**Workflows supported in POCS**

The table below shows how POCs progressively provide support for business workflows.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Mnemonic** | **Workflow** | **Main peers\*** | **Theme** | **First demonstrated with POC** | **Key simplifying assumptions** |
| **ORDER** | Orders | INV – Investor  TAG - Transfer Agent | Operations | 1. Transactions & Holdings Registry | European style open ended fund  No investor onboarding  No fund registration  No realistic settlement  No orders rescinded  No corporate actions or technical transfers |
| **ORDER** | Orders | INV – Investor  TAG - Transfer Agent | Operations | 101. Cross-jurisdiction orders |  |
| **ORDER** | Orders | INV – Investor  TAG - Transfer Agent | Operations | 104 Adjustments and transfers |  |
| **ORDER** | Orders | INV – Investor  TAG - Transfer Agent | Operations | 105. Corporate Actions |  |
| **NAV** | NAV publication | ACC - Accountant | Operations | 1. Transactions & Holdings Registry | Daily NAV, no valuation calendar management  No NAV rescinded |
| **REGISTER** | Fund registration | REG – Regulator  ISR - Issuer | Legal – Fund master | 1. Product Master – Registration |  |
| **AGREEMENT** | Domestic distribution agreement | REG – Regulator  ISR – Issuer | Legal – Fund master | 1. Product Master – Registration |  |
| **LEI** | LEI identification | LEI – LEI authority | Legal – Fund master | 1. Product Master – Registration | One scenario only: LEI is a non-participant entity |
| **SECID** | Security identification | SIA – Security identification authority | Legal – Fund master | 1. Product Master – Registration | One scenario only: SIA is a participant entity |
| **DVP** | Circulation on settlement system | STL – Settlement system oerator | Legal – Fund master | 1. Product Master – Registration | One scenario only: SIA is a participant entity |
| **SERVICE** | Investment services agreement | ISR – Issuer | Operations | 1. Product Master – Service agreements | All service providers mentioned in this contract are participant entities |
| **SETTLEMENT** | Orders settlement |  | Operations | 1. Orders settlement – SWIFT |  |
| **SETTLEMENT** | Orders settlement |  | Operations | 1. Orders settlement – Settlement scenarios |  |
| **SETTLEMENT** | Orders settlement |  | Operations | 107. Dividend settlement |  |
| **DISTRIB** | Distribution agreement | DIS – Distributor  ISR – Issuer | Intermediated dist. | 1. Intermediated Distribution |  |
| **DISTRIB** | Distribution agreement | DIS – Distributor  ISR – Issuer | Intermediated dist. | 102. Distribution fees management |  |
| **DISTRIB** | Distribution agreement | DIS – Distributor  ISR – Issuer | Intermediated dist. | 108. Trailer fees management |  |
| **ONBOARD** | Investor onboarding | DIS – Distributor | KYC | 1. Investor onboarding & KYC |  |
| **PARTY** | Party Agreement | REG – Regulator  CTK - Caretaker | Legal – Fund master | 116 – Party Agreement |  |
| **FISCAL** | Fiscal filings | ? | Legal – Filings | 115 – Fiscal filings |  |
| **KIID** | KIID production | ? | Legal – Filing | 106. Document production |  |
| **PRIIPS** | PRIIPS production | ? | Legal – Filings | 106. Document production |  |

**Progressively deploying the architecture with POCs**

The table below shows how the various chain codes defined by our architecture are progressively deployed with POCs.

The full list of chaincodes is provided in Appendix B document. Each chaincode implements a number of APIs to support workflows.

|  |  |  |  |
| --- | --- | --- | --- |
| **Chaincode** | **POC** | **Implemented APIs** | **Notes** |
| **Fund** | 1 | OrdersAPI  HoldingsAPI | Not in POC |
| 2 | LegalAPI | Not in POC |
| **RegFundDir** |  |  | Registration POC |
|  |  | This a private view (fund master) enriched with private data (e.g. mandate structures, links to internal CRM, …), shared between a FM and a specific Investor |
|  |  | Not in POC unless some “privately held funds” use case emerges (insurance?) |
| **DistFundDir** |  |  | Should distributor(s) be a peer? (Privacy issue)  I think not. However, distributor may participate the consensus service and perform certain query operations |
|  |  | Idea dedicated to a specific vehicle (not a fund) to model life insurance contracts invested in funds (special purpose POC) |
| **Party** |  |  | Interesting case of a chaincode for which peers evolve in time |
| **Investor** |  |  | Fund manager registers funds or direct clients as investors. The regulator may audit this record |
|  |  | The global list just keeps a crypto-seal to check investor identity |
| **Portfolio** |  |  |  |
| **Time** |  |  |  |

**Regulators and jurisdictions**

The platform is assumed to operate under strong regulatory constraints and to support multiple jurisdictions. Therefore, for any supported jurisdiction, its regulator – an oversighting authority – participates to the platform.

Note: the regulator role could possibly be assumed by another party, which then act as a proxy to the real regulator, and assumes all its responsibilities regarding other participating parties.

A jurisdiction generally corresponds to a sovereign country. There are however some special cases. It may be finer (e.g. Isle Of Man, Jersey…) or coarser (e.g. European Union).

Our design should provision the notion of sub-jurisdiction (e.g. EU / French / Epargne Salariale ; US/ 401K …)

**Funds**

A “fund” on the platform corresponds to an investment vehicle made available to investors. A “fund” chaincode handles all share classes (or compartments, i.e “sub-funds”) of a fund.

Its design is very flexible in order to support any kind of investment product and legal peculiarities. Remarks about the model:

* A fund is registered under one jurisdiction (registration workflow)
* A fund, sub-fund or share class may receive agreements to be distributed in several jurisdictions (*domestic* distribution agreement workflow)
* A single, cross-jurisdictional agreement, may trigger such agreement for several jurisdictions (e.g. UCITS passport). With UCITS, the agreement workflow becomes a simple notification process
* A fund chaincode is an autonomous entity which state describes:
  + Asset valuation history (net, possibly gross as well, in order to support some specific products)
  + Full legal description, including agreements - verifiable thanks to “crypto-seals” provided by various authorities
  + Some more descriptions: fiscal & commercial
  + Detailed investors holdings & history of transactions
  + A fund may cover several investment vehicles: e.g. a fund may maintain the state of several “fund shares” which differ only by fees, currency or dividend policy. A separate registry is maintained for each investment vehicle

Issuing a new product corresponds to the issuer deploying a new chaincode, which is then submitted to the registration & agreement workflows.

As an issuer may freely launch such chaincodes, it is necessary that an oversighting technical authority (caretaker, possibly delegated to some specialized entity) validates the code of a chaincode. This validation takes time and thus we suggest that it is implemented as an “out-of-band” process, using seals to be reused by other workflows (e.g. a technical authority may take a couple months to validate a new chaincode proposed by an issuer: a seal may be delivered to this chaincode for reuse by any subsequent deployment). As each fund is a distinct chaincode, one may derive any specific behavior from the “base” chaincode by adding some slight variations.

A fund is not necessarily a legal entity: it could be a private agreement with an investor (e.g. mandate, managed account). A fund does not necessarily get a security code (e.g. dedicated funds, French FCPE’s…).

**Participants and roles**

The platform operates all features made available to participants according to *functional roles* played by various actors.

Roles are universal and correspond to a job to be done. They do not depend on jurisdiction. It is very important to remember that a role comes with some responsibilities, which may differ with the jurisdiction.

Roles may be regulated (a regulator may enforce that some roles are allowed to certain registered parties only), under a given jurisdiction. A given party may operate with multiple roles, pursuant to the relevant regulation. Disintermediation is simply the result of a given party accumulating several roles, provided the regulation permits it.

Functional roles are listed in Appendix A document. An important point in our design is to support role delegation.

**Business & Legal Terms**

Some variations across jurisdictions remain to be clarified, in order to define the most accurate set of roles (our architecture may accept new roles to be defined in the future).

* issuer/promoter/sponsor/underwriter?
* registrar/trustee?
* fiducie (Lux, Can.)

**Directories, Private instances and enrichments**

A fund may be referred to by one or several fund directories.

Fund Directories are flexible objects which may be deployed for several purposes.

The main purpose is to list available chaincodes (so as not to forget the ID and lose the fund…).

A secondary purpose is to split data according to data privacy constraints (e.g. a regulator does not see detailed holdings).

Eventually, Fund Directories may be deployed as private instances shared by some platform members to define scopes. This feature may be used by any party to define a private scope and decide with whom to share it. For instance, a fund manager could deploy a fund directory of mandates and decide to share it with it customer.

The Directory model is flexible enough so that such private deployments may come with extra data structures (privately defined).

Applications of Fund Directories:

* products regulated by a regulator (scope: jurisdiction, enrichments: various agreements and identifying codes)
* products distributed by a distributor (scope: distributed products, enrichments: marketing documentation, referred to distribution agreements)
* products managed by an issuer
* …



Figure 1 Chaincode Directory pattern

**Peers vs participants to operations**

With Hyperledger, peers maintain a copy of a chaincode.

Among peers, “endorsing peers” pre-validate operations on chaincodes.

Other participants may initiate operations by proposing them for validation to the endorsing peers. This validation may apply business rules and access local chaincode data [we assume it is also possible to query another chaincode managed as a peer]. Such participants do not store a copy of the chaincode and must query it through an operation to access its data.

Eventually, a consensus service (voting on the order of transactions only) may cover a larger population of participants and with a simpler, more generic protocol [consensus participants – voters – do not see operation details].

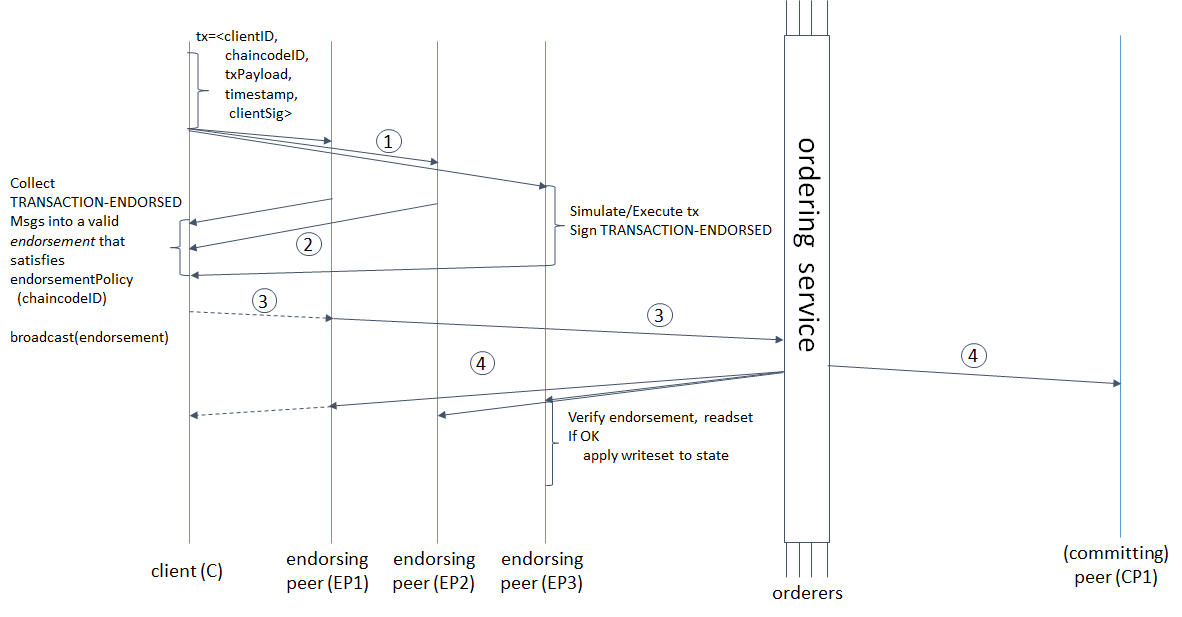


Figure 2Hyperledger Transaction Workflow

**Technical stakes**

* Data privacy
  + We would like to have distributors and institutional investors as peers of a fund.
  + But this poses a privacy issue
  + Unless equipped with a full-fledged granular privacy solution, we are forced to keep investors and distributors as non-peers. They may still participate the consensus service.
  + This is a long-term issue since this situation creates an unbalance between actors
* Maintaining consistency between chaincodes
  + Redundant data between a Directory and a Fund must be synchronized
  + A newly registered fund gets its chaincode referenced in Directories
  + One may assume the issuer is responsible for this sync
  + To make sure the system sets the proper incentive, a fund chaincode works only if the Directory is okay. Therefore, all endorsing peers participating to a fund chaincode must be able to verify the FundDirectory.
  + …
* Waking-up clients during workflow transitions
  + Events are posted between chaincodes. There is no support for pushing messages to SDK clients
  + An alternative design could be a polling carried on by clients, querying events queues managed by the chaincode (part of the state)
  + Event sample code: <https://www.ibm.com/developerworks/cloud/library/cl-ibm-blockchain-chaincode-development-using-golang/index.html>
* Role change during the life of a fund and more generally, chaincode upgrade
  + A possible design would be a “migrate” operation to migrate to a new chaincode under the new conditions, while keeping past data. Such an upgrade would be subject to a stronger consensus
* Key revocation
  + Any functional encryption scheme should take provisions for access revocation

**Appendix**





