Internet Financial EXchange (IFEX)

Searc

Our Proposals > IFEX Protocol >

2012-04-11 Partial Draft

Contents

Related Content

The Internet Financial Exchange (IFEX) Protocol

The Internet Financial Exchange (IFEX) protocol provides a mechanism for the identification, negotiation, description, execution and management of financial transactions over their lifetime. It can be used with arbitrary financial instruments or currencies, and arbitrary settlement systems.

Preliminaries

Scope

Due to the need to maintain applicability for low latency deployments (high frequency trading systems, etc.) per-message / per-transaction overheads should be minimized, however this does not justify burdening other user communities with exotic transport requirements.

IFEX therefore limits its scope to that of message content, remaining transport-neutral and allowing deployment over arbitrary transports to suit user requirements.

Specification Language

Message content is defined in the [JSON-SCHEMA] format.

The format was selected for its alignment to [JSON] and its relative structure, concision, extensibility and type affinity, despite relative novelty.

Mandated Fields

For reasons of extensibility and breadth of deployment, the definition of `required' fields is largely delegated to implementors.

Transport Neutrality

IFEX utilizes a request/response model over a point to point topology in in most cases, though the event notification (multicast or broadcast) model is also supported due to its importance to major parts of the

financial messaging community (see Edge Cases, above).

Subject to desired topology constraints, IFEX can be deployed via multiple transports, for example [TCP], [UDP], or [ZEROMQ].

Concepts

Nodes

IFEX Nodes are participants within an IFEX network.

Links

IFEX Links are logical communications channels (ie. transport layer sessions) opened between two or more IFEX Nodes in order to facilitate message passing. In the event that a transport layer does not inherently support sessions, then the IFEX Link is considered to be temporally bound between the first and last communications between the IFEX Nodes on that channel.

Financial Transactions

Financial transactions are the foundation of IFEX. They are identified with a Financial Transaction IDentifier (FTID), and may transition through a discrete series of states throughout their lifetime.

Financial Transactions

Financial Transaction Identifiers (FTID)

Financial transactions are identified with a financial transaction identifier (FTID; pronounced 'eff-tid').

The FTID for a transaction may be created by any party (though in practice this will usually be the originating party or "sender").

A FTID is 29 to 61 characters long and consists of three required elements and one optional elements, in the following order.

- **Account of Interest Identifier** (AOI; 8-34 characters)
  An arbitrary capitalized alphanumeric identifier that is used to identify the financial account of concern. Note that while the length of this field is adequate for an IIBAN or IIBAN to be used as the AOI, the use of an IIBAN or IBAN is NOT a requirement.

  Examples:

```
AA12011123Z56 (example IIBAN)
00000007 (example of a legacy, sequentially allocated AOI. Note that sequential
allocation is not recommended.)
MT84MALT011000012345MTLCAST001S (example of a Maltese IBAN)
K18DS4XP (example of a randomly allocated AOI)
```

- **Ledger of Interest Identifier** (LOI; 1-4 characters)
  An optional, one to four character alphanumeric identifier that is used to identify the financial ledger within the financial account of concern. This optional identifier is intended to ease the support of multi-currency accounts, ie. those with multiple ledgers. Note that while the length of this field is adequate for an ISO4217 currency code to be used as the LOI, the use of an ISO4217 currency code is NOT a requirement.

  Examples:

```
CNY (perhaps indicating Chinese Yuan Renminbi in a system only incorporating
currency-specific ledgers within the ISO4217 scheme)
XBTC (perhaps indicating Bitcoin in the X-ISO4217-A3 vocabulary)
ZUSD (perhaps indicating the United States Dollar code in the X-ISO4217-A3
```

```
vocabulary))
0 (numeric ledger identifier, understood by originating party)
FWIW (random but perfectly valid example)
```

- **UTC Second of Origination** (UTCSO; 14 characters)
  The UTC time of transaction origination, expressed in most to least significant order.

  Example:

  ```
  20120131032251 (ie: 2012-01-31 @ 03:22:51AM UTC)
  ```

- **Intrasecond Transaction ID** (ISTI; 5 characters)
  An arbitrary capitalized alphanumeric identifier to distinguish the transaction from others made within the same Account of Interest (AOI), optionally Ledger of Interest (LOI), and UTC Second of Origination (UTCSO).

  Example:

  ```
  Z4N6S
  ```

An example of a complete FTID, *excluding* the optional LOI component, is as follows:

```
AA12011123Z56-20120131032251-Z4N6S
```

Examples of complete FTIDs, *including* the optional LOI component, are as follows:

```
AA12011123Z56.XBTC-20120131032251-Z4N6S
00000007.ZUSD-20120131032200-7H3WS
MT84MALT011000012345MTLCAST001S.0MYR-20121110090807-N3YP9
```

In addition to a complete FTID, the final 20 character portion of the FTID that excludes the Area of Interest (AOI) specifier may be referred to as the **Ledger-Local Financial Transaction Identifier** (LL-FTID). Thus, corresponding examples of the LL-FTID are:

```
20120131032251-Z4N6S
20120131032200-7H3WS
20121110090807-N3YP9
```

The FTID and LL-FTID formats may be expressed in ABNF [RFC5234] as follows:

```
ftid = aoiportion dash utcso dash isti ; eg: 'AA12011123Z56-20120131032251-Z4N6S'
llftid = utcso dash isti ; eg: '20120131032251-Z4N6S'

aoiportion = aoi / aoi period 1*4char ; eg: '00000007.B', 'AA12011123Z56.XBTC',
'MT84MALT011000012345MTLCAST001S.0MYR'
aoi = 8*34char ; eg: '00000007', 'AA12011123Z56', 'MT84MALT011000012345MTLCAST001S',
'K18DS4XP'
utcso = 14digit ; eg: '20120131032251'
isti = 5char ; eg: 'Z4N6S'

char = digit / letter
digit = "0" / "1" / "2" / "3" / "4" / "5" / "6" / "7" / "8" / "9"
letter = "A" / "B" / "C" / "D" / "E" / "F" / "G" / "H" / "I" / "J" / "K" / "L" / "M" /
"N" / "O" / "P" / "Q" / "R" / "S" / "T" / "U" / "V" / "W" / "X" / "Y" / "Z"
period = "."
```

> *NOTE (2012-10-18): UTCSO specification will be revised for greater (but necessarily incomplete) validation of date/time values.*

Financial Transaction State

In order to unify disparate settlement systems and their particular processing requirements, IFEX mandates that all financial transactions MUST be considered to be in one of a small number of basic states at all times.

IFEX divides financial transaction states in to three core phases:

Pre Phase

All time prior to the establishment of a live transaction. This may include, for instance, the gathering of quotations for the execution of a transaction's settlement prior to its actual initiation, the end user selection of a settlement path based such quotations, or the period prior to receiving a positive response to the finally issued request to execute the transaction.

- **Initial State** (`INITIAL`)
  Covers all of the scenarios described above.

Live Phase

The period of time from when a transaction is entered in to the originating party's books ("fiscal ledger") and remains valid for any form of processing by that originating party, including as a result of external parties' activity (such as the initiation of cancellation or refund procedures).

- **Pending State** (`PENDING`)
  Subsequent to execution, but prior to the completion of settlement-related processes.
- **Settlement State** (`SETTLED`)
  After settlement, but prior to asbolute completion. For instance, the period for which a settled transaction remains exposed to third party cancellation or refund/reversal procedures.

Post Phase

After any possibility of additional transaction processing.

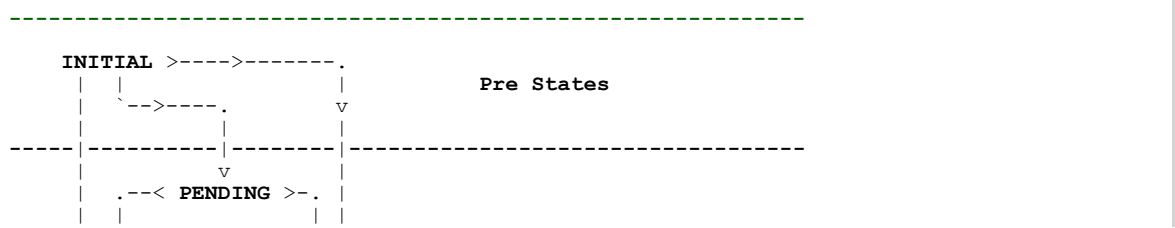
- **Success State** (`SUCCESS`)
  The transaction completed successfully.
- **Failure State** (`FAILURE`)
  The transaction did not complete successfully.
- **Partial Success State** (`PARTIAL`)
  The transaction was partially successful, for example in the case where only part of the expected settlement occurred. Implementation of this state should be avoided where possible.

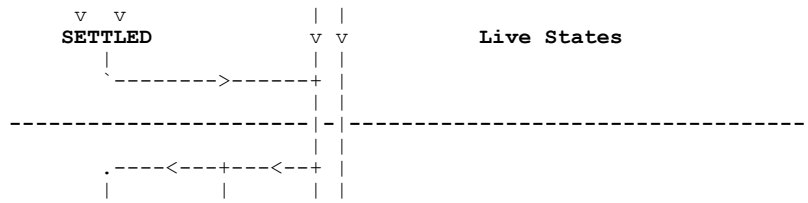
Transaction states may be expressed in ABNF [RFC5234] as follows:

```
txstate = prestate / livestate / donestate
prestate = 'INITIAL' ; negotiation, quotation, etc.
livestate = 'PENDING' / 'SETTLED' ; progress vs. 'still exposed'
donestate = 'SUCCESS' / 'FAILURE' / 'PARTIAL' ; result
```

Note that implementations MAY fail to make use of certain states, in particular `INITIAL`, `SETTLED` and `PARTIAL`. In addition, not all state modifications are valid; in the diagram below, state flows unidirectionally and downward at all times.

Transition Diagram





In summary, transactions MAY be created in any state. However, valid state transitions are as follows:

- **INITIAL** state may transition to PENDING, SETTLED or FAILURE.
- **PENDING** state may transition to SETTLED, SUCCESS, PARTIAL or FAILURE.
- **SETTLED** state may transition to SUCCESS, PARTIAL or FAILURE.
- **SUCCESS**, **PARTIAL** and **FAILURE** states are final and may not transition.

IFEX Messages

IFEX messages function to create, modify or describe financial transactions between two or more parties. All IFEX messages are equivalent in the sense that they may be viewed as attempts by the originating party to modify the perception (state) that one or more remote parties hold regarding the financial transaction in question.

Creation and modification of financial transactions may be made either `by decree' or `by negotiation'.

- Messages made **by decree** are those in which a message is sent with no attempt to negotiate acceptability to or input from the receiving party.
- Messages made **by negotiation** are those in which an attempt is made to negotiate the acceptance of, or some input from the receiving party. In this case, the desired state change may require remote confirmation and therefore may not occur immediately, or at all.

Message Identification

IFEX supports the following message identification strategies.

- **No Identification**
  In situations where message identification is not required (such as real time, 'execute or forget' transaction scenarios running over reliable transports), IFEX allows the exclusion of any form of IFEX message identifier.
- **Sequential Message Identifiers (SMI)**
  Use of sequential identifiers enables playback or network error recovery for IFEX feeds (high volume messages sent by decree), such as those deployed in broadcast or multicast (one-to-many) topologies for financial market data distribution.
- **Arbitrary Message Identifiers (AMI)**
  Use of arbitrary identifiers enables message identification and reference for scenarios where deployment of non-predictable (non-sequential) identifiers is preferred for security, legacy systems integration or some other reason.

Message Types

- **Request for Quotation** (RFQ)

Requests the receiving party or parties to provide a Quotation (`QUO`) regarding their capacity to execute AND/OR interest in executing the described transaction; as opposed to actually requesting its Execution (`EXE`).

- **Quotation** (`QUO`)
  Issued in response to a Request for Quotation (`RFQ`) message, this message describes the sending party's capacity to execute AND/OR interest in executing the transaction described in a prior Quotation (`QUO`).

- **Execution** (`EXE`)
  Transactions are exclusively requested to be performed (ie: moved from Pre State to Live State) via this message type. An EXR and/or RPT will follow from the recipient node.

- **Execution Response** (`EXR`)
  A response pertaining to an IFEX Message of the Execution (`EXE`) type, typically confirming execution or transitioning the transaction to `FAILED` state.

- **Request for Report** (`RFR`)
  Requests a report (`RPT`) regarding the status of a particular transaction from a remote party.

- **Report** (`RPT`)
  Except for the Execution response (`EXR`) message type in response to an Execution (`EXE`) message, any party wishing to inform another of transaction status changes MUST utilize this message type.

Party Identification

In recognition of the disparate nature of financial network identifiers and to avoid lock-in, IFEX does not mandate a particular party identification strategy. However, implementations SHOULD support the specification of financial endpoints via [IIBAN].

General Thoughts

```
party identification
 - signatures
    - gpg
    - other types (open support)
 - name
 - address
 - formal identifiers (references human and organizational entities)
    - tax id
    - corporate reg #
    - national id #
    - ... etc.
 - arbitrary properties
    - allow general classes only
       - allow definition of specific vocabularies outside
         of spec., but referenced.
```

Error Description

General Thoughts

```
Kinda like ICMP in IP... reviewed ICMP RFC and came up with the following notes.

    - distinguish between nodes and gateways as per ICMP???
      potentially useful for settlement path policy
      restrictions; see 11/0 below. no need to be explicit
      if this is the handling style.

    - option to define as a secondary protocol, in a similar
      fashion to ICMP being segregated from IFEX?
         3/0 = net unreachable;
         3/1 = host unreachable;
         3/2 = protocol unreachable;
```

```
      3/3 = port unreachable;
     ..... some of these could be related to specific
           errors within agreed/requested settlement
           path under IFEX, eg:
            - (intermediate?) asset unobtainable
            - destination [permanently?] unreachable
                . eg: IBAN doesn't exist.
                      Settlement nework no longer avaialble.
            - destination unknown

      3/4 = fragmentation needed and DF set;
     .... relates to alternate policy-based limitations,
           for instance retry and/or timeout policy.
      3/5 = source route failed.
     .... most obvious parallel is settlement path
           specification.

     11/0 = time to live exceeded in transit;
    ..... path timeout? allow specification of no other
          parties, asset types, or jurisdictions being
          involved and this type of failure message?
     11/1 = fragment reassembly time exceeded.
    ..... as per 3/4

     12/0 = pointer indicates the error
    ..... similar to notion of using field-based
          feedback for input errors.
      4/0 = source quench
     .... should be proactively handled with retry
          policies instead?  maybe still useful?

      5/* = redirect based on various things
     .... presumably not useful.

  <8|0>/0 = echo/reply <13|14>/0 = timestamp/reply <14|16>/0 = info
req/reply
     .... presumably not useful
Example errors:

 - MESSAGE NOT ACCEPTED
    - parse errors
       - message too big
       - structural parse error
    - field-specific input error
       - specific field
    - cryptographic error
       - signature invalid
    - temporary processing error (retry later)
       - optionally lengthen retry time?
    - business level error
       - not authorized
          - message type-specific?
          - message state modification?

 - MESSAGE ACCEPTED
    - TRANSACTION LEVEL ERROR
       - possible to have a recoverable/temp one?
          - include interaction with retry policy
          - estimated processing time
          - alternate settlement path report
          - optionally AUTOMATED transition to
            FAILURE

       - temp/perm unspecified processing error
          - retry flag? could just use tx state modification?
Possible delineations of error classes:

 - message level vs. transaction level
    - most critical distinction

 - message accepted vs. not accepted
    - seems more useful than some of the below
      distinctions from an implementer's perspective
       - complex/spurious in certain communications
         topologies, ie: broadcast/multicast

 - input vs. processing
    - useful?
       - kinda distinguishes between 'throw back to user
```

```
            with specific feedback' and 'transaction level'
            though the latter will likely be translated to
            an alteration of transaction state (ie. FAILED)
            so is perhaps of dubious use.

    - transaction level vs. message-level vs. field-level
        - 'transaction state reflected' vs. 'should be
          a 'temporary error' (for the latter two cases)...

    - temporary vs. permanent

        - the main thing is that if this distinction can be
          made by the remote node then it's only going to
          relate to the overall transaction state (ie.
          FAILED is permanent, otherwise it's not...
          except perhaps in that case that the message was
          a QUO response, in which case the receipient node
          might like to either leave open the potential for
          a revised and less-specific RFQ... timing out if
          not received?)
        - in other cases, the ability for a human or process
          to modify the request is going to be determined
          based upon the specific transaction stimulus and
          local system on the sending side ... so it's really
          not meaningful to talk about temp vs. permanent
          since it's implementation and/or situation specific
        - conclusion: not useful as a distinction.

    - technical vs. business
       - as per [EBICS] which has such error classes
         ('general technical error' & 'general business error')
       - difference is worth sharing?
          - if this occurs on a remote system then there
            is little purpose in being overly explicit
            since the only meaningful information that
            could contribute to a solution is whether it
            is temporary or not; and/or when to retry.

   --- rewrite when clear --- Rather than taking the approach of
   defining an exhaustive list of input related errors, IFEX messages
   MUST reference the field or fields in question using the source
   message schema (JSON Schema specifier?)
   ---------------------------
```

Temporary Errors

The remote system(s) MAY retry the transaction; if a Retry Policy has been negotiated then the remote system(s) MUST abide by it.

- **OVERSIZE**
  The message exceeded the maximum message size of the originating IFEX node.

- **UNKNOWN**
  An unknown temporary error state occurred.

Permanent Errors

```
error description
- link to input fields to avoid copious duplication
- general classes only
- allow definition of specific vocabularies outside of
spec., but referenced
```

Transaction Types

Certain situations MAY require financial transactions to be categorized in to one or more Transaction Types.

IFEX therefore allows financial transactions to be associated with one or more Transaction Type categories, either in a free-form manner or linked to a named and agreed Transaction Type vocabulary.

Outstanding:

```
    * When and how to negotiate transaction type affinities?

       * Use of JSON Schema to specify makes sense

       * Presumably normally in INITIAL state (RFQ, QUO)

          * Quotation selection should include the OPTION to
            accept/reject transactions based upon type within
            an agreed vocabulary; this is a requirement for
            regulators at the very least, and MAY also be of
            use for individual system implementers to provide
            Transaction Type based routing or processing
            policies.

          * Lock-down of type affinity required? One assumes
            this may be a de-facto requirement since a quote's
            transaction specification should probably be
            cryptographically signed (and thus unalterable.)
             * Such an approach makes logical sense re: any
               specific transaction specification fields,
               present or future, should be equally treated
               within a transaction.
                 * What about preferencing that isn't fixed
                   and is included in a quotation? Such things
                   might include what information?
                    * "Route around <x> jurisdiction"?
                       * Example of end-user empowerment, a
                         'good-thing'
```

Asset Specification

A financial transaction under IFEX may include an arbitrary number of assets.

```
     * Good idea? Makes things a lot more complex for
       implementations re: routing since many routes may
       be single route linked.

        * On the other hand, this forces implementers to
          implement PROPERLY (ie: flexibly), which might
          be a good thing.... really an important decision.

          Noting that transactions may include <x> asset
          for <y> asset, even in the digital currency for
          conventional currency buy/sell case, and this
          makes such handling unavoidable (at laest in a
          directional sense), this is possibly worth
          forcing?

       - arbitrary # of assets, each of which has:
          - asset class
          - asset type
          - quantity
          - denomination
          - reference per issue (serial num, ref/contract #, etc.)
```

Asset Class

Asset Class defines which Asset Type registry the Asset Type field references for asset identification; ie. the Asset Class and Asset Type fields are a hierarchical pair that together provide an extensible, categorized identification system for arbitrary financial assets.

Asset classes are defined in a new IANA-managed registry, the details of which are provided in the *IANA Considerations* section.

Two example Asset Classes follow.

- **ISO4217-A3**
  This Asset Class refers to ISO4217 Alpha-3, the standard registry for three letter currency codes used globally with conventional currencies. This Asset Class is managed by the International Standards Organization (ISO). Example types beneath this Asset Class are `NZD` (New Zealand Dollar) and `LAK` (Lao Kip).

- **ISO4217-A3-X**
  This Asset Class refers to a new, IANA-managed ISO4217 Alpha-3 extended code set, an extension registry used for three letter currency codes in the form ISO4217-A3 but not officially mandated through the ISO. As the purpose of this registry is to disambiguate use of such codes between arbitrary parties for the purposes of international settlement negotiation, for example via IFEX, the first documented use of a code is defacto claim to that entry and codepoints are never re-issued.
  - New digital currencies, eg. `BTC` (Bitcoin).
  - Historic ISO4217-A3 codes, eg. `ESP` (Spanish Peseta)
  - Unofficial denomination references, eg. `GBX` (Great British Penny Sterling)
  - Currency codes of currencies or states not recognized for technical reasons by the ISO but in commercial use, eg. `GGP` (Guernsey Pound), `JEP` (Jersey Pound), `IMP` (Isle of Man Pound or Manx Pound), `KRI` (Kiribati Dollar), `SLS` (Somaliland Shilling), `PRB` (Transnistrian Ruble), `TVD` (Tuvalu Dollar).
  - Historic market specific clearing-related derivative codes, eg. `USS` (United States Dollar; Same Day) and `USN` (United States Dollar; Next Day)

Asset Type

Asset Type defines which particular asset is being referenced within the Asset Class defined registry. Together, these fields provide extensible identification of arbitrary financial assets.

For example, in the Asset Class `ISO4217-A3`, the Asset Type `LYD` references the Libyan Dinar.

Asset Quantity

Assets may be specified using any of three methods.

- **Decimal quantity of a certain type of asset** in its default (or whole) denomination, as per conventional finance. For example `1.23` of the Asset Type `MYR` (Malaysian Ringgit) within the Asset Class `ISO4217-A3` (ISO4217 Alpha 3) would represent one 'ringgit' and twenty-three 'sen'. Similarly, `1.00000001` of the Asset Type `BTC` would represent one whole bitcoin plus one unit of the fraction colloquially known as a 'satoshi' after the pseudonymous Bitcoin author.

- **Market based assets** such as stocks of listed companies may be identified in one of two ways.

  - **The IMIC Asset Class**
    Within the IMIC Asset Class, an additional Asset Market field is used. An [IMIC] code is assigned to the Asset Market field to define an internet-based financial market in a manner that is superset-compatible with the ISO's existing Market Identification Code (MIC) standard [ISO10383]. Beneath the market, Asset Type is used to identify either the market-specific asset in either the market-native format or through an [ISIN] (in the case of multi-market assets traded on that market).

    For example, an asset specification using the `IMIC` Asset Class, the `XNAS` (or NASDAQ) market, and the Asset Type `PHII` would identify the stocks of 'PHI, Inc.', a helicopter company.

        - **The ISIN Asset Class**
          Under the 'ISIN' Asset Class the Asset Type specifies the [ISIN] of a financial asset. The important difference with ISIN-based specification is that ISIN specifies only an asset and does not provide a mechanism for identifying which market it is traded on; ISIN is often used with multi-market assets.

    - **Individual assets** identified by per-instance identifiers.

      This provision seeks to handle assets such as private bonds, contracts and warrants, transferrable options, complex financial instruments, serial-numbered notes, coins, or precious metal ingots with unique identifiers that, whilst commonly traded, MAY lack market affinity though be individually identifiable.

Transaction Terms

IFEX supports the negotiation of various important terms surrounding a transaction. IFEX's capacity to make such terms succint, explicit and machine parseable offers a significant benefit over many existing settlement systems that often define such terms in offline legalese.

Transaction Reversal or Cancellation Policy

Reversal or cancellation is a frequently encountered requirement in most mature financial systems; a fact for which the escrow facilitation industry bears adequate testament.

Timeout Policy

asoasoisa

Retry Policy

```
    - reversal/cancellation/timeout/retry policies
       - who pays fees
       - regulatory ingress handling
       - fraud liability shift
    - settlement preferences
       * worth making generic at all, or best leaving aside?
        - settlement system specific options
```

Temporal Specification

```
 - temporal specification
    - theoretical vs. practical (unforeseen delay, regulatory hold, etc.)
    - unknown support
    - min/max
    - range
    - average
    - guarantee
    - estimate?
    - timezone clarity
       - ... probably should use XML format.
    - various dates (at each hop of settlement path)
       - order date
       - dates of attempted action
```

Arbitrary Properties

- arbitrary properties
  - dividend preferencing for some assets
  - information resulting from localized legislative requirements

Fee/Tax/Discount Support

Fees, taxes and discounts are often a part of a mature financial network. Previously, poor handling of these parameters at the protocol level has resulted in inelegant, ad-hoc, system-specific solutions (a prime example is the complex realm of United States
state tax calculation).

IFEX recognizes the requirement for various types of fees, taxes and discounts and provides broad support using a two-class approach, respectively termed 'calculable' and 'incalculable'.

Calculable fees are preferable in that they enable financial transactions to be executed between parties that do not require addition temporal overhead due to redundant communcations and processing at the processor (recipient) node(s).

Calculable

A calculable fee, tax or discount is one in which an algorithm executing on a node considering the transaction (`QUO` received) CAN be supplied with adequate information to calculate the appropriate fee, tax or discount's effect upon a financial transaction's overall execution and settlement.

For example, a financial transaction in a certain type of good or service MAY be subject to a fixed, 3% tax at the state or province level of the local jurisdiction in which the financial transaction occurs (such local taxes are frequently found in the United States). In such cases, the node may provide this information for calculation to the considering party via the Quotation (`QUO`) message type.

Incalculable

An incalculable fee, tax or discount is one in which an algorithm executing on any node considering or privy to the transaction CANNOT be supplied with adequate information to calculate the appropriate fee, tax or discount's effect upon a financial transaction.

For example, a transaction with a particular merchant may be subject to a 3.99 `NZD` (New Zealand Dollar) discount when a certain voucher code is supplied. In this case, it would be beneficial for both the system representing the end user (potential purchaser of some good or service) had this information available. However, because for reasons of fraud prevention third party systems cannot be privy to the description of which voucher codes are valid or have been previously used, it is necessary to pass the incalculable transaction to the processing party in order to determine the validity of voucher-based discounts. In such a case, the merchant-side processing node MUST provide a revised Quote (`QUO`) to the originator detailing the proposed resulting impact on the overall transaction. In extreme cases, it MAY be possible (though undesirable) for the nodes to negotiate the immediate settlement of the transaction, wherein the merchant-side processing node MUST provide a Report (`RPT`) to the originator detailing the final impact on the overall transaction.

```
fee/tax/discount support
 - type
    - "node-local", or "transaction-linked" (affects all settlement paths)
    - settlement path specific
       ... providing for both means shorter messages where 1+ all-path
           taxes/fees/discounts exist.
       ... not providing for both is simpler. seems better.
           premature optimization... and all that! ;)
 - locally levied/withheld, or remotely (3rd party)
 - identifier
    - authority / specific fee/tax/discount type
    - calculable cases
       - rationale (%-based, tiered, flat, etc.)
         - option to sidestep variants based upon peer-estimate vs.
exhaustive specification?
    - incalculable cases
       - quotation based, *possibly* typical % to aid in est?
          - feeds back to need for reputation system
```

Settlement Paths

Settlement paths are a core feature of IFEX: by allowing IFEX Nodes to reason meaningfully and in real time about available settlement paths, IFEX creates a free market for financial settlements and enables redundant financial routing.

To acquire potential settlement paths, IFEX Nodes issue a Quotation Request (`RFQ`) message to peers. Some of these peers MAY respond with Quotation Response (`QUO`) messages, each of which contains information regarding one or more potential settlement paths.

The originating node considers the resulting quotation(s) and MAY select one as appropriate for execution based upon local policy. The originating node then sends an Execution Request (`EXE`).

In order to prevent undue latency and network overhead caused by sending one Quotation Request (`RFQ`) message to all peer IFEX Nodes, per transaction, Quotation Response (`QUO`) messages may provide arbitrary periods of validity for each of the settlement paths offered in Quotation Response messages. In the case that Quotation Responses have previously been received by the originating node, these MAY be considered on a "newest quote from identical node with identical requirements takes priority" basis with the newly received quotations, or may be considered locally without creating an `RFQ` / `QUO` process for the transaction at all.

When an Execution Request (`EXE`) is sent to effect a given transaction, it references the message identity of the Quotation Response (`QUO`) upon which it is based.

When multi-hop paths are used in settlement, individual IFEX transactions may be created between IFEX Nodes participating in a given hop. For example, given the multi-hop path:

```
A -TX1-> B -TX2-> C
```

1. [TX1] IFEX Node 'A' may issue a Quotation Request (`RFQ`) for delivering funds to an IIBAN associated with IFEX Node C.
2. [TX1] IFEX Node 'B', thinks that it can deliver to C (for a fee). Because their IFEX Link is not permanent, Node 'B' seeks to verify this immediately as follows.

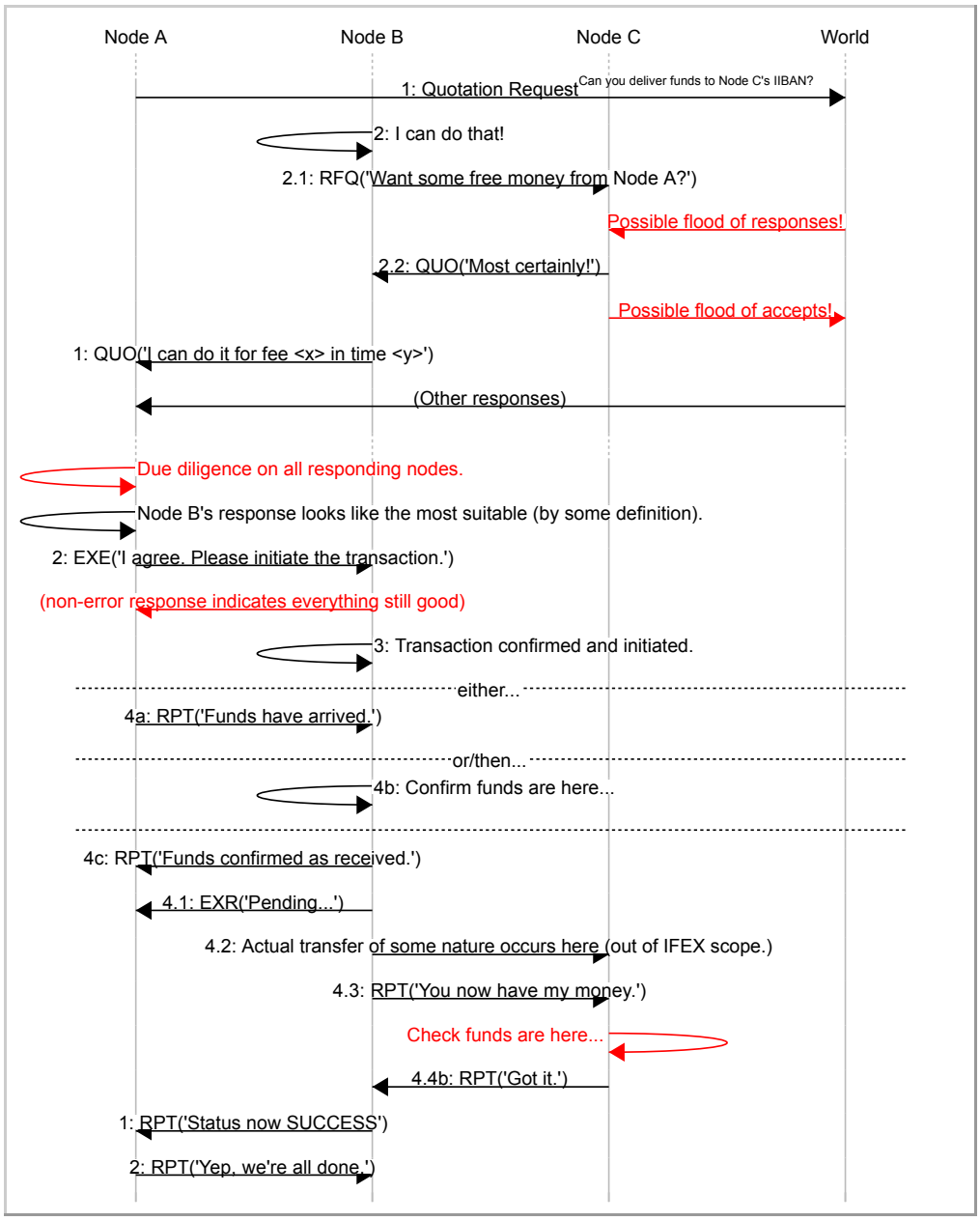
    1. [TX2] IFEX Node 'B' opens an IFEX Link to Node C, creates a new IFEX transaction, and asks IFEX Node C in a Quotation Request (`RFQ`) message if it wouldn't mind receiving some free money from IFEX Node 'A'.
    2. [TX2] IFEX Node 'C' graciously indicates its willingness to receive free money at the IIBAN in question (simultaneously indicating its validity) by replying with a `QUO` message to this effect.

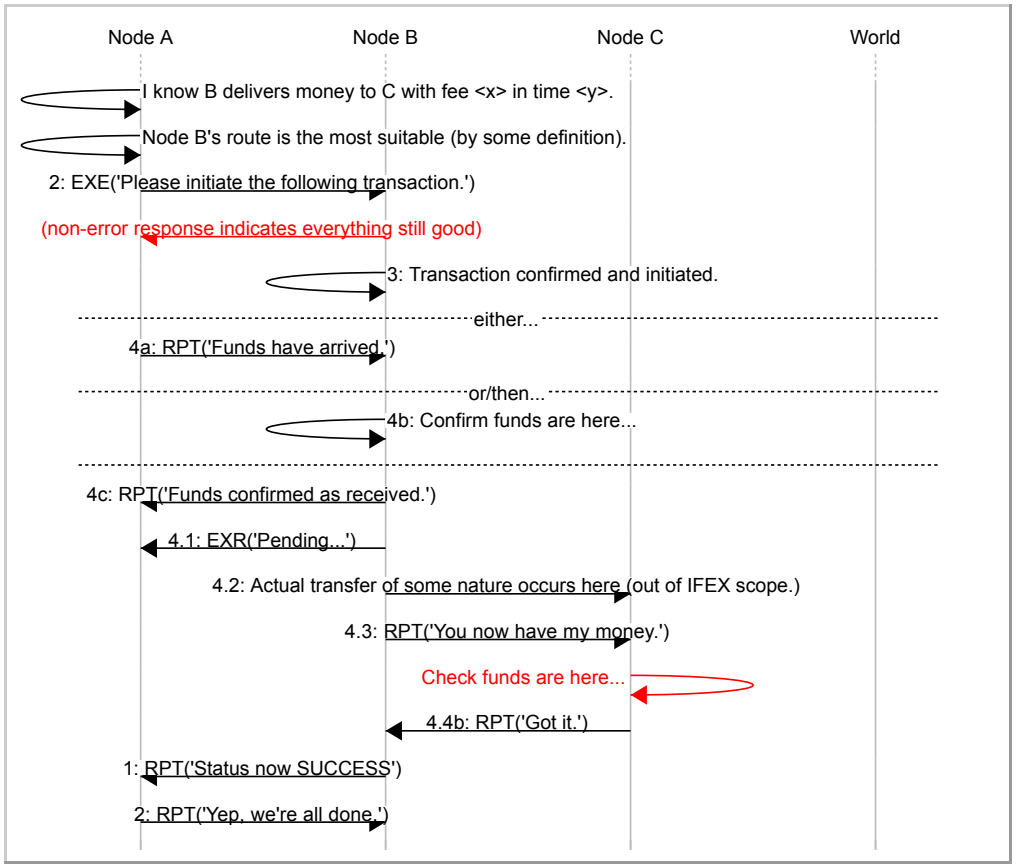
1. [TX1] IFEX Node 'B' is now safe in the knowledge that the transaction can be effected with IFEX Node 'C'. It responds to the original IFEX Node 'A' Quotation Request (`RFQ`) with a Quotation Response (`QUO`) including a small fee.
2. [TX1] IFEX Node 'A' selects IFEX Node B's quotation response from the set it received due to having lower temporal and/or fee requirements, issuing an Execution Request (`EXE`) to Node 'B'.
3. [TX1] IFEX Node 'B' receives the Execution Request (`EXE`) from IFEX Node 'A'.
4. [TX1] IFEX Node 'B' receives a Report (`RPT`) From IFEX Node 'A' indicating that funds have settled (transaction state: `SUCCESS`) and acknowledges this after verification with a Report (`RPT`) message.

    1. [TX2] IFEX Node 'B' issues an Execution Response (`EXR`) to Node 'C', indicating that TX2 has entered `PENDING` state.
    2. [TX2] IFEX Node 'B' forwards funds to Node 'C'.
    3. [TX2] IFEX Node 'B' issues a Report (`RPT`) message to Node 'C', indicating that TX2 has entered `SUCCESS` state.
    4. [TX2] IFEX Node 'C' acknowledges this with a Report (`RPT`) message.

1. [TX1] IFEX Node 'B' issues a Report (RPT) message to Node 'A', indicating that TX1 has entered `SUCCESS` state. (Possibly including cryptographic proof from IFEX Node 'A')
2. [TX1] IFEX Node 'A' acknowledges this with a Report (`RPT`) message.

NOTE: Please compare the mscgen diagram of said process - with the alternative lowest-latency process where B assumes C is reachable from a previous quote, and A already has B's quote to deliver.






```
settlement path
 - multi-hop settlements paths vs. quotation mechanism
    - active path validation after RFQ and before 'QUO' could result in DDoS!
    - need for a lighter-touch RFQ mechanism that doesn't necessarily have network results at
'C'
    - issuing of a signed, previous QUO from C regarding its willingness to receive funds to A
from B could serve as an adequate replacement for real-time validation
    - sometimes realtime validation is nice, look at this as a later extension or non-standard
mode? (disable by default)
 - network identification
       - risk exposure
          - temporal specification (as above)
          - reliability
             - cryptographic trust metrics
          - legal jursdictions traversed
          - transparency (status tracking availability...)
       - conversion regimes (FX...)
    - sum total of above
 - multi-path
    - split transfer for security/reliability
```

Transaction References

The Financial Transaction IDentifier (FTID) of a transaction may be used within a message for following purposes:

- **Assertion**
  The creation of a Financial Transaction IDentifier for a transaction that has not yet had such an identifier specified. The association process MUST occur in a Request for Quotation (`RFQ`) message, a Quotation Response (`QUO`) message, an Execution Request (`EXE`) message, or an Execution Response (`EXR`) message. Initial assertion MUST NOT occur in a Report (`RPT`) message.

- **Association**

Messages prior to EXR (`RFQ`, `QUO`, `EXE`) message MAY reference the FTID supplied in a Request for Quotation (`RFQ`) message. It can be created if none is yet known.
All Execution Response (`EXR`) messages MUST reference an FTID. The FITD can be created if none is yet associated.
Report (`RPT`) messages MUST reference the existing FTID.

- **Error Recovery**
  Any error message should reference the FTID and of the message whose transaction it affects.

- **Reversal/Cancellation**
  Subject to the transaction-specific Reversal / Cancellation Policy negotiated between nodes, a node may wish to reference a transaction in a subsequent message in order to reverse or cancel it.

Subpages (10): IANA Considerations Implementation Considerations Introduction Lock-in Considerations Protocol Design Considerations References Requirements Robustness Considerations Security Considerations Terminology

Comments

You do not have permission to add comments.