PART: A Protocol for Asset Receipt and Transfers on Contract-Based Distributed Ledger v0.5

Currently Mutual Funds Only

James Partridge

January 26, 2018

Abstract

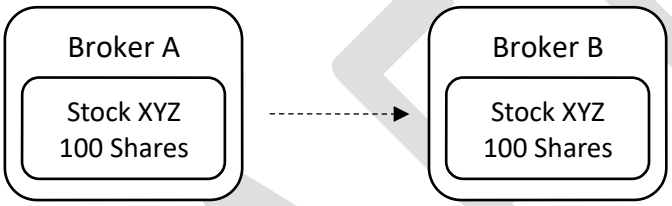
The following document describes a procedure to simplify the current ACATS (Automated Customer Account Transfer Service) system with the implementation of a contract-based distributed ledger. This document is solely intended to serve as a broad proposal regarding the possibility of a secure and efficient means of transferring mutual funds with existing industry infrastructure. As it stands, this document does not constitute a blueprint for all asset transfers; though with the adoption of distributed ledger accounting among the broker-dealer and clearing/ settlements industry, the following concepts may be drawn and expanded upon freely, and encouragingly.

Introduction

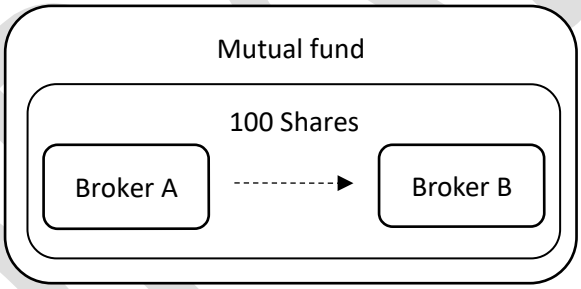
The establishment of the Depository Trust Company (DTC) in 1973 and National Securities Clearing Corporation (NSCC) in 1976 allowed for a revolutionary change within financial markets: ownership of securities became information, and information was transferred from paper certificates to electronic registrations. What previously represented a risk of lost or forged documents, tedious processing, and stagnated information flow was transformed to a relatively uniform system of electronic record keeping and transfers. These transfers became known as ACATS.

Mutual funds are unique in that ownership of a fund, in this case we'll use the example fund PARTX, is not held in street name at a broker-dealer. If one wanted to transfer PARTX from one broker to another, it is not picked up and transferred like an individual stock. Instead, ownership of PARTX is simply overwritten at the fund family.

Typical Equity Transfer



Typical Mutual Fund Transfer

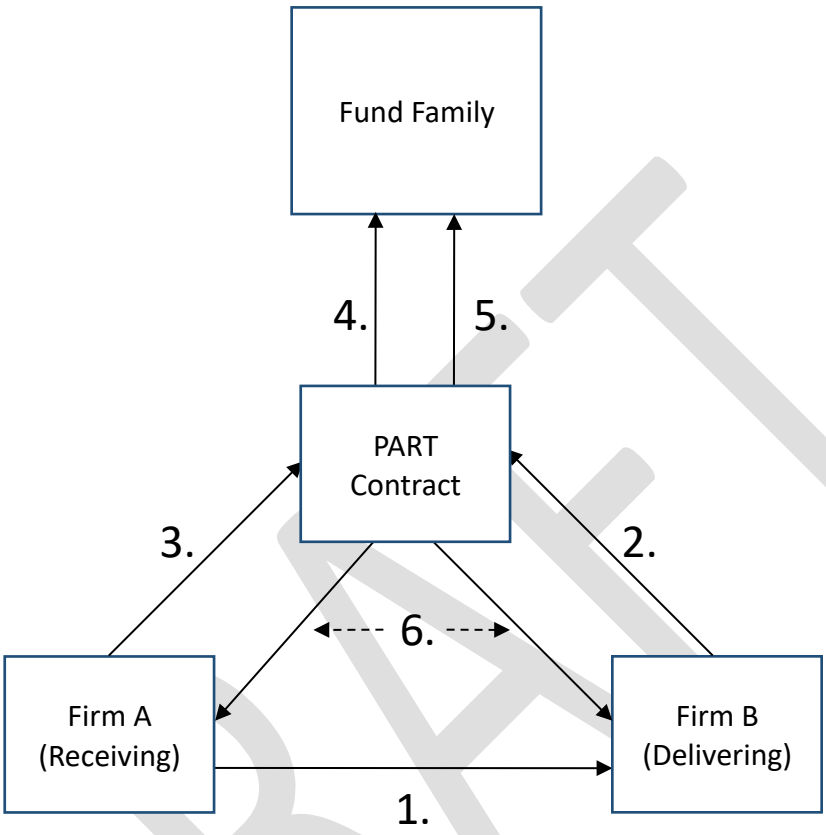


While innovative and its design and successful in its original goal, this process has come under continuous pressure as the speed of electronic transactions have quickened their pace. At their core, any type of financial transaction relies on two principles: trust and record keeping. Trust in the counterparty that the assets are legitimate, and a mechanism to reliably remove assets from the sender and appropriately deliver assets to the receiver. The following document details a means to improve the latter.

This protocol is the suggestion for existing institutions that safeguard assets to continue doing so; though for the reregistration of mutual fund shares to be accomplished via a lightweight, transparent (if desired), and independent distributed ledger. Such independence would require a ledger maintained and monitored by a centralized organization. This is a proposal that will be left to the industry to determine responsibility, but can be easily maintained by the SEC, its derivatives, or the collective power of FINRA organizations, etc. This is not a suggestion for the creation of a cryptocurrency, but only that of a private distribute ledger. Prior knowledge of distributed ledger accounting and ACATS may be necessary to fully understand this proposal.

Process and Application

The following is a simplified process to conceptualize the role of a contract based agreement on a distributed ledger. A more robust documentation of the process (along with steps 2 and 3) will be detailed within "Further Analysis."



1. The Receiving Firm submits a request of for shares to the Delivering firm.

2. The Delivering firm confirms the client's identity, verifies ownership of shares, and broadcasts a smart contract. This is known as the PART Contract.

3. The Receiving firm locates the smart contract, and accepts or rejects the request.

4. Upon acceptance, the contract submits a request to move shares at the fund family. This is a request to overwrite the current registration of the Delivering firm with that of the Receiving Firm.

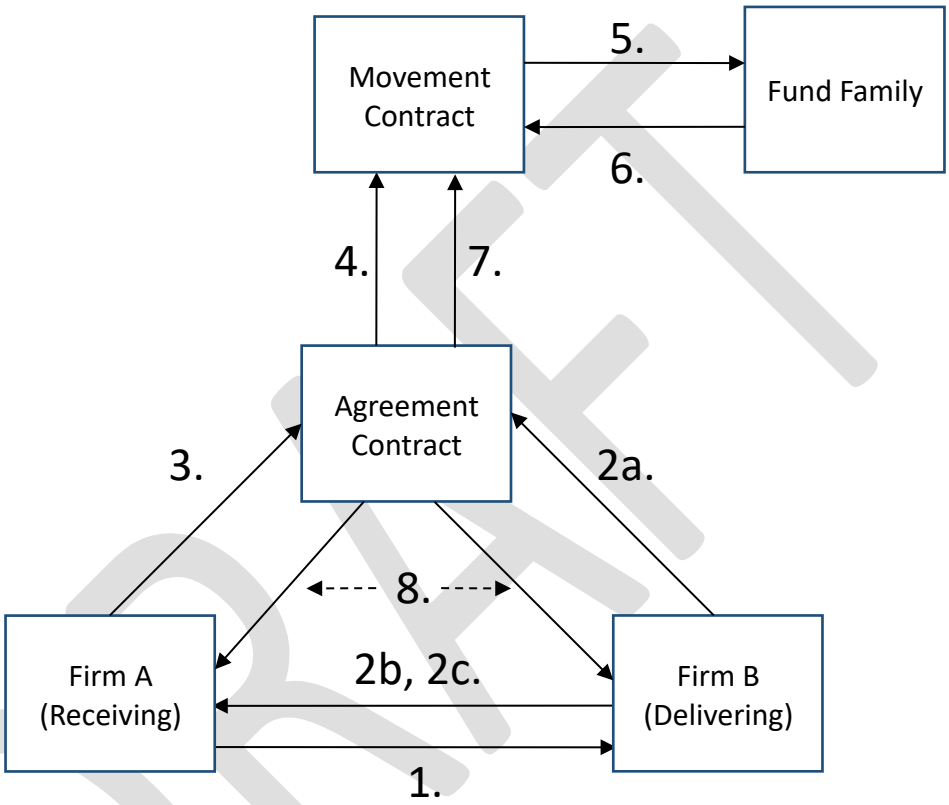
5. After a predetermined amount of time, blocks, etc, the contract looks for a successful or failed movement of shares at the Fund Family.

6. Regardless of the outcome, the PART contract is published to the distributed ledger. Information is then electronically communicated to each firm concerning the successful (or failed) movement of shares, and the firms' books are updated accordingly.

Further Analysis

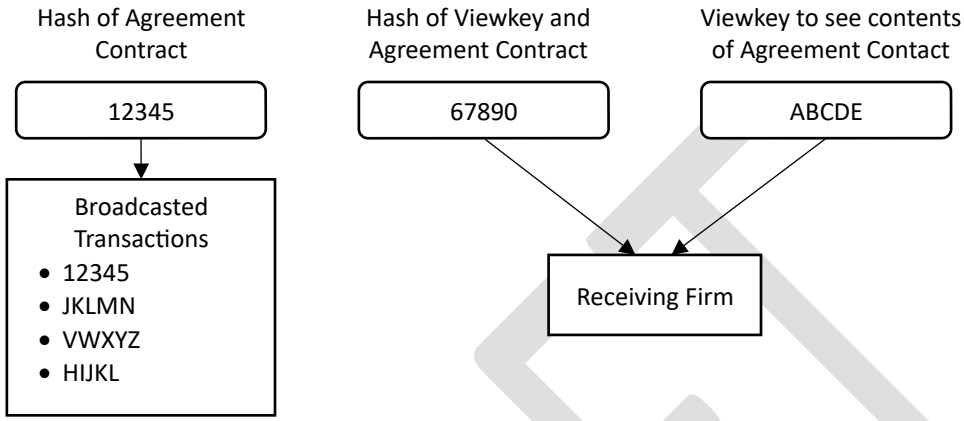
Detailed Process Map

A means for delivering firms to securely broadcast the smart contract (and for receiving firms to locate it) can easily be accomplished using any hashing algorithm and any preferred method of communication. Regardless of the chosen method, cryptographically secure requests to transfer shares can be achieved through the following process.



1. The receiving firm notifies the delivering firm its desire to receive shares. This can be done via any communication method chosen, or can be routed over the NSCC, etc.
2.
    a. The delivering firm creates the request under an established set of formatting rules (all client information can be formatted according to existing NSCC requirements), hashes the request, and creates a viewkey for the contract contents to be seen (this would be similar to the current state of Monero wallets (reference Monero white paper)). This hashed request is then broadcast over the NSCC, and is known as the Agreement Contract.
    b. The hash (identical to the Agreement Contract that is currently being broadcast to the NSCC) is then hashed again with the viewkey.
    c. Once the signed request is broadcast to the NSCC, shares tied to the smart contract are locked and cannot be transacted upon. The viewkey, as well as the hash of the request *with the viewkey,* can then be provided to the receiving firm over any method chosen, or via a Diffie-Hielman exchange. **This step is crucial to the transfer process: a rejection to transfer shares by the delivering firm can still be included in the Agreement Contract with a specific error message. Additional information can be found within "Rejections" on page X.**

3. With the viewkey, and hash of the request with the viewkey, the receiving firm can securely locate the original request that has been broadcast anonymously to the NSCC. This is simply done by hashing all broadcasted transactions with the provided viewkey. If any of these hashes match the provided hash of the request *with the viewkey,* the receiving firm will know which smart contract had been broadcast by the delivering firm, and can view the contents of that contract with the viewkey. Once located, the receiving firm signs the Agreement Contract. A simplified demonstration is provided below:



The receiving firm hashes all broadcasted transactions with ABCDE (ABCDE is hashed with 12345, JKLMN, VWXYX, HIJKL, etc). If the result of one of these is 67890, receiving firm knows it can use ABCDE to view 12345.

4. The Agreement Contract authenticates the receiving firm's signature, verifies that the request to transfer has not expired, and verifies the delivering firm's signature. Upon verification that all instructions are in good order, the smart contract recursively creates a Movement Contract with instructions to move assets. Upon creation of the Movement Contract, the Agreement Contract becomes read only and cannot be modified.

5. The Movement Contract generates any and all files necessary for the fund family to interpret the request to transfer shares. The Movement Contract submits the request to the fund family among the existing channels that assets are transferred. This process can implement the current methodology of 018/019 records, etc.

6. Once fund family receives the request from the Movement Contract, the reregistration of assets is executed. If the request cannot be completed (for whatever reason) the request is rejected. The result of this transfer is then transmitted back to the smart contract. **This step is crucial to the transfer process: any rejection can return a specific error message that will void and close the contract. Additional information can be found within "Rejections" on page X.**

7. Upon receipt of the transaction, the Movement Contract is completed and movement (or failure) of shares is recorded on the distributed ledger. Until this step, shares have only transferred/failed to transfer at the fund family; no shares have been rerecorded/updated at the delivering and receiving firms.

8. After a predetermined number of blocks, time, or other signaling event determined by the industry, the Agreement Contract can call on the Movement Contract to view the results of the transaction. Depending on the result, the Agreement Contract can update the contrafirms' books accordingly (similar to the current settlement of mutual fund MFC file), and have distributed and verifiable record of the transaction.

It is important to note that only step 7 is written on the distributed ledger, regardless of whether reregistration at the fund family is successful or not.

Proposal for Separated Agreement and Movement Contracts

The primary purpose for accomplishing mutual transfers with smart contracts is to prevent mutual fund ACAT fails to receive (FTR) and fails to deliver (FTR). Presently, these occur when ACAT is submitted to the NSCC, but the transfer does not actually occur at the fund family, and these fail at a rate of around _____ percent. With separated contracts for the agreement and movement of transfers, the distributed ledger can be lightweight and provide for additional security.

A failure of the Agreement Contract would prevent creation of the Movement Contract, and allow for contrafirms to know the transfer would fail before it was attempted. A failure of the Movement Contract would allow for contrafirms to know the transfer did not occur at the fund family, allow release the funds back to the delivering firm held in the agreement contract, as well as provide the receiving firm with a reject reason. This structure prevents allows only the transfer or failure to transfer of shares within the Movement Contract to be written to the distributed ledger.

Separate contracts also allows for increased security during the transfer process. After the Movement Contract is created recursively from the Agreement Contract, the former can be locked for a predetermined number of block confirmations, while the latter can be converted to read-only. If, after a certain number of confirmations and no movement of shares has occurred at the fund family, the Movement Contract can become invalidated and written to the distributed ledger as such. Additionally, implementation of the STATICCALL (Op Code 0xfa) function would allow the now read-only Agreement Contract to query the Movement Contract to look for a result. The read-only Agreement Contract could then let both contrafirms know whether the transfer has occurred successfully or not, and its answer would be based on the distributed record of a successful (or failed) transaction, only viewable by the two contrafirms.

Compatibility with Existing Infrastructure

As the process currently stands, the implementation of this protocol shares the most similarity with networked level3 funds (though additional recursive contracts could be programmed to account for funds held within an omnibus registration).

Many of the existing file types associated with mutual fund transfers can be incorporated and binded within the Agreement and Movement Contracts themselves (b50s, b52s, AT record, FR record, 018 record, 019 record, etc). The Agreement Contract would be required to accept file types currently sent by broker-dealers, as well as send settlement files to broker-dealers to after shares have transferred according to the Movement Contract, or release shares if they have not. The Movement Contract would be required to produce file types currently accepted by fund families, and communicate the result at the fund family to the Agreement Contract.

Perhaps the most significant advantage of this protocol is the elimination of cash reconciliation contained within MFC files. Due to the nature and dependence of the two contracts, a confirmation (or failure) of share movement is not provided and published to the distributed ledger until the event has occurred at the fund family (or shortly thereafter). This dependence allows for firms to confidently enter into agreements to transfer shares without the need to provide cash should a firm fail its obligations, and also allows firms to update their records according to a published (though cryptographically obfuscated if desired) distributed ledger.

Rejections

Any rejections can be accomplished by utilizing something similar to the "throw" function within Solidity 0.4.9 of the Ethereum VM combined with an independent relaying of the reason, or more simply with the "require" function (Op Code 0xfd)[1] in a future version Solidity employing RETURNDATASIZE and RETURNDATACOPY to relay the reject reason. The usage of these functions can be implemented in both the Agreement and Movement Contracts. A rejection of the Agreement Contract would return a reasoning to the requesting firm without writing any transaction to the distributed ledger. A rejection of the Movement Contract would be written to the distributed ledger and return a reason for the rejection to the contrafirm, but would prevent a mutual fund ACAT to fail according to the current procedure.

Security

If deemed necessary by the industry, the requestor and volume of shares transacted upon in the Movement Contract can be further obfuscated. View keys can be generated by the recursive creation of the Movement Contract and provided to the owner(s) of the Agreement Contract, and a method would similar to ring signatures and ring-confidential transactions within Monero could be implemented (see MONERO/RING SIGS/RING CT REFERENCE). This would allow for multiple Movement Contracts to be grouped and signed according to sending or delivering firm, security type, or some other method determined appropriate, without revealing the identity or number of requestors. This can obscure the contract details to anyone without the viewkey generated by the Agreement contract.

If desired by the industry, and if information is bundled within ring signatures as detailed above, the connectivity of the Movement Contract and the Agreement Contract can separated completely. Similar to the process of broadcasting the Agreement Contract in steps 2a – 2c, viewkeys can generated by the Agreement Contract, and each signature that compiles into the Movement Contract can be hashed with a string of randomly generated information. This hashing can then be shared with the Agreement Contract via a Diffie-Hellman exchange. The Movement Contract can be written such to hash the total amount of shares transferred and rejected, along with the random information corresponding to each lot of shares. With the shared string of random information, and because the Agreement Contract is never written to the distributed ledger, each Agreement Contract could be "de-linked" from any movement of shares, and a completely anonymous transfer can occur at the fund family with any possibility of revealing who had requested it (at least to any possible outside party). This would allow the authors of the Agreement Contract to prove to each other the individual request was successful or not based on a distributed, but anonymous, ledger.

Conclusion

TBD

References

Ethereum Yellow Paper: “ETHEREUM: A SECURE DECENTRALISED GENERALISED TRANSACTION LEDGER EIP-150 REVISION,” Dr. Gavin Wood. http://yellowpaper.io/

Youtube Video detailing Byzantium Upgrades: “Ethereum Byzantium Breakdown,” https://www.youtube.com/watch?v=BSkUGfilJCo

NSCC Fund/SERV Reject Codes: “Fund/SERV Reject Codes,” https://dtcclearning.com/index.php?option=com_docman&view=document&layout=default&alias=1526-fund-serv-reject-codes&Itemid=852

http://www.ust2.com/pdfs/ssc.pdf

The following example assumes an ability for the receiving firm’s request to be properly broadcasted and matched with the delivering firm. As such, whichever medium chosen for broadcasting smart contracts only serves as a

means to direct client requests without possession or knowledge of the contract contents**. A more detailed proof of concept will be detailed under "Matching and Broadcasting Smart Contracts" on page X.**

It is important to also note that each mutual fund within the transfer request will establish a separate smart contract, though this can be automated.

1. Firm A (receiving firm) creates a request to transfer all or partial balances from Firm B (delivering firm) to Firm A, and signs the smart contract with their private key. This is known as the AGREEMENT contract. The AGREEMENT contract is and provided to and confirmed with Firm B.

latter can be locked for a predetermined number of block confirmations.

the AGREEMENT contract

to check for successful/failed movement of shares at the fund family.

With the movement contract permanently written to the blockchain whether successful or not, an AGREEMENT contract that could not be altered would only be able

This would allow for the MOVEMENT contract to be written to the Ethereum blockchain independent of the agreement between to fund families, if deemed necessary.

rather than the single contract delivering information to each contrafirm following reregistration,
If it is determined easier by the industry, the delivery of information directly from the AGREEMENT contract to the fund family can be implemented, though it may introduce certain risks ascribed in the following paragraph.

the contract between two firms can then call upon this reregistration contract with the query-only STATICCALL function of Byzantium within Solidity 0.4.x (any upcoming version of Solidity that employs the STATICCALL function).