**Orders earmarking**

The idea of order earmarking is to be able to aggregate orders according to various criteria. This is useful to figure out various intermediation fees. A useful criterion is the distribution channel: distributors may aggregate orders (and the resulting holdings) to figure out distribution fees. Typically, a distribution agreement smart contract would take a distributor tag as input and yield an aggregate holding amount on a fund (more details below).

Distribution channels may however be quite complex, as distributors and financial advisors may pile up along the distribution chain. Every actor in this chain might be eligible for a fee, either paid upfront (“loaded shares”) or as a trailer fee, usually paid back by the issuer on the management fees collected, then back again from distributor to sub-distributor.

Our smart contract should be able to accurately compute upfront fees as well as trailer fees, regardless of the length of the distribution chain. Every actor along this chain should be able to verify that holdings and calculations are correct.

Orders and holdings are thus “tagged” to mark an entry point for fees calculation. Tags must be designed such that any actor of the distribution chain may trust them, in order to avoid costly dispute resolution procedures.

We shall distinguish between “upfront fees”, which are calculated and settled with the order, and “trailer fees”, which are rebates periodically invoiced.

Every charged upfront fee is traceable and must have been, at some point, acknowledged by the end investor.

Trailer fees are not directly disclosed to the end investor (that"s precisely the point of trailer fees not to be disclosed…). However, orders clearly refer to distribution agreements involving trailer fees: the existence of a trailer fees agreement is disclosed, but not its terms.

**General idea**

Every order must have a tag zone filled with valid signatures. An order with an invalid tag zone is rejected by any FUND cc.

Tag zone is a linked list of signed tags. The first tag of the list MUST be signed by the party with the investor role.

Any link to a subsequent tag must be acknowledged by the owner of the previous tag in the chain, i.e. links between tags are signed. A tag may have several successors. Every tag has at most one predecessor.

Each tag is preceded by an “acknowledgment tag”.

**Earmarking model**

tag: {

currentTag: {

tagCategory: "investor",

tagOwner: {E-Cert},

markHoldings:"true",

agreementCC:"032345",

},

[

successorTag:{

tagCategory: "advisor",

tagOwner: {E-Cert},

agreementCC:"9993445"

}

],

tagSignature: "EZDSQQSFQSSFSFQF" /\* tag struct hashed & signed with investor PKI \*/

}

earmarking; [ { tag},…]

**Earmarking structure verification**

* Must have at least investor tag, possibly with empty successor (direct distribution)
* Investor tag"s certificate is consistent with investor identity (with possible delegation)
* For each tag, save "investor", there exists exactly one predecessor with a valid signature
* Chain ends when successor tag is empty
* Successor tags with no existence (“dangling” links) are considered as empty (e.g. investor allowed an intermediary to mark his order, but this capability has not been exploited
* Invalid tags (no signed predecessor, …) are ignored (further enrichment may be added in subsequent phases)

**Applications**

This mechanism provides many ways to “mark” the action of an intermediary on an order.

Each party along the intermediation chain must acknowledge, by signing a successor tag for a third party.

It is thus possible to identity many activities participating the formation of an order:

* Advisory
* Delegated execution (acting on behalf of)
* Allocation w/o execution
* Fund selection w/o execution
* Routing (e.g. platform access)
* …

Tag signers may participate the DLT or not. Anyhow, they are identified to the platform (with a certificate). This shall help independent advisors & consultants to operate with the platform without being full-fledged members.

Tag signing is a SDK API

Tag verification is both a SDK and CC API

Distribution agreements may refer to tags to establish their calculation basis.

Order tags are enriched as order pass through the different SDK layers (advisory, distribution, etc...). No further tag enrichment is performed once the order reached the fund object.

PTF chaincodes, when performing order routing, may enrich orders.

*Examples:*

* Direct distribution: investor fills its tag without any successor
* Investment advisory, direct execution: investor fills its tag then validates a successor for advisor, the advisor enriches the order with its own tag
* Investor chose one allocator, one execution advisor then delegate execution to another
* Advisor is related to an order routing distribution platform: investor signs for both
* …

**Impact on the data structure of holdings**

Since earmarking is a driver for establishing fees (upfront or trailer fees), holdings should be split according to such marks.

Accordingly, holdings structure maintained by a FUND cc should be split by (INV, earmarking) (we have as many entries in holding as there exist intermediation circuits).

Since many earmarking variations are not sensitive to holdings (e.g. upfront fees), the split is made according to tags which request it.

E.g. the tag structure becomes (optionally):

currentTag: { tagCategory: "distributor, tagOwner: {E-Cert},markHoldings:"true"}

Holdings are always split at least by investor (and btw, there must always be an "investor" tag, which defaults to markHoldings:"true").

**Triggering fees and enriching the order structure**

We discuss here intermediation fees, not management fees, which are exogeneous (provided by fund accountant).

Intermediation fees are described in a separate structure not controlled by the fund CC. Fees descriptions may or may not refer to a specific FUND, however.

*Upfront fees (“commissions de souscription / rachat”) ("subscription or redemption fees payable to third parties ")*

* Fees for “loaded shares”
* At tag sealing time, check if a distribution agreement has been signed between predecessor and successor tag owners
* If this is the case, invoke the fees calculation service(how?)
* Upfront fees are expressed as a percentage of the final settled amount + possible fixed part
* Fees vs VAT relationship must be explicit
* Before order is submitted a recap may be displayed to the investor
* Fees are owed to the tag owner and paid by:
  + Either the tag predecessor owner: cascading fees
  + Or (more likely), the owner of the root tag (investor): stacked fees

*Trailer fees (rebates)*

* Fees paid by one party collecting other fees, back to another party
* Common use case: a share of fund management fees is paid back (as rebates) to the distributor
* Mark the order with the trailer fees agreement ID

Note: upfront fees may be subject to rebates (e.g. cascading fees rather than stacked fees).

Note: upfront fees are an intermediary-only affair, excluding the issuer. Trailer fees may be based on management fees (i.e. the agreement is between issuer and distributor)

*Other kind of fees: deferred fees*

* Performance fees/success fees: fees calculated after a grace period (# performance fees for fund manager)
* Carry fees (?)
* Mark the order with the fees agreement ID for future reference

*Fees for the issuer*

* Unless the issuer has a distributor or advisor role, there is no intermediation fee for the issuer
* If any such arrangement does indeed exist, proceed with rebates or reverse rebates (negative fees)
* Issuer enrich the final tag structure to endorse trailer fees: this enrichment comes with the final order version (either as simulated or real "invoke" order)

*Retained fees (“commissions acquises à l"OPCVM") (" fees payable to the fund ")*

* Such fees are perceived by the fund and collected by the fund custodian
* They are reported by the fees service (QUERY) exposed by the FUND cc (most of the time, it is zero).
* The FUND cc may choose more complex rules than the ones currently prevailing, e.g. variable fees depending on the time of the day (e.g. extended cut-off, …), order well planned in advance, etc…
* At order simulation time / order initiation, the FUND fees service is called to inquire about such fees and fill the indicative fees section of the order (indicative since: (i) final amounts are not known, (ii) fees rates may change between order submission and settlement (this should be non blocking, but trigger an event notice in the order fees structure)

*Simulation vs real and fees changes*

Simulated orders compute the full upfront fees structure and trailer fees agreements references.

However, agreements and retained fees may be revised across time (with an effective start date).

* When revising fees (retained fees at the FUND cc level or distribution agreements), SDK should check if any pending orders are impacted and if so, warn the modifier of the impact. Still, the revision is legit
* Orders with a final fee different from the simulated fee should include a notification of this change

**Fees structure model**

orderFees: {

feesStatus: 'simulated|final',

upfrontFees: [

fee: {

feeType: 'advisory|allocation|execution|retained|…' /\* Descr. in schema \*/

feePaidBy: {party}

feePaidTo: {party}

feePaymentType: {internalPTF|external}

tagSignature: {signed hash of reference tag}

feeAmount: '36.33', feeVAT: '00.00',

feeAmountCurrency: 'USD'

}, …

]

trailerFees: [

fee: {

}, …

]

}

**Order formation workflow**



In this example, the business arrangement between parties runs as follows:

* An advisor counsels an investor: fees are paid upfront
* The advisor uses a distributor as a routing platform: fees are rebates based on holdings
* The advisor has a trailer fees agreement with the issuer: fees are rebates based on holdings

This results in the following order structure at step 6:

* Tag structure: involved parties acknowledge to be part of certain fees arrangements
* Fees structure: disclose upfront frees, disclose reference to trailer fees

earmarking: [

/\* reference to advisory agreement: investor acknowledges advisor"s role \*/

tag: {

currentTag: {

tagCategory: "investor,

tagOwner: {E-Cert investor},

markHoldings:"true",

},

[successorTag:{

tagCategory: "advisor",

tagOwner: {E-Cert advisor},

agreementCC:"032345"

}

],

tagSignature: "EZDSQQSFQSSFSFQF" /\* tag struct hashed & signed with investor PKI \*/

},

/\* ref. to order routing agreement: advisor acknowledges distributor's role \*/

tag: {

currentTag: {

tagCategory: "advisor",

tagOwner: {E-Cert advisor},

markHoldings:"true",

agreementCC: "032345" /\* ref. to advisory agreement \*/

},

[successorTag:{

tagCategory: "distributor",

tagOwner: {E-Cert distributor},

agreementCC:" 346676778" /\* ref. to order routing agreement \*/

}

],

tagSignature: "XYZ" /\* tag struct hashed & signed with advisor PKI \*/

},

/\* ref. to distribution agreement: distributor declare its own role \*/

tag: {

currentTag: {

tagCategory: "distributor",

tagOwner: {E-Cert distributor},

markHoldings:"true",

agreementCC:"346676778", /\* ref. to order routing agreement with advisor \*/

},

[successorTag:{

tagCategory: "distributor",

tagOwner: {E-Cert distributor},

agreementCC:" 566778" /\* ref. to distribution agreement with issuer \*/

}

],

tagSignature: "ABCD" /\* tag struct hashed & signed with distributor PKI \*/

}

]



Tag structure after final order response by FUND cc:

earmarking: [

/\* reference to advisory agreement: investor acknowledges advisor"s role \*/

tag: {

currentTag: {

…

tagSignature: "EZDSQQSFQSSFSFQF" /\* tag struct hashed & signed with investor PKI \*/

},

/\* ref. to order routing agreement: advisor acknowledges distributor's role \*/

tag: {

…

tagSignature: "XYZ" /\* tag struct hashed & signed with advisor PKI \*/

},

/\* ref. to distribution agreement: distributor declare its own role \*/

tag: {

currentTag: {

tagCategory: "distributor",

tagOwner: {E-Cert distributor},

markHoldings:"true",

agreementCC:"346676778", /\* ref. to order routing agreement with advisor \*/

},

[successorTag:{

tagCategory: "distributor",

tagOwner: {E-Cert distributor},

agreementCC:" 566778" /\* ref. to distribution agreement with issuer \*/

}

],

tagSignature: "ABCD" /\* tag struct hashed & signed with distributor PKI \*/

},

/\* ref. to distribution agreement: issuer (FUND cc) validates trailer fees agreement \*/

tag: {

currentTag: {

tagCategory: "issuer",

tagOwner: {E-Cert issuer},

markHoldings:"false",

agreementCC:"346676778", /\* ref. to order routing agreement with advisor \*/

},

[], /\* no further chaining \*/

tagSignature: "1678899" /\* tag struct hashed & signed with issuer PKI \*/

}

]

In this example, fees arrangements stop at the issuer's: more involved use-cases could specify an even lengthier chain managed by the FUND cc tag enrichment phase: e.g. certain orders are marked differently to refer to other agreements.

**Fees transparency**

Fees charges to the investor. What do we want?

1. For every order, upfront fees are known (at least %) before submitting the order
2. For every order, upfront fees retained by the fund may be known (as %)
3. Trailer fees are not disclosed to the end investor. Perhaps a disclaimer mentioning the existence of an agreement could be automated (on option)
4. Every party involved in a distribution agreement may check the collected fees an may inquire FUND cc for audit/justification

Ideally, the investor could simulate an order at any time to figure out actual fees. The order data structure carries on tags and indicative fees. Every tag owner must alter the contents of the order with its tag.

Several available options for fees settlement:

* At settlement time, TAG (transfer agent) executes the order (i.e. post instructions)
* If the order is tagged, TAG inquires the various fees agreements involved
* TAG enrich the order structure with final amounts, including fees and (possibly) VAT
* Tags must also refer to agreement CC

If agreement refers to a trailer fee, TAG invokes agreement CC to provision trailer fees. If agreement refers to “exotic” fees (such as success fee), TAG invokes the agreement which will figure out if any amount is actually triggered.

Taking our previous example, the fees structure resulting from order simulation or order invoke, is as follows:

orderFees: {

feesStatus: 'final', /\* response to invoke order \*/

upfrontFees: [

fee: {

feeType: 'advisory', feePaymentType: 'internalPTF',

paidBy: {investor}

paidTo: {advisor}

tagSignature: {signed hash of reference tag}

feeAmount: '36.33', feeVAT: '2.12',

feeAmountCurrency: 'USD'

},

fee: {

feeType: 'retained', feePaymentType: 'internalPTF',

feePaidBy: {investor}

feePaidTo: {fund}

/\* retained fees specified as part of distribution agreement \*/

tagSignature: {signed hash of reference tag}

feeAmount: '36.33', feeVAT: '00.00'

feeAmountCurrency: 'USD'

},

]

/\* trailer fees amounts are not disclosed, but existence of such agreements is \*/

trailerFees: [

fee: {

feeType: 'routing',

feePaidBy: {distributor},

feePaidTo: {advisor}

tagSignature: {signed hash of reference tag}

},

fee: {

feeType: 'distribution',

paidBy: {issuer}

paidTo: {distributor}

tagSignature: {signed hash of reference tag}

},

]

}

**Fees invoicing and payment**

* The objective is to streamline the intermediation business completely with a full STP fees invoicing workflow
* Fees charged to the investor are settled with every order, each fee portion is cleared with the order (upfront fees, retained fees). This make a large distributed transaction. What happens if one single party fails?
* The order fees data structure allows to generate an order receipt (an invoice)
* Fiscal features of the PTF cc may interpret this to figure out the fiscal result of the PTF



*Trailer fees / Deferred fees: provisions, invoicing, payment*

* Agreement CC with trailer fees are **not** continuously updated by TAG with each order (too slow). However, every settled order invokes the agreement CC's involved to check if the FUND cc is already referred in the list of FUND covered (therefore, the state of the agreement only changes when the FUND is not already covered by the agreement)
* It is important that the list of FUND cc's to be queried is maintained in sync with orders
* However, it could be excluded from the settlement phase and added to the initial order posting phase (to relieve the settlement phase from too many CCs to be synchronized)



* At invoicing time, distributor invokes the Agreement CC which inquires all concerned FUND cc's
* Accounting times: Agreement CC contracting parties (e.g. issuer, distributor) may inquire the cc for the amount to be provisioned
* At invoicing time (e.g. quarterly), Agreement CC generates an invoice, with due payment date
* On due date, payment instructions are posted and cleared
  + All contracting parties must have their own PTF cc to manage their cash and payments (just like investors)
  + Fails on payments trigger a fail handling workflow (dunning)

*Invoicing /payment for non-participating parties (e.g. independent advisors…)*

* Billing / payment process is marked as "external" in the Agreement cc
* Invoices are generated as PDF docs by the Agreement cc.
* Payments may be checked (by an operator as the beneficiary) or not (the agreement CC does not perform any check)
* Upfront fees which would be external must be handled internally by a participating member with a PTF. Most likely, a distributor will handle an account for each of his sub-distributors/advisors in its own PTF.

**Payment address**

* Internal payment's address is a valid PTF chaincode, possibly with a segregated account
* External payment address is a surrogate PTF chaincode, mandatorily with a segregated account
* In case of segregated accounts (e.g. the owner of the PTF is the account holder for the final account owner), a valid account holding convention must have been signed (this signature is part of the PTF cc model)

**Disputes**

* The system ensures that every order is consistent with all existing fee agreements
* The dispute rate should therefore be very low
* However, disputes may arise because of:
  + A bug
  + A wrong configuration is the agreement
  + An unduly given acknowledgement, possibly without the explicit consent of the party
* Most of the times, disputes should be settled by a simple inquiry process
  + Inquiring the agreement to disclose terms and scope
  + Inquiring funds to disclose holdings
* Following such an inquiry, parties may choose to settle the dispute amiably: e.g. pay a compensation

Delegating tag enrichment