Tanay Trivedi and Padraic McAtee

PH 360

Project Proposal

Derivatives are complex financial instruments created between two institutions that offer specialized exposure and payoff functions for isolating and hedging risk. They were invented for commodity based businesses like farmers and merchants whose livelihood depends on the price of their good. These businesses typically are paired in contracts with speculators who are willing to take on risk in order for future profit. Creating networks that allow institutions to safely engage and execute either side of these contracts is a nontrivial problem in consensus, because left to their own devices institutions will usually not be able to agree on the value of a contract, how much debt is to be exchange, and how to centralize payments such that the number of payments a single institution has to make is minimized when thousands of contracts exist. Blockchain is an ideal technology to solve a problem which is currently solved by centralized counterparty institutions, which take an enormous amount of fees to essentially create consensus and execute contracts. If derivatives are successfully settled on the blockchain, Goldman Sachs estimates that $10 billion in transactional fees can be saved every year.

The concept of settling derivatives on the blockchain is not unique to our application. The concept was discussed in an extensive, mostly legal document published by Fordham Law that details what a solution would like look, how traders would use it, how risk management personnel would evaluate it, what legislation would need to adjust to accommodate it and how the blockchain would need to function in order to work in this space. It also details what features the solution would need to fully subvert the current central counterparty establishment.

The Depository Trust and Clearing Corporation (DTCC) is a traditional accounting and clearinghouse company that is developing a blockchain derivative settlement scheme for the largest dealers in this space. DTCC's solution uses Solidity to settle debts through smart contracts. According to blockchain related press, DTCC was able to put most of the "business logic", which is to say the valuation of contracts and trading infrastructure, on the blockchain.

A multi-agent systems application embedded in this project has to do with the minimization in number of transactions that parties in a system need to perform. Each major broker-dealer, like JP Morgan or Citibank or Goldman Sachs, may have hundreds or thousands of derivative contracts in existence at any time with numerous other organizations. Having to pay each contracts debt separately would incur hefty fees in the current transactional system, as most derivative settlement projects have accepted that actual payment for assets happens off the chain in more traditional electronic communications. Instead, we can work to make the list shorter with respect to number of transactions and consolidate debt into larger, singular transaction. Take this example debt network:

$$
\begin{array}{lcl}
\text{Alice} & \xrightarrow{10} & \text{Bob} \\
\text{Alice} & \xrightarrow{20} & \text{Dick} \\
\text{Carol} & \xrightarrow{30} & \text{Bob} \\
\text{Carol} & \xrightarrow{40} & \text{Dick}
\end{array}
$$

If each debt was paid separately, there would be 4 payments. A more optimal network (in terms of number of transactions) is:

$$\begin{array}{lcl} \text{Bob} & \xrightarrow{30} & \text{Alice} \\ \text{Bob} & \xrightarrow{10} & \text{Carol} \\ \text{Dick} & \xrightarrow{60} & \text{Carol} \end{array}$$

By saving one transaction, we have saved some transactional costs in this network. This action of reduction is called “netting”. Centralized counterparties that currently settle derivatives net in a centralized manner. A decentralized solution like ours will need to do this as well. The algorithm for doing so in psuedocode looks like this:

Step 1. Determine the balance for each person.

Observe that the sum of all balances equals zero.
Let P be the total amount of positive balances, and N the total amount of negative balances. Hence, $P = -N$.
The minimum total amount to be transferred equals P.

Step 2. While there is still someone with a nonzero balance, do:

Step 2a. Select a person A with a negative balance $S < 0$, and a person B with a positive balance $T > 0$ (these exist).

Step 2b. Let M be the minimum of $-S$ and T. Hence, $M > 0$.

Step 2c. Include the transfer $A \xrightarrow{M} B$ in the settlement.

Step 2d. Increase the balance of A by M and decrease the balance of B by M (the total balance remains zero).

Observe that after Step 2d, at least one of A and B now has balance zero.

Step 3. All balances are zero, hence the included transfers settle all debts.

The total amount transferred equals P, and hence is minimal. The repetition of Step 2 terminates, because in each iteration at least one nonzero balance is reduced to zero. Therefore, the number of transfers is at most N. In fact, it is at most $N - 1$, because the final two nonzero balances cancel each other in a single transfer.

Our proposed solution follows the standard daily workflow of a clearinghouse. At the start of the trading day, contracts that have expired or have otherwise become invalid are removed from the pool of active contracts in what is known as “pruning” contracts. Once this is done, orders can be accepted and logged over the course of the trading day. An order is a request from a market participant to buy or sell a particular derivatives contract, defined by the type of contract (option, future, etc.) as well as the underlying commodity. The order will also specify time until expiry, the price of the contract and underlying, and the notional amount in USD that the contract is worth. At the end of the trading day, matching orders between buyers and sellers are paired to create contracts. These contracts as well as already existing contracts are then re-evaluated based on the changed time until expiry and any changes in the spot price of the underlying. This procedure is known as “marking” contracts. Based on any changes in the value of each contract, there will be updated debts between the buyers and sellers in each contract. This debt network can become very complex given that one trader may be a participant in multiple contracts, but nevertheless reducible to smaller set of payments. This reduced debt network is the result of the final step of the clearinghouse’s daily obligations which is known as netting.

Our solution will be built on top of the IBM Hyperledger infrastructure. Hyperledger simplifies the blockchain network into a series of participants, transactions and assets. In our implementation the participants in the network are traders and a single admin, whose responsibility is to initiate the state transitions from Start Of Day (SOD) to End Of Day (EOD). The assets in this network consist of orders and contracts. The transactions types are specific to the different types of participants. Traders will have access only to the PlaceOrder transaction, which takes user defined order information and publishes to the network. The admin can submit transactions which progress the network through it’s different states. The SOD transaction from the admin will prune orders and begin accepting orders. The EOD transaction will stop accepting orders, mark contracts based on end of day spot prices and determine net settlements.

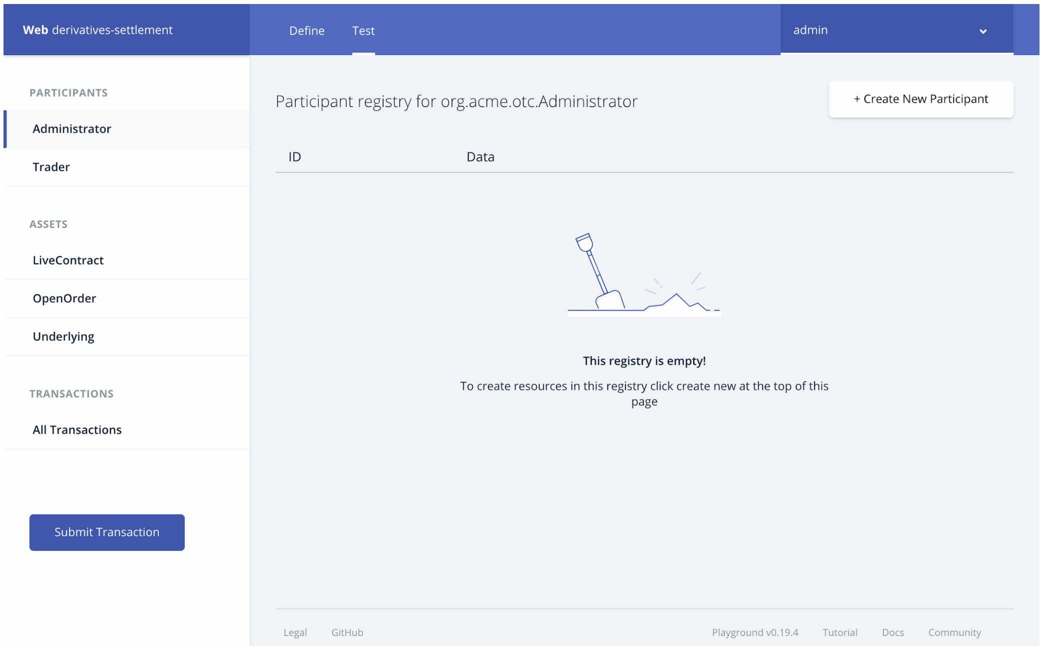
An important demonstration to show that our application works will be a successful trading day demonstrated for the entire class. In this, we will have three accounts on the chain: one for the administrator and two for traders. The users will interact on the same network by using a secure shell connection into a central server and logging into different accounts on the Hyperledger interface. The entire workflow described above will be followed, displaying each functionality.

The above workflow is the “low hanging fruit” for this project, but there are several additions that can be made to improve upon and make the MVP closer to reality. For example, monitoring margin accounts and having traders verify in some manner that the appropriate payments for that trading day’s debt have been made external to the chain will bring this system closer to what the current central counterparties are able to handle on their own infrastructure.

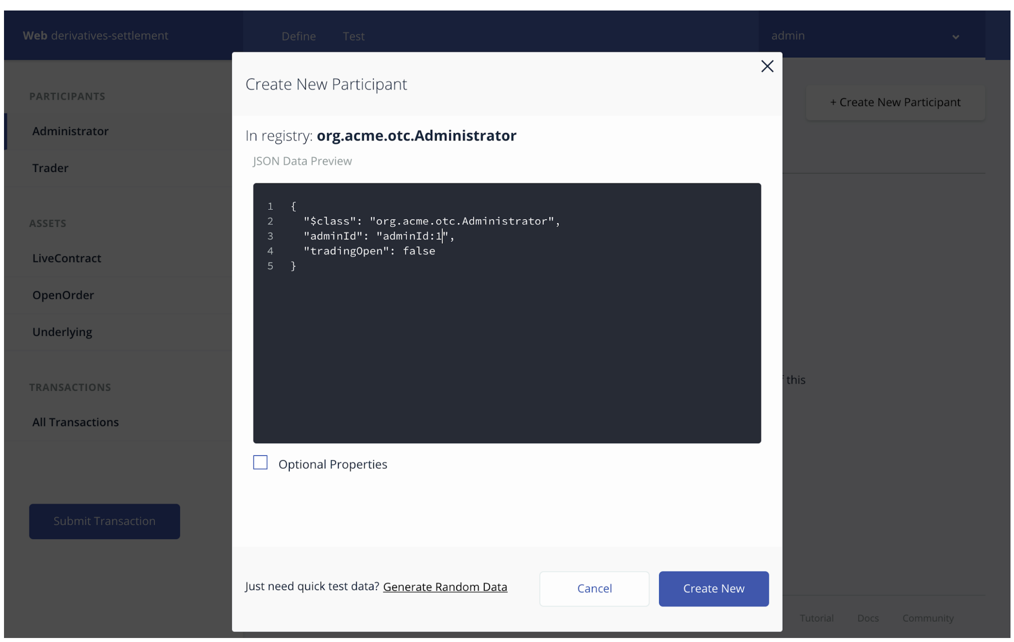
CCP’s also retain the ability to provide an insurance fund to the network, composed of small amounts of margin from every traders account that can be used to pay out creditors when debtors cannot meet margin calls. They can lockout institutions that haven’t paid from trading further with the network. Once the margin monitoring addition is created, this further modification can be added on top.

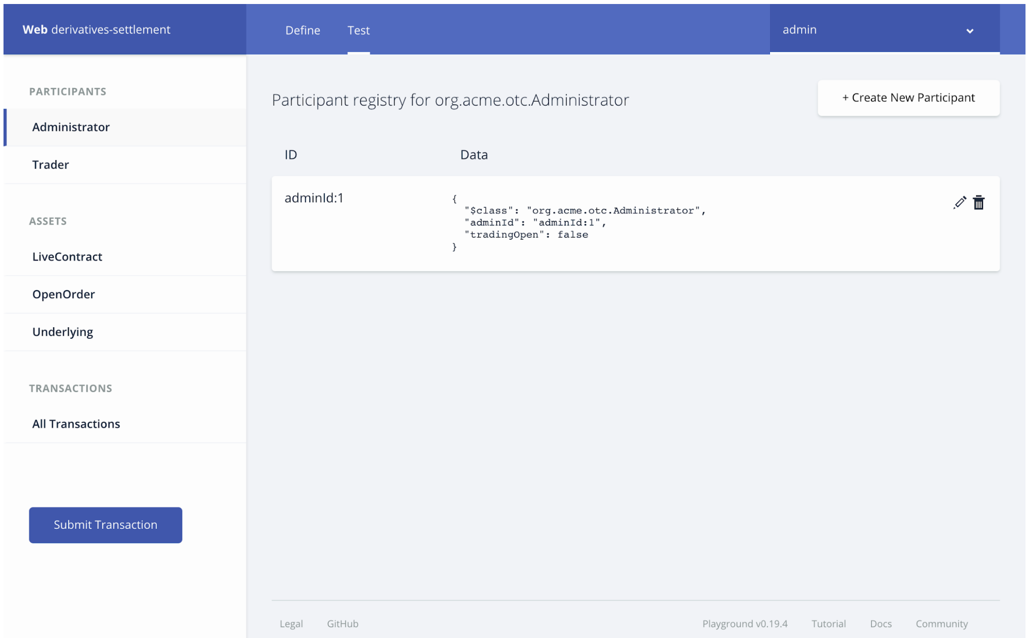
The final and biggest modification that can be made is full exchange functionality. The assumption to this point is that traders agree on contracts using some direct connection between trading floors and then the contracts are added to chain. If the blockchain application can handle all modifications to this point, there is no reason why a full decentralized exchange for derivatives contracts can be created with the same infrastructure, with the entire trading workflow from marketplace to settlement to margining. If the base functionality can be finished by April 27th, these modifications can take shape.

PROJECT DEMONSTRATION:

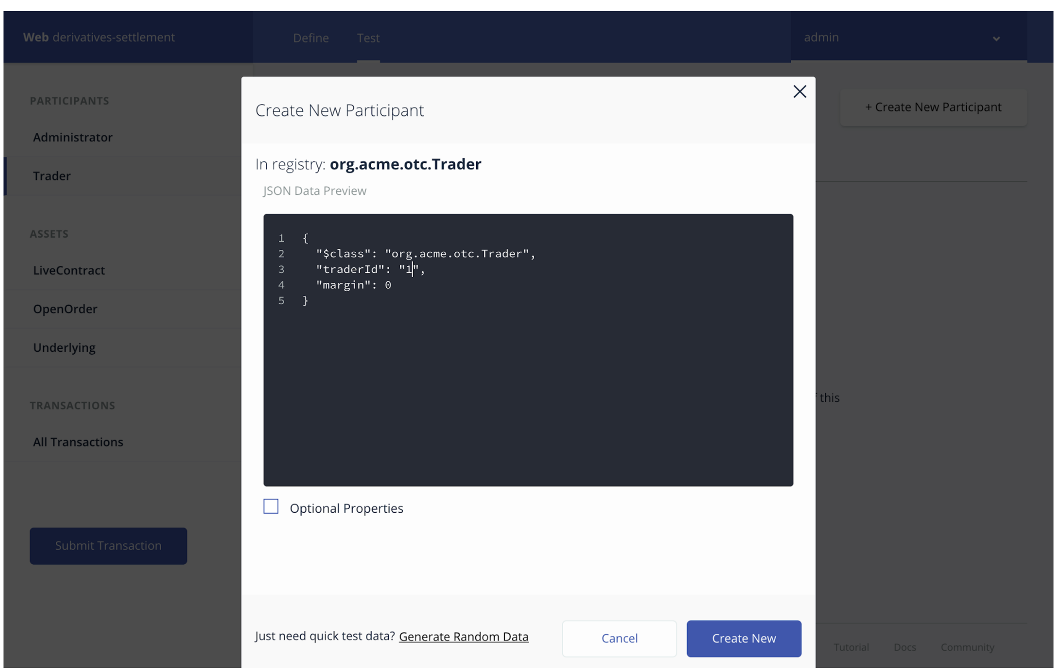


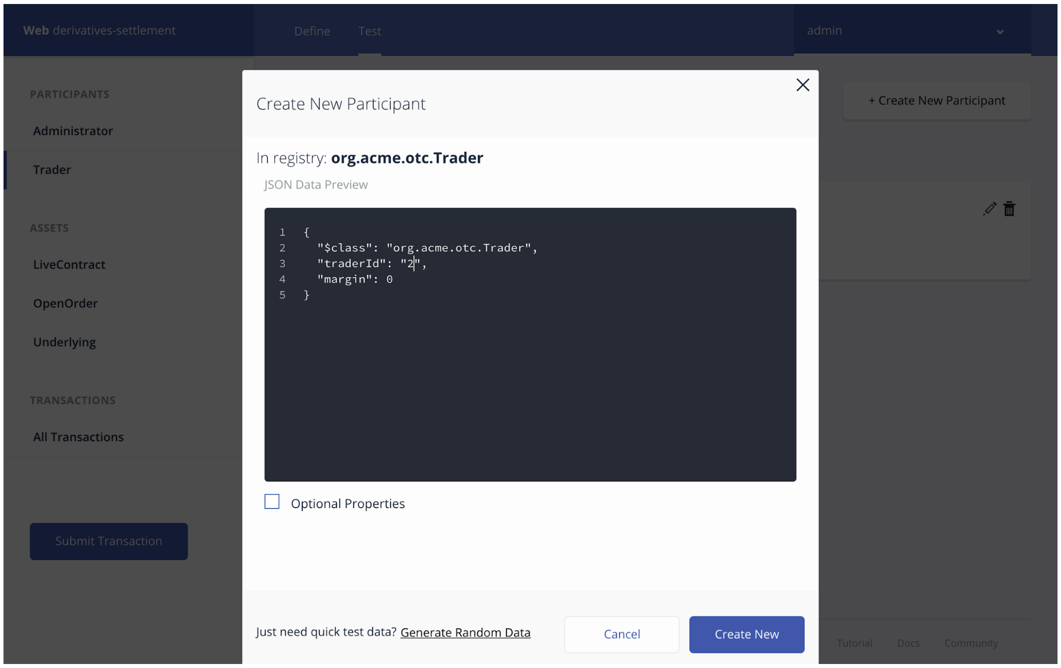
Starting with a fresh instance of the Hyperledger Composer sandbox, we begin by making an Administrator participant.



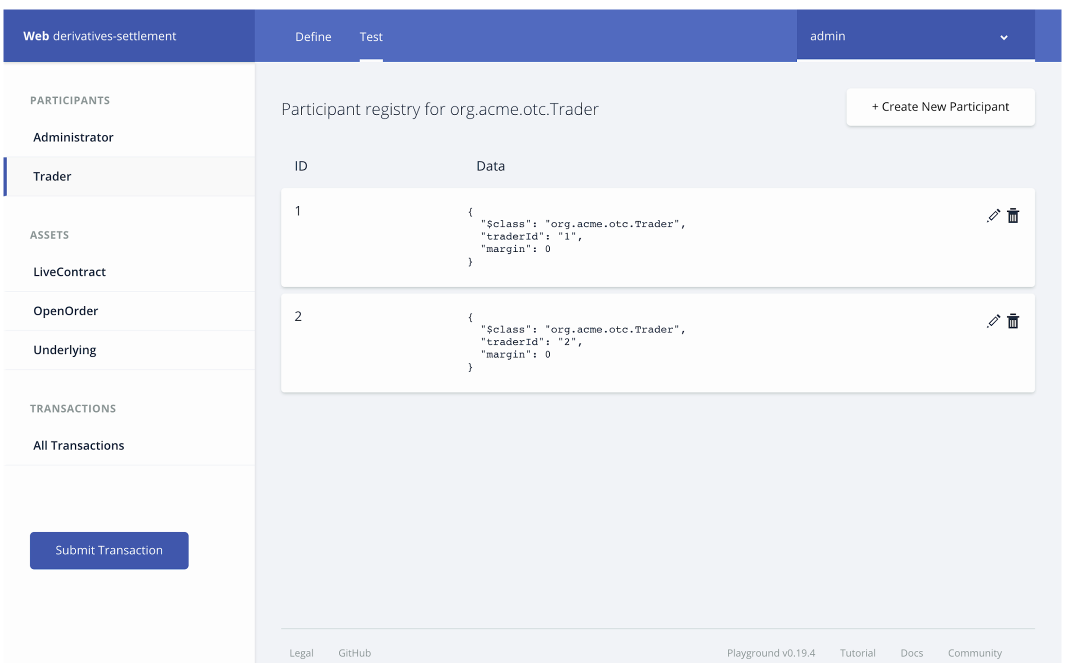


Here, we see the Administrator is identified by “adminId:1” and has a boolean value of “tradingOpen”. It is this bool that is referenced to determine if orders can be accepted by the network. We then create Trader participants, which are each identified by a number.

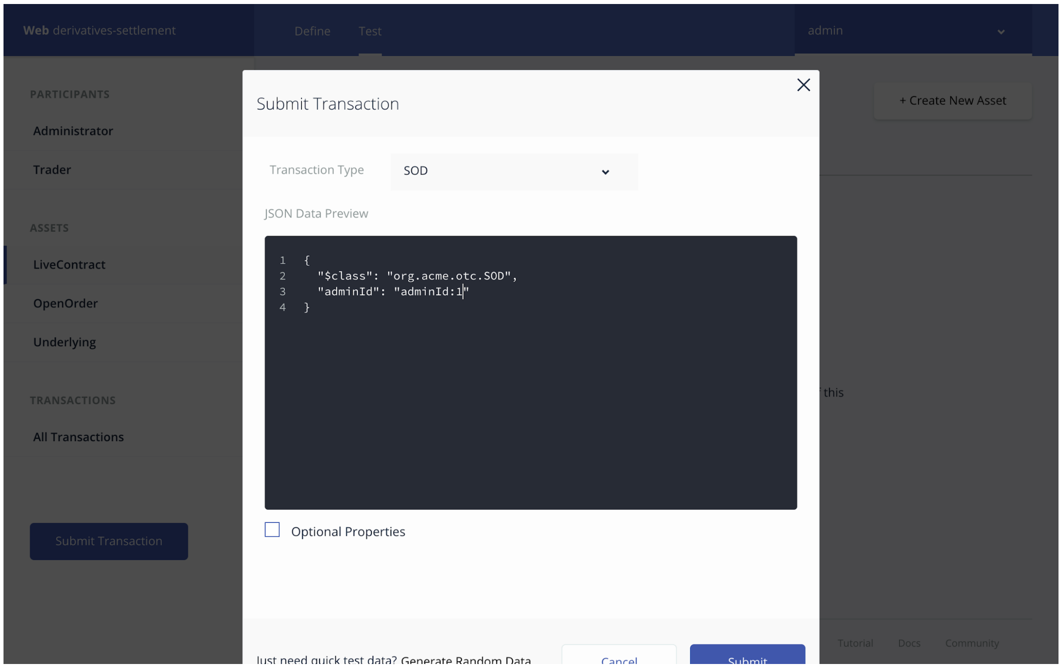


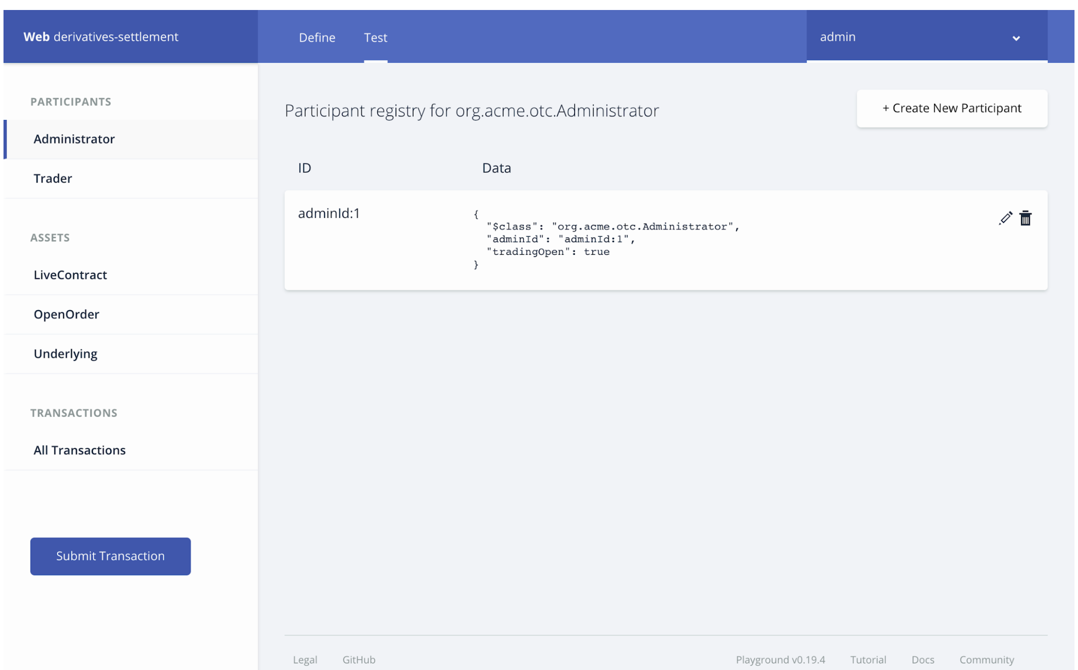


The traders also have a field for a floating point number entitled "margin". The value of this field reflects the capital the traders have in their margin accounts. Below, we see that both traders are present in the participant registry.

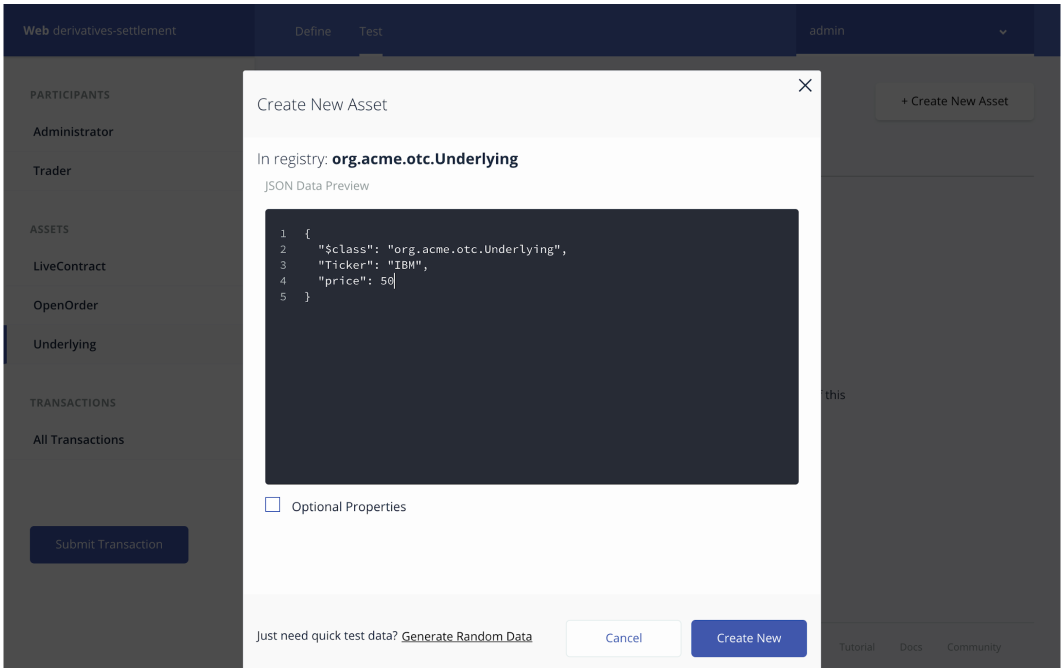


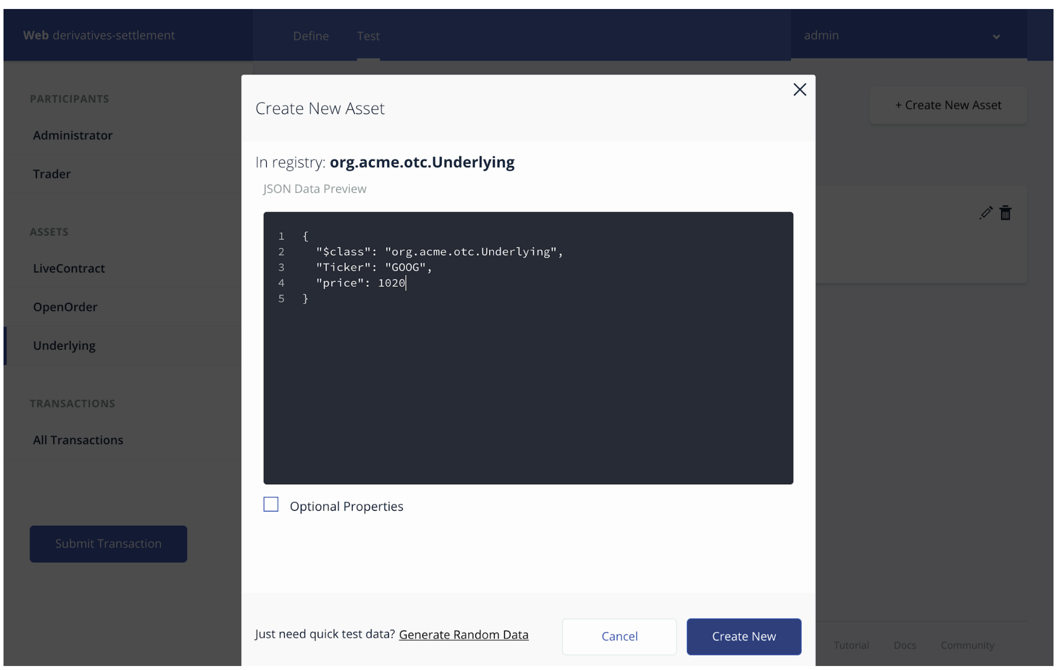
From this point, the admin can submit a SOD (Start of Day) transaction, changing the boolean value of “tradingOpen” to True.



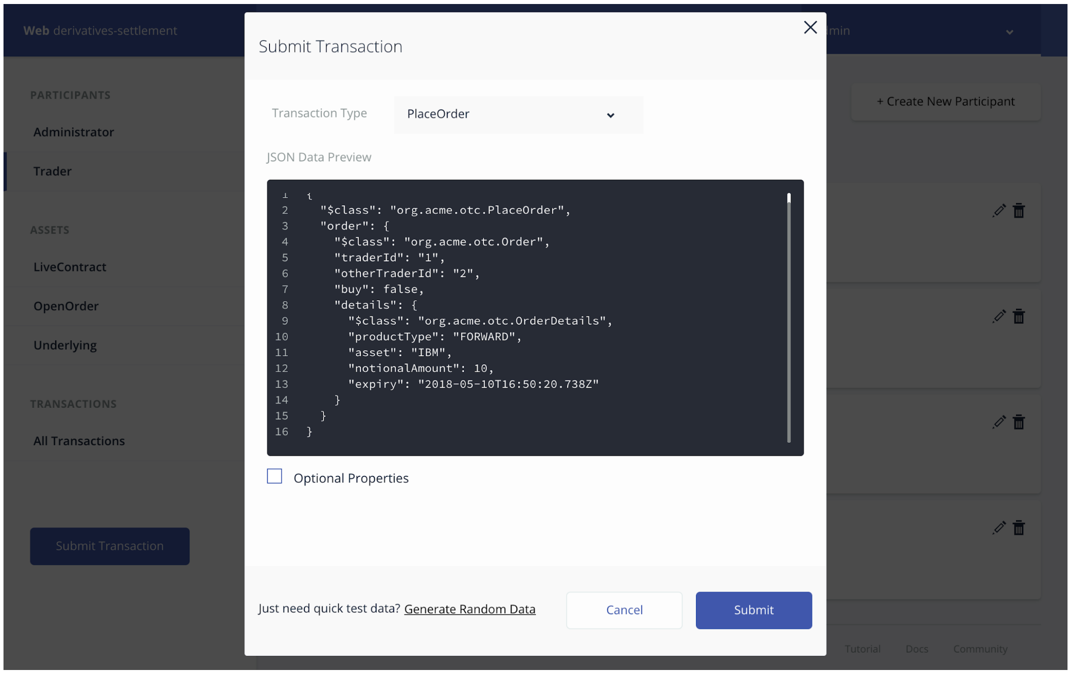


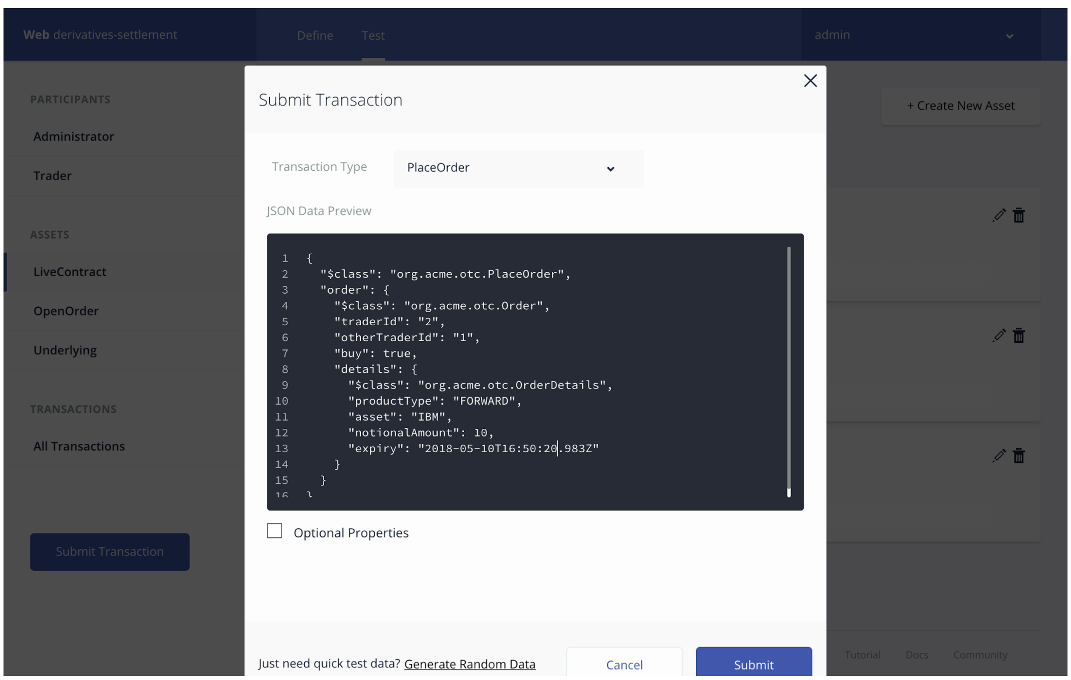
We will also need to add Underlying assets to the network such that their prices may be referenced in the evaluation of contracts. We show this below:



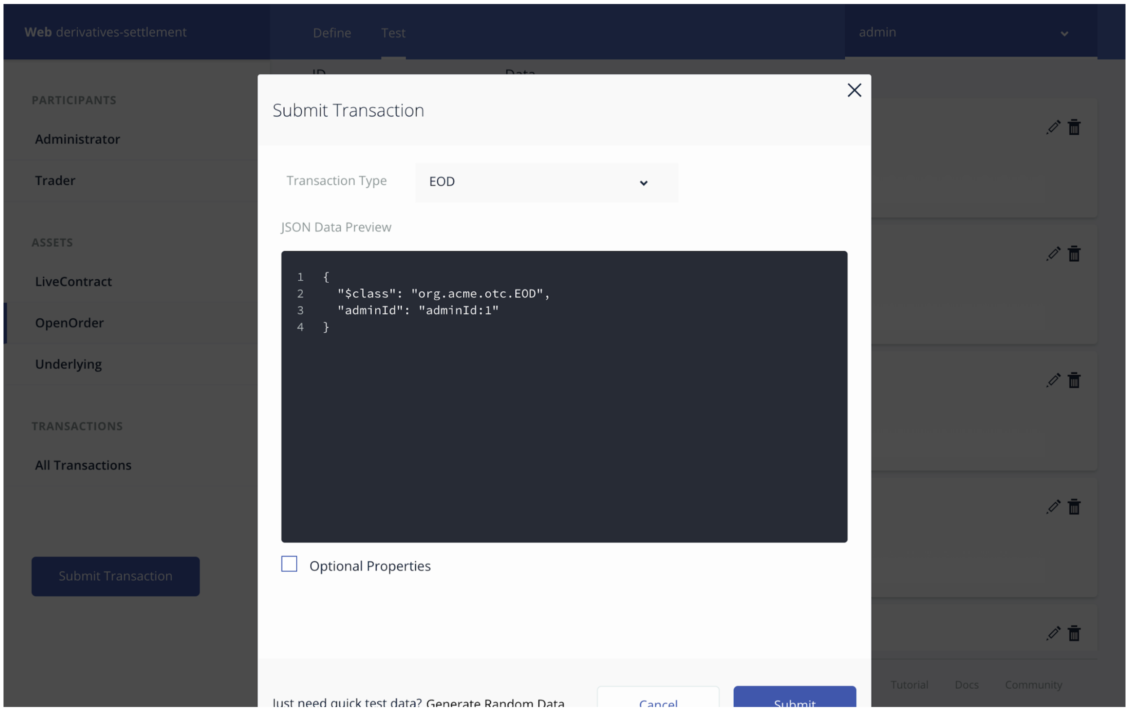


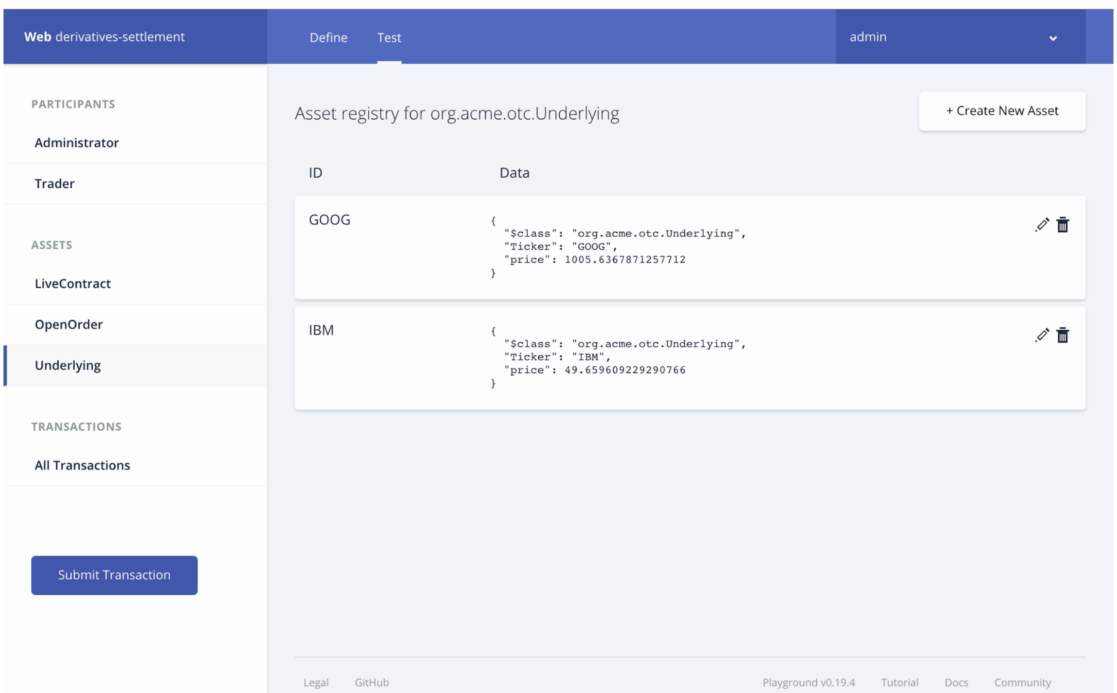
We now demonstrate the submission of orders to the network by use of the PlaceOrder transaction. This is shown below between two traders who are respectively buying and selling a forward contract on IBM for the notional amount of 13.

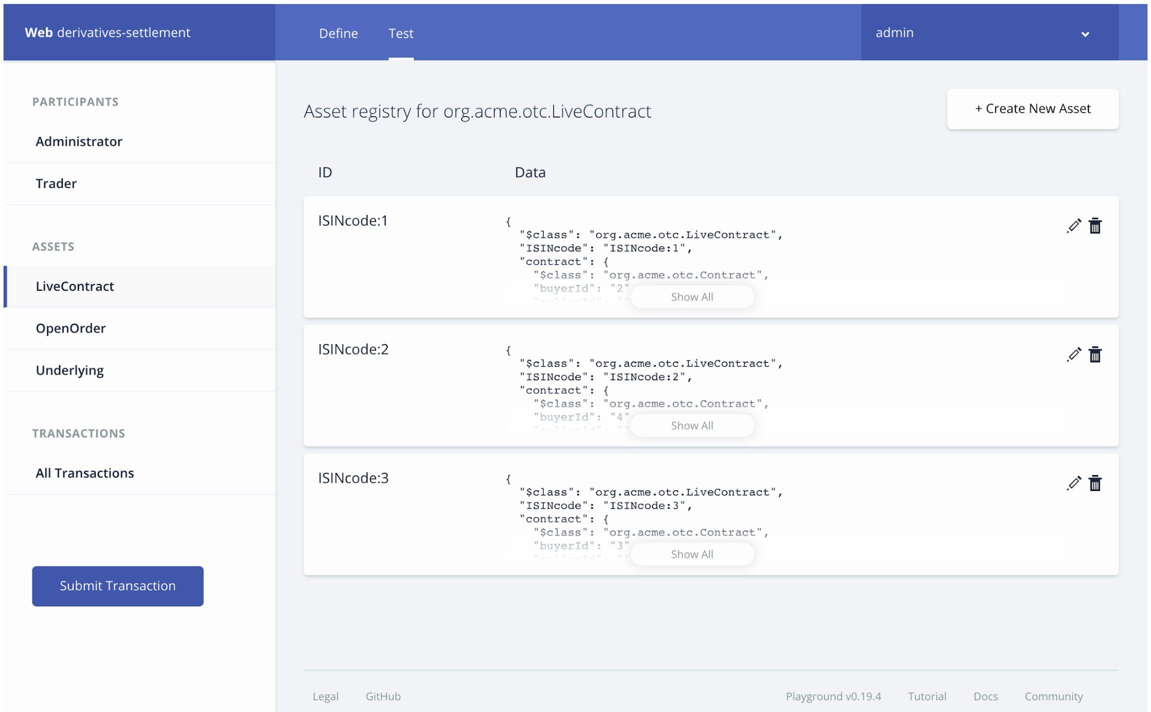




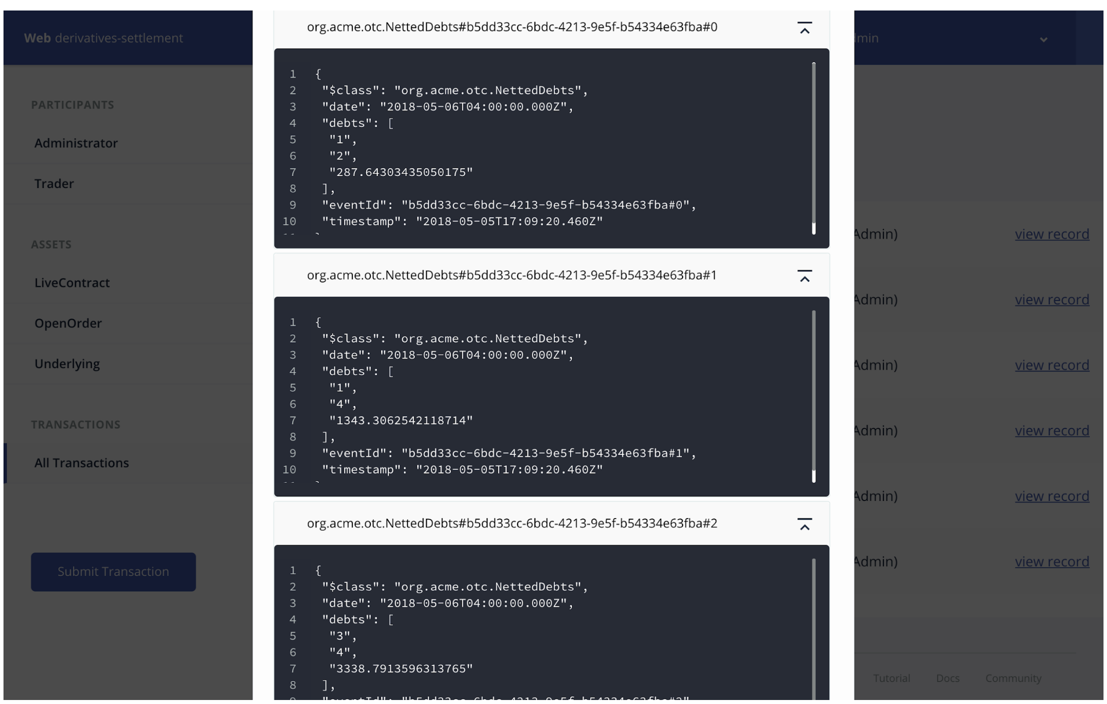
By having the admin submit an EOD (End of Day) transaction, we demonstrate the updating of asset price data, the formation of contracts from matching orders, and the re-evaluation of all contracts in the registry based on the changed asset prices.


Web derivatives-settlement
Define
Test
admin
PARTICIPANTS
Administrator
Trader
ASSETS
LiveContract
OpenOrder
Underlying
TRANSACTIONS
All Transactions
Submit Transaction
Submit Transaction
Transaction Type
EOD
JSON Data Preview
1 {
2 "$class": "org.acme.otc.EOD",
3 "adminId": "adminId:1"
4 }
Optional Properties
Tutorial
Docs
Community


Web derivatives-settlement
Define
Test
admin
PARTICIPANTS
Administrator
Trader
ASSETS
LiveContract
OpenOrder
Underlying
TRANSACTIONS
All Transactions
Submit Transaction
Asset registry for org.acme.otc.Underlying
+ Create New Asset
ID
Data
GOOG
{
"$class": "org.acme.otc.Underlying",
"Ticker": "GOOG",
"price": 1005.6367871257712
}
IBM
{
"$class": "org.acme.otc.Underlying",
"Ticker": "IBM",
"price": 49.659609229290766
}
Legal
GitHub
Playground v0.19.4
Tutorial
Docs
Community



Finally, we observe that the debt settlement and netting procedure succeeded as shown by following “nettedDebts” object.



Interestingly enough, our group found that the easiest portions of this assignment were the trading and valuation logic. On the trading side, it was incredibly easy to accept orders as long as the traders did not make errors while entering values. During the trading day, there is virtually no computation allowing for high usage and low latency when it matters.

On the contract valuation side, the equations are highly deterministic in their closed form. For example, the code to process forward contracts is just a few lines:

```
                                var exp_val=Math.exp(.03*day_diff);
                                newValue =
(u.price*exp_val)*contract.contract.details.notionalAmount;
                                debt = newValue-contract.contract.value;
```

Most of the computation in this step is done in the getting of contracts, removing invalid contracts and iterating through each one to record debt.

The most difficult parts of this project involved interfacing with the Hyperledger project, working through all of its quirks and abnormalities and writing the appropriate JavaScript, which is a new language for our whole group. Many of the irregularities of JavaScript have to do with its strange object oriented nature, particularly Prototypes, Promises and general Object casting. Another irregularity is the Await, Async functionality that allows functions to execute asynchronously, which proved troublesome to design around and control. Fortunately, Stack Overflow and the hired Hyperledger support team that answers related questions on the forum allowed us to iterate through bugs. No precompilation and minimal syntax error catching in the Hyperledger environment made for painful coding sessions.

How does our application compare with the state of the art CCP system?

Cost:

Obviously, a system run on servers run by the broker dealer, investment banking network for their own transaction in a private, permissioned system has zero transactional fees other than server costs and upkeep development man hours. In comparison, Goldman Sachs estimate would apply here for the CCP system.

Transparency:

For people in the permissioned, private derivatives settlement system can more easily understand the transactions they are involved in and know what events transpire that would require socialized action. For example, investment banks can find the anonymous trader that engaged in a series of leveraged trades that kicked off a margin call and insurance fund liquidation. While transparency isn't something that most financial companies ask for, regulators certainly would like a view inside the OTC derivatives system.

Custom Contracts:

A blockchain derivatives settlement system can support non standard contracts, or so called “exotic” contracts, through a system of soft forking. For example, a broker-dealer and a customer could agree on a pricing equation and terms as necessary, add code in the JavaScript files to support the marking and push the transactions on. Other nodes would not be able to verify, but the debt would be generated on the broker dealers system and be netted along with all other transactions. It would enjoy all the advantages of the blockchain settlement system, including no transactions fees and easy settlement, alongside all other transactions.

Legal Reversibility:

There are acceptable and relatively frequent instances when derivatives trades and money must be reversed. Judge orders and legal suits would require illegal trades involving insider trading and collusion to be removed and gains from these trades erased. How would a blockchain settlement system handle this, keeping in mind that blockchains be append only and incorruptible? In its current implementation, our system would only support this through a judge mandated hard fork, where all institutions would be required to switch to a version of the chain where this data doesn't exist, and the money in the margin account gained from the trades removed. Central counterparties can do this by hand in the state of art right now.

There are a few things that would make our application more robust and useable:

Legal Participants:

Looping in regulators and judges directly into the system can do some good for the system. While financial corporations would prefer to keep their system indirectly connected to regulators, keeping them directly aware of OTC trades would allow for financial compliance of trades to regulation like Dodd-Frank and MiFiD to be immediately checked and reversed with no complex litigation and fines required for either parties. Trades involved in civil suits could have margin accounts frozen. Having legal participants in the Hyperledger infrastructure would be ideal, compared to forcing hard forks on every court order. An Hyperledger version of OpenLaw on the Ethereum network would be the correct implementation.

Automate Start of Trading Day and End of Trading Day:

The real version of this blockchain would automatically trigger start of day and end of day through the system clock on each node when accepting orders.

Margin Account blocking:

When traders refuse to pay their margin or are unable to, a settlement system should prevent the adding of orders involving that trader until the margin is met and debts paid.

Insurance Fund:

In the previous situation, the ideal network would liquidate a precreated and continually taxed insurance fund made from small amounts of all traders margin amounts in order to pay the debts owed by the indebted trader. The network would continually assess the market risks of all the positions and adjust the rates taxed from all the accounts in case a trader is suddenly wiped out and cannot cover the margin.

Order Book:

The ideal system would be able to create order books on any customized product. Putting an OTC exchange on the blockchain would be possible, considering the relatively slow trading of OTC derivatives contract, and making them would make the market significantly more transparent and price discovery more realistic.

Final Contributions:

If our current system was put to market today, it would be implemented as such:

- All major broker-dealers and investment banks would run nodes on the system, hosting Hyperledger Fabric code
- All customers and traders would interface with their brokers REST API, placing orders and maintaining payments with their broker

- As new contracts are created and engineered between brokers and customers, the Hyperledger Fabric code is soft forked such that specialist pricing equations are included
- Teams of developers at each major bank and trading institution maintain the Hyperledger code, forming a community in charge of maintenance and creating new functionality

References:

- Surujnath, Ryan. “ OFF THE CHAIN! A GUIDE TO BLOCKCHAIN DERIVATIVES MARKETS AND THE IMPLICATIONS ON SYSTEMIC RISK .” *Fordham University*, Fordham University School of Law, 1 June 2017, news.law.fordham.edu/jcfl/wp-content/uploads/sites/5/2017/06/Surujnath-Note_pdf_publishing.pdf
- Verhoeff, Tom. *Settling Multiple Debts Efficiently: Problems*. *Settling Multiple Debts Efficiently: Problems*, Eindhoven University of Technology, 2004.