates a copy of the request and changes some fields, where:

* messageId - new random identifier,
* previousMessageId - request message’s identifier,
* timestamp - current time,
* issued - quantity of issued tokens,
* redeemed - quantity of redeemed tokens;

1. The TTP creates new sender for queue with name “TTPBalanceIssue[messageId]”;
2. The TTP sends the message to the broker, saves it;
3. The exchange receives the message, saves it, determines its type;
4. The exchange gets a summary (of issued and redeemed tokens).

**Balance request process (ttp from exchange):**

1. The TTP creates the TTPBalanceIssue model for the request, and fills parameters:

* messageId - random identifier (must be different for each message),
* previousMessageId - null for the first message,
* timestamp - current time,
* exchangeId - exchange identifier ((exchange node X500name),
* ttpId - ttp identifier (null at the moment),
* requestorId - issuer identifier (hardcode name at the moment),
* startTime – start point of the time period for calculating summary (null/not implemented at the moment),
* endTime – end point of the time period for calculating summary (null/not implemented at the moment),
* subType – parameter to filter transactions by sub type (null at the moment),
* requestStatus – parameter to filter transactions by status (completed at the moment),
* tokenIdentifier - token identifier (“GBP” for example),
* null in other fields;

1. The TTP creates new receiver for queue with name “TTPBalanceIssue[messageId]”;
2. The TTP sends this message to the broker, saves message;
3. The exchange receives message from the broker, saves it, determines its type;
4. For TTPBalanceIssue the exchange counts summary (of issued and redeemed tokens);
5. The exchange creates a copy of the request and changes some fields, where:

* messageId - new random identifier,
* previousMessageId - request message’s identifier,
* timestamp - current time,
* issued - quantity of issued tokens,
* redeemed - quantity of redeemed tokens;

1. The exchange creates new sender for queue with name “TTPBalanceIssue[messageId]”;
2. The exchange sends the message to the broker, saves it;
3. The TTP receives the message;

*(*

*In the current Web app which is emulating the TTP:*

* *The TTP saves received message and determines its type;*
* The TTP gets a summary (of issued and redeemed tokens).

*)*