**POC Name**

**Transactions & Holdings Registry – General case**. DLT Technology: Hyperledger fabric

**POC objectives**

To demonstrate a viable model to handle investors orders on an ordinary European-style mutual fund (not necessarily a UCITS fund). We strive to demonstrate the following “concepts” with the chosen technology:

* **Order processing:** to process an investor (*i*) subscription or redemption order (*o*) on a fund product (investment vehicle) (v)
  + A subscription order paid in cash results in the issuance of new shares of the investment vehicle credited to the investor (or his intermediary and ultimately credited his to securities account). New shares are uniquely identified.
  + A redemption order results in a payment in cash made to the investor’s cash account. Redeemed units must have been previously subscribed.
* **Holdings:** to maintain a registry of investors’ holdings
* **Cash settlement:** cash transactions are assumed to be externally processed using an existing cash transfer infrastructure. The status of a cash transfer is provided by a trusted party. The Transfer Agent is trusted to provide this oracle. This POC demonstrate this interaction by processing messages to and from this infrastructure. The actual message format is not relevant: the POC merely outline a basic protocol.
* **Unified model:** the demonstration works for funds registered under a CSD model as well as funds under a TA/Registrar model.
* **Delayed settlement:** transactions on funds are not settled instantly, but on T+1 (simplifying assumption, in order to avoid to manage a fund valuation calendar at this POC stage).
* **Carried on checks:** we demonstrate that a number of checks may be carried on a diverse entry points. In this context, checks remain simplified, and we focus more on the steps at which checks may be performed than on the completeness of the checks themselves.
* **Consensus protocol:** the validation of a transaction results from a consensus between relevant parties only (for more details on the different phases to reach a consensus, see Appendix C document) .

**Assumptions on actors**

**Direct investment**: in this POC, investors are assumed to be directly connected to the platform and may initiate orders by sending messages to peers. Investors are not themselves “peers” of a fund they invest in: they do not get a copy of the fund’s data. In other words, investors are registered parties known to the platform, but trust other parties (peers) to process their orders. Other POCs shall detail the chain of distribution with various intermediaries. Our POC thus exhibits a platform on which institutional investors such as large corporations could invest directly.

**Settlement actors**: we analyze several settlement scenarios, depending on whether the investor’s bank (split in a cash account holder role and an investor’s securities custodian) directly participates the settlement consensus or not (e.g. nominee model). The POC is going to figure out the easiest scenario to implement. Other POCs will explore and demonstrate the validity of our model for all settlement scenarios.

**Not in the scope of this POC**

We do not demonstrate the following in this POC, and shall address such required items in other dedicated POC’s:

* Data privacy: although we limit data to be shared by a limited number of trusted actors (peers), we do not demonstrate here how a finely grained data access policy may be established. In this POC, a peer has access to all fund’s data. However, non-peer participants access to fund’s data through chaincode operations which limit their visibility on a fund’s data
* Settlement interfaces details: securities custodians’ interfaces details (on behalf of the investor, on behalf of the fund). This POC merely exposes an abstract messaging model for settlement operations.
* NAV construction and validation details: NAVs are exogeneous and published by a trusted peer
* Investor onboarding & suitability checks details (i.e. addressing the retail market segment)
* Indirect distribution through a chain of Intermediaries & order earmarking
* Fund master data management and fund distribution agreement workflow are not detailed in this POC: we assume everything has been initialized proper.

**POC simplifying assumptions**

* As stated above, the demonstration is limited to investments vehicles (“funds” or “shares”) with a daily valuation on business days (to simplify, we omit holydays), with orders expressed in units and processed at an unknown price with settlement a T+1 (this avoids the subtleties of orders expressed in cash or funds without units as well as fund valuation calendar management)
* An order is expressed in fund’s units. For orders posted before cut-off (12:00 AM), the price is known at the end of day, based on the daily published NAV. Orders become irrevocable after cut-off: they are deemed “executed”.
* This transaction is in effect settled on the morrow. After 12:00, the order is likewise processed the day after. Later, one may set a configurable cut-off, and manage orders settled at a known-price.
* For the POC, we consider only one investment vehicle (e.g. ISIN code, or share class) for a fund. Funds with multiple shares are not demonstrated here.
* Orders are allowed every day, but processed only on business days. Cut-off for orders is 12:00 AM
* The are no subscription or redemption fees paid upfront. Orders are not expressed in currency amounts, but in number of units only
* There no other limitations, such as a cap on subscription or a lock-up period for subscribed shares
* Any investor may post orders. Compliance and suitability checks are assumed to have been passed beforehand.
* We assume that units are decimalized up to 3 digits. Similarly, NAVs are decimalized up to 3 digits (obviously, in a general setup, such settings are product dependent)
* NAVs are an exogenous information provided by an oracle: the fund accountant.
* The settlements procedure is simplified, assuming all trust is put in the TA (e.g. following a “direct order” model). Testing hybrid models supporting more complex setups (e.g. a CSD model with non-domestic investors custodians) would require a dedicated POC.
* Cash transfer status is decided instantly by the oracle. No delay is introduced. A fail during cash transfer simply cancels the associated order: there is no postponing or additional attempt.
* Order can’t be cancelled and may be settled only in the NAV’s currency. There are not cross-currency orders in this POC.
* Time based events and conditions are derived from a single time and time zone, provided by an external oracle. For the sake of the simulation, time shall be accelerated on the platform, say 10s to simulate one day
* Orders are only subscriptions or redemptions initiated by an investor. We do not cover in this POC other kind of transfers or corporate actions on the fund.
* We assume for simplicity that the fund has already been initiated and we do not have to process specifically orders with an “initial NAV”
* We assume that fund shares are not directly transferred from an investor to another: shares are always issued and cancelled for every investor

Although simple, this set of assumptions captures many existing issues. The objective is not to demonstrate how the technology may cover tricky cases, but rather, what is the value created by the technology on such a simple common case.

Distributors do not appear in this POC: investors are assumed in our simplified context to directly operate the platform. The accurate representation of the distribution chain requires a specific POC (“Intermediated distribution” & “Investors Onboarding”)

**Object model**

This POC animates one chaincode object: the Fund. To get an overview of our chaincode model for the whole TFDC platform, see Appendix B document.

We assume that all parties (including investors) are already properly registered and identified on the platform, by some previous trusted process (other POCs shall detail this part). We also assume that all required fund data have already been initialized, by some previous trusted process (another POC shall also detail this part). In other words, we assume that all is ready to process investor orders.

We assume further that all participants know the chaincode ID of the funds they are interested in. Therefore, we shall not deploy FundDirectories in this POC.

The fund maintains a state which describes:

* Its main characteristics (name, ISIN, legal form) [Fund Master model cartridge]
* For each involved party (not investors), the vector of supported roles [Parties section of Master model cartridge]
  + Each pair (party, role) is enriched with a description:
    - Private fund identifier for this party (e.g. link to legacy systems)
    - Private account number for this party
* Liabilities Balance Sheet model cartridge
  + The total number of outstanding shares
  + A registry of the shares detained by each investor, with a distinction between the order status (executed, settled) and the expected next event dates (valuation, expected settlement)
* Asset Balance sheet model cartridge:
  + the latest validated NAV {NAV date, NAV amount, NAV currency}
* Pending orders with their status, from “executed” to “pending settlement”. Settled orders are not kept (left in the ledger only).

To maintain this state, the chaincode relies on a ledger of transitions which are of two kinds:

* Order transitions (o)
* NAV publication transitions (n)

An order (o), is described as follows:

* An anonymous order identifier
* Order posting Timestamp
* The investor identifier (INV) and signature
* The investment vehicle identifier (FUND)
* An anonymous identifier representing the cash account of (INV)
* An anonymous identifier representing the securities account of (INV)
* Order type: {‘SUBSCRIPTION’, ‘REDEMPTION’}
* Order status: {‘EXECUTED’,’PENDING’,’SETTLED’, ‘REJECTED’, ‘FAILED’}
* The signed number of units (+ for ordinary subscriptions, - for ordinary redemptions)[[1]](#footnote-1)
* The settlement date
* The NAV date and amount used for order valuation
* Order execution Timestamp
* Order settlement Timestamp
* The settled amount and currency

A NAV transition is described as follows:

* {NAV date, NAV amount, NAV currency, Fund’s Net Assets}

**Supported fund administration models**

RTA model (e.g. Luxembourg-style funds, French-style FCPE): a single participant has both registrar and transfer agent roles.

CSD-model (e.g. French-style funds): the CSD has a registrar role (jointly with a registrar which only maintains the total number of issued shares), the TA the transfer agent role.

Registered form model: the fund’s registry is directly administered by the issuer (i.e. shares are issued under a registered form): the fund manager acts as registrar and issuer, and possibly transfer agent as well.

**Identified straightforward generalizations**

The following generalizations toward a more realistic platform could be easily developed from this POC:

* Cut-off time may depend on the product.
* Same for decimalization of units and NAVs
* NAV follows a more complex calendar: weekly, monthly for instance, holydays are also to be considered
* Orders may be cancelled before cut-off (rescinds)
* Executed orders may fail to be properly settled (e.g. cash is not available). This may be true for subscriptions and for redemptions as well (“fund run”)
* Orders passed at the “initial NAV” are valued differently (i.e. shares are purchased at par price)
* Orders may be specified in amount rather than units (the registry is still maintained in units)
* Orders may be passed at a known price rather than unknown (related with generalized cut-off)
* Orders may be constrained within a given period (e.g. limited subscription period)
* Orders or holdings may be capped by a constraint (e.g. investment ratio, absolute cap)
* Orders may be subject to upfront fees paid to some other party than the fund itself (such fees structure constitutes an enrichment of the role description)
* There may be several TAs, for instance for different countries (e.g. Prime TA and local agents)
* There may be several fund managers (for instance, each managing a given carve-out). The fund manager role may be detailed into a “principal” fund manager (which participates the NAV consensus), a “receiver” fund manager (which participates the order consensus, e.g. the fund manager of a treasuries carve-out)
* The investor’s custodian may be a different party than its cash account holder (e.g. FCPE funds, registered form vehicles…) (this is in the base model, but a specific scenario has to be set up to demonstrate this)
* The registry may be enriched to follow inventory accounting rules
* Funds may manage several investment vehicles (e.g. share classes)

**More complex generalizations**

Examples of more complex generalizations, that would require an extended rework of data model and workflows:

* Extending order types, not necessarily at the initiative of the investor (e.g. administrative transfers & adjustments, corporate actions, …)
* Introducing delay in cash payment, the ability to manage settlement fails
* Cross-currency orders; cross-jurisdiction orders, which must abide by multiple sets of constraints
* Hedge-funds, European-style: managing vehicles in “series”, with GAV and NAV handled separately
* Hedge-funds, US-style: orders in amount, no NAV, but “equalization factors” to convert amounts into units
* Lock-up periods
* Funds with no units
* Orders representing “commitment”, with a settlement triggered when funds are called upon by the fund manager (e.g. infrastructure funds, real-estate funds, private equity…)
* Rescind NAVs (most complex…)
* …

**Business processes covered by this POC**

We cover 2 interoperating processes:

* Orders, which is divided in two phases:
  + Order validation
  + Order settlement
* NAV publication

Note: in this POC, we do not allow for investors to cancel their orders. In a more general setup, orders may be rescinded before cut-off. Formally, this will add a new state in our workflow: Executed orders, i.e. that may not be rescinded.

The figure below exhibits these business processes expressed as state-transition diagrams.



**Platform participants and roles in this POC**

For the 3 workflows supported, the table below shows the roles of each participant (see Appendix A document for a description of the various roles).

In Hyperledger, the consensus to validate a transition in a workflow (i.e. the state of the chaincode changes) is established by a two-fold process: endorsement, then ordering, then validation. A single party may play several roles.

Roles in the consensus essentially depends on the settlement scenario retained. In the table below, we do not detail the roles played by the investor’s bank does, since it heavily depends on the settlement scenario retained (see below).

|  |  |  |  |
| --- | --- | --- | --- |
| Participant | Order validation | NAV publication | Order settlement |
| Investor (INV) | Non-peer initiator | - | - |
| Issuer (ISR) | Endorsing peer | Endorsing peer | Endorsing peer |
| Fund Accountant (ACC) | Endorsing peer | Initiator (peer) | Endorsing peer |
| Fund Custodian (CUS) | Endorsing peer | Endorsing peer | Endorsing peer |
| Transfer Agent (TAG) | Endorsing peer | Endorsing peer | Initiator (peer) |
| Fund Registrar (RGR) | Endorsing peer | Endorsing peer | Endorsing peer |
| Investor Cash Account Holder / Nominee (CSH) | (see variants) | - | (see variants) |
| Investor Sec. Custodian (SEC) | (see variants) | - | (see variants) |
| Fund Manager (FMG) | Non-endorsing peer | Endorsing peer (optional) | Non-endorsing peer |

**Chaincode transitions**

Important: a “transaction” on a distributed ledger may not correspond to a “financial transaction”. Formally, a database transaction instructs the database to evolve from a validated state at t0 to another validated state at t1.

The table below shows how each state transition (transaction) is managed: a transition is a message proposed by an initiator to a fund’s endorsing peers defined for this transition. Technical checks (signature, well-formed messages) ae implemented as part of the endorsement policy (ESCC). Business checks are implemented as part of the validation rules (VSCC). Other automated business rules are part of the chaincode state maintenance operations, which are applied once the transition is validated.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Chaincode Operation | State transition | Initiator  (transition proposal) | Endorsing Peers | [business rules] |
| PostOrder | Order: (nil) -> INITIAL | Investor (INV) | Issuer (ISR)  Fund Accountant (ACC)  Fund Custodian (CUS)  Transfer Agent (TAG)  Fund Registrar (RGR) | (1) |
| ValidateNAV | Nav: INITIAL -> VALIDATED | Fund Accountant (ACC) | Issuer (ISR)  Fund Custodian (CUS)  Fund Manager (FMG) | (2) |
| ValueOrder | Order: VALIDATED->VALUED | Fund Accountant (ACC) | Issuer (ISR)  Fund Custodian (CUS)  Transfer Agent (TAG)  Fund Registrar (RGR) | (3) |
| InstructOrder | Order: VALUED -> PENDING | Transfer Agent (TAG) | Issuer (ISR)  Fund Accountant (ACC)  Fund Custodian (CUS)  Fund Registrar (RGR) | (4) |
| SettleOrder | Order: PENDING -> SETTLED  Order: PENDING -> FAILED | Transfer Agent (TAG) | Issuer (ISR)  Fund Accountant (ACC)  Fund Custodian (CUS)  Fund Registrar (RGR) | (5) |

Please mark that proposed transitions which are not endorsed are not tracked on the ledger. Endorsed transitions that eventually fail to be committed by the consensus service are kept on the ledger in a “failed” state. Such failed transitions do not participate the simulation of the next state of the chaincode.

Note that the presence of optional endorsers, or different endorsers depending on the operation, is not directly supported by HL: a single endorsement policy has to be defined once and for all when deploying the chaincode.

The way to work around this is to declare all endorsers then, in the detail of the endorsement code, to check for roles and force unconcerned peers to systematically endorse the proposal[[2]](#footnote-2). At the moment, checks performed in the ESCC method may be performed by any peer: endorsing policy would require at least one endorsement signature only.

For the moment, we leave open the problem of such a strict endorsement policy, which does not tolerate the failure of an unimportant node. We expect that post-V1 HL will bring some improvement.

Matrix of checks and state maintenance actions detailed for each transition. Well-formed message checks leverage JSON schema 4 capabilities.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Endorsement policy | | State Maintenance performed upon transition validation | |
| (1) | Valid signature of investor  Check order is well formed  Check mandatory fields  Check decimalization constraint (a quantity expressed with 4 decimals is rejected for instance) | An investor may not redeem more than its current stock (unitary check).  Current stock includes stock in “executed” status. | Orders before cut-off  Update registry with “executed status” positions  Orders registered in state with “executed status” and today’s NAV | Orders after cut-off  Orders registered in state with “executed status” and next NAV date |
| (2) | Valid signature of fund accountant (checked against roles for fund)  Check that NAV x outstanding shares = net asset + Epsilon, with Epsilon < Net asset X 10E-(NAV digits)[[3]](#footnote-3) | NAV is > 0 | Update latest NAV  Search state for orders in executed status.  Post ValueOrder proposal to peers for the NAV published date  One single transition affects all orders. | Optional: materialize in registry the “residue” of non-attributed units due to NAV rounding. In other words, ensure that assets = liabilities on the fund’s balance sheet |
| (3) | Check that the proposed NAV date is published in the chaincode state | NAV > 0  An investor may not redeem more than its current stock (net quantity check) | Orders in “validated status” with the published NAV move to “valued” status. |  |
| (4) | Settlement instructions amounts match the valued order |  | Orders in “valued” status are netted and move to “pending settlement” status | For each validated transition, the TAG posts settlement instructions to the external cash/securities system |
| (5) | Settlement receipts (ACK messages) match the pending settlement order |  | Matches: orders in “pending settlement” status move to “settled” status.  The registry is updated  The number of outstanding shares is updates. | Mismatches: orders in “pending settlement” status move to “failed” status. |

Pending design decision: granularity and contents of the ValueOrder message: technically it is possible that a single “NAV

Pending design decision: contents of settlement instructions

**Notes on the ordering of transitions**

Transitions are not in general commutative: the order matters.

Example: starting with a stock of 5 units, it is not the same to subscribe 10 units then redeem 9 (allowed) than doing it the opposite way (forbidden).

We may tolerate that in the first stage of the process, all orders are checked unitary:

* If we start with a validated stock of 5, no redemption of more than 5 is allowed
* However, we have no way before the validation is reached to figure out if many smaller redemption orders where passed. The check performed at the order level takes into account the “transitory” stock, explained by other executed transactions.
* Therefore, an additional check is performed at valuation time on the net quantity, taking into account several orders. The behavior adopted in this POC is to reject the full set of orders from this investor in “EXECUTED” status.

Hyperledger enforces the serializability of transactions by ordering the final consensus (PBFT consensus).

Endorsed orders refer to a version of some keys in the chaincode (“anchor”), so as to avoid fund modifications while validating transactions. The keys chose for anchor correspond to the “fund master cartridge” (note that this suppose that each of our JSON cartridge is stored under a different HL key).

The “orderers” (participants which vote for endorsed transactions and finalize the validation with a PBFT consensus) are all the listed participants. We assume that this service may remain private (as documented by HL) to the community of stakeholders for a given fund (thus potentially larger than the population of peers)

In other words, we shall open an “ordering service channel” for each fund. For the sake of simplicity, however, in the POC we just publish one single ordering service for everyone.

**Managing external events**

**The platform supports the following external events, which are managed by a peer which is trusted to act accordingly.**

* Time-event: is actually not managed as an event, but as a service (caller asks for trusted universal time whenever needed). Time is used to manage cut-off and end-of-day procedures.
* NAV events: the fund accountant is trusted to receive NAV calculations from its own accounting system (i.e. managing the assets side of the fund balance sheet) and publish them accordingly by initiating a transition on the chaincode. The end-of-day event is local to the fund accountant peer to trigger a NAV publication: it is not managed within HL. ACC thus triggers a transition by publishing NAV.
* Settlement instructions: outgoing event. The TAG is trusted to process every order which has been successfully moved to “InstructOrder” and generate the appropriate cash / security set of instruction (to be detailed)
* Settlement instructions: incoming event. Similarly, the TAG is trusted to process every incoming payment/delivery acknowledgment message by initiating a “SettleOrder” transition.

In the POC, settlement instructions and receipts are stylized versions of the real SWIFT messages, containing most important information such as amount, currency, quantity and security identifier if relevant. Instructions shall cover the transfer of both cash and securities. In this POC, we assume no netting procedure is being used.

An important feature in this message may be to ensure an “end-to-end” authentication by conveying in the instruction the reference to the net valued order and (depending on the settlement scenario) the cryptographic seals of the investor and the fund.

In our POC implementation, listening to incoming events and producing external events is carried on by the SDK associated to the relevant peer. Therefore, we have:

* A SDK for the investor (posting orders, querying the outcome of its orders)
* A SDK for the fund accountant (posting NAVs)
* A SDK for the transfer agent (receiving and posting settlement instructions)

**Internal events**

Internal events are designed to be notifications sent by the chaincode instance to some SDK client to “wake up” this actor and perform an action.

Another use is for the chain code to ask for some external information before taking a decision. For instance, a fund custodian would possibly verify that the cash has been received on a fund’s cash account before validating a “SettleOrder” transition.

Such events are typically managed within the chaincode validation logics. In our implementation, there may be a different set of logics for each role.

Unfortunately, the HL SDK API does not support such signaling mechanism from the chaincode down to the client.

Left as an open – irritating - question for now, although this feature is much structuring to solve settlement checks and eventually settle an order.

**Querying a Fund chaincode**

Peers (endorsing or non-endorsing peers) maintain a full copy of the chaincode (state and ledger). Technically, they may access any key/value pair in this chaincode. In our setup, peers must not be constrained by privacy issues (if this is the case, consider how to split a part of the payload in another chaincode…)

For non-peer participants, we deploy a query API to retrieve a view on the state of the chaincode.

Such queries check beforehand that the querying party has an authorized role to see the data (this is implemented by our specific Query() function in the chaincode). The sub-model (possibly transformed before returning data) is specified in the query argument.

Query matrix vs roles.

|  |  |  |  |
| --- | --- | --- | --- |
| Participant | Orders | Holdings | Fund Master Data |
| Investor (INV) | Query state its own orders  Query ledger for past, settled orders | Query its own holdings  Total amount of outstanding shares  NAV  Ledger of past published NAV | Query master, including roles but not roles enrichments |
| Issuer (ISR) | Every state variable | | |
| Fund Accountant (ACC) |  |  |  |
| Fund Custodian (CUS) |  |  |  |
| Transfer Agent (TAG) |  |  |  |
| Fund Registrar (RGR) |  |  |  |
| Investor Cash Account Holder / Nominee (CSH) |  |  |  |
| Investor Sec. Custodian (SEC) |  |  |  |
| Fund Manager (FMG) | Query state net total orders  Query ledger net past settled orders | Total amount  NAV & ledger of past published NAV |  |
| Other participants |  | Total amount  NAV & ledger of past published NAV | Query master, including roles but not roles enrichments |

With forthcoming POCs demonstrations, we shall extend this matrix to support queries by distributors (e.g. aggregating holdings & orders at their level).

**Network topology**

Each participant runs a platform node and is securely connected to the others over the Internet (or a private network).



**Technical challenges with this POC**

* A stateful “smart contract” (chain code) implements the workflow of orders
  + Participants the functional consensus are identified dynamically, depending on the characteristics of (v) and (i)
  + Other participants are either non-voting or with zero-knowledge proof only capabilities
* Events
  + External events are managed by a simulator.
  + Events may directly “wake-up” the workflow.
* Messages
  + Messages are managed by a simulator. They are implemented as a persistent queue.
  + It is not necessary for this queue to be implemented as a blockchain (messages are managed by the TA participant only).
* GUI
  + Setting up a GUI would require a basic web server to be deployed at the investor’s node

**Events, messages and oracles**

The following events are relevant:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Event type | Event | Data structure | Origin | Handled by |
| Incoming Message | Cash transfer reception | Transfer ID(t)  Payer: Sender’s Cash bank account ID  Beneficiary: Recipient’s Cash bank account ID  Transferred amount and currency | Investor’s bank (simulated) | (ta) participant, acting as oracle |
| Incoming Message | Cash transfer Acknowledgement | Related Transfer ID | Investor’s bank (simulated) | (ta) participant, acting as oracle |
| Time | Time event (hour, end-of-day) | Timestamp | Platform caretaker acting as oracle (simulated) | All participants |
| Internal | Daily NAV availability | Investment vehicle ID(v)  NAV date | Fund accountant acting as oracle | All participants |

Outgoing messages are generated to effectively demonstrate the interaction with the existing infrastructure.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Event type** | **Event** | **Data structure** | **Origin** | **Handled by** |
| Outgoing Message | Cash transfer instruction | Transfer ID(t)  Payer: Sender’s Cash bank account ID  Beneficiary: Recipient’s Cash bank account ID  Transferred amount and currency | TA | Investor’s bank (simulated) |
| Outgoing Message | Security transfer instruction | Transfer ID(t)  Payer: Sender’s Securities account ID  Beneficiary: Recipient’s Securities account ID  Transferred amount and currency | TA | Investor’s custodian (simulated) |

Simplifications adopted with this protocol:

* Exchanges with the fund’s custodian are not represented: this protocol is assumed to be handled by the (ta) backstage
* Same regarding exchanges with the investor’s custodian: XXX

**GUI**

The GUI associated to the POC platform is intended to:

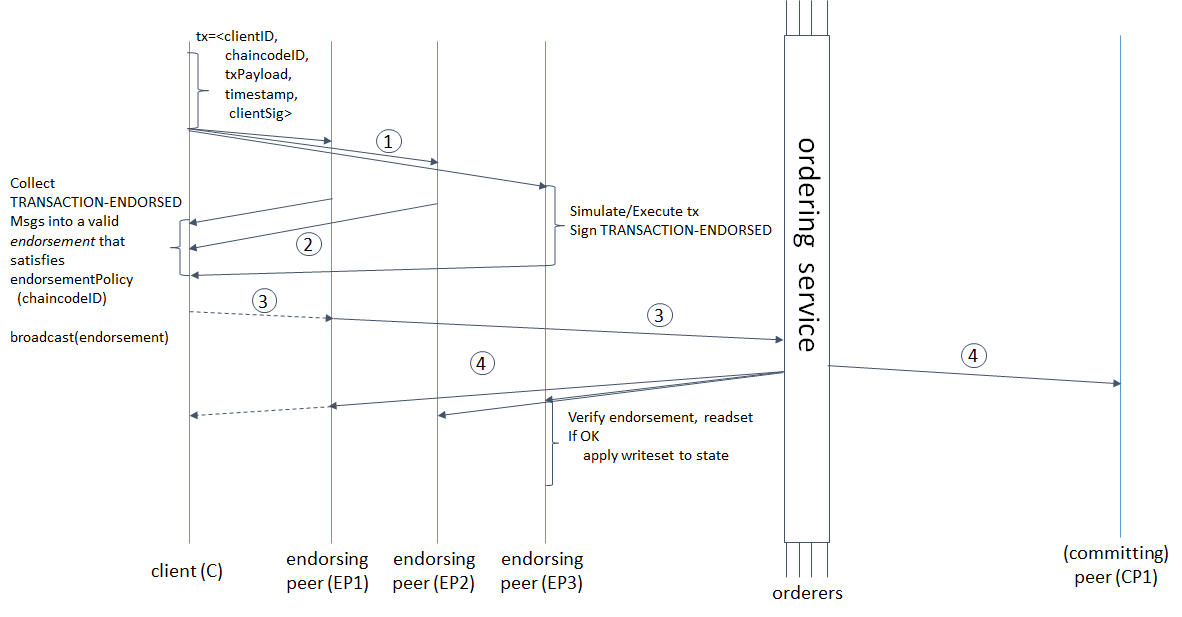
1. Allow the investor to enter an order
2. Visualize the different steps of the workflow, e.g. graphically
3. Show outgoing messages
4. Allow the investor’s bank to respond to platform messages and test different situations
   * Post expected responses to challenges
   * Post incorrect payment message
   * Fail to properly acknowledge redeemed cash

A single HTML page should be enough, provided it is periodically refreshed to display changes (or refresh events may be pushed by the platform, depending on available technical options).

Example:



**Appendix: the Hyperledger consensus workflow**



**Appendix: object model schema and examples**

JSON 4 schema specification

If NAV is exogeneous then we could price the orders in monetary amount + POC supposed to handle cash amounts

Nav is revealed after the order has been validated

Execution is perhaps misleading in the context of funds since this term is suited to products traded on a market.

En bon français on azdit “ordre centralisé » puis « ordre dénoué ». On ne parle jamais d’exécution pour un opcvm. Bon c’est du vocabulaire..a

1. In the limited context of the POC, the transaction type is redundant with the sign of the operation. Redundancy disappears once other transaction types are introduced. [↑](#footnote-ref-1)
2. Check roadmap to figure out if a more flexible endorsement policy is in view for forthcoming releases [↑](#footnote-ref-2)
3. i.e epsilon may be explained by NAV rounding. Alternative: the residue is immediately split among all investors as a separate “non investable” cash amount, but sounds overkill since this residue is only given back to investors when the fund eventually closes [↑](#footnote-ref-3)