Decentralized Lending of Cryptocurrency

Douglas A. Bebber
dab@i2pmail.org

1st draft - February 12, 2017

Abstract
This paper describes a decentralized pattern for lending, targeted at enabling the margin trading of cryptocurrencies on decentralized trading exchanges. The pattern provides for offers (borrow offer and loan offer) that when matched create smart lending contracts utilizing a Collateralized Debt Position (CDP) where collateral (initially bitcoin) is secured through the use of a 2-of-3 multi-signature address. Automated software arbiter functionality is provided via a decentralized consensus algorithm. The decentralized lending protocol utilizes a Merkle DAG[1] data structure to provide a secure, tamper-proof record of the lending workflow that the decentralized consensus algorithm processes.

1 Introduction

Presently, popular exchanges that enable the trading of cryptocurrencies are architected in a centralized manner. Those exchanges that enable margin trading typically provide additional liquidity over those exchanges that do not provide margin trading. This makes sense as it allows traders opportunities to profit on the long side (as prices rise) as well as on the short side (when prices decline).

Centralized exchanges however expose traders to counterparty risk. Traders are forced to trust the centralized exchange. Cryptocurrency traders must transfer their cryptocurrency to the exchange where the exchange holds the funds in trader owned accounts. The exchange thereby becomes a necessary trusted counterparty. Traders assume increased risk of losing their funds as a result of fraud, theft or regulatory capture. Some recent examples include mtGox, bitfinex, Huobi, OKCoin, and China.

Decentralized exchanges, such as Bitsquare (https://bitsquare.io/) provide a new and exciting model for what the future holds[2]. The decentralized exchange model eliminates these counterparty risks by enabling peer-to-peer (P2P) trading and as a result, should

[1] Merkle DAG description in IPLD
[2] Bitsquare - Decentralized Bitcoin Exchange

disrupt and obsolete the centralized exchange model moving forward. Barriers to the success of the decentralized P2P exchange model, as it stands today, are many. Two primary barriers are:

- Decentralized P2P trading exchanges lack automation and require manual human actions and responses that create extremely long cycle times when compared with centralized exchanges. A trade on a centralized exchange can be completed in seconds whereas a similar trade on a decentralized P2P exchange could take hours or days.

- Margin trading does not exist on decentralized P2P exchanges, so short selling is not supported and this limits liquidity and participation.

This paper will provide a model to address the lack of margin trading on decentralized P2P exchanges. Our model will eliminate the necessity of having a trusted counterparty, reduce cycle times through the use of automation and improve liquidity by enabling short selling via a decentralized pattern.

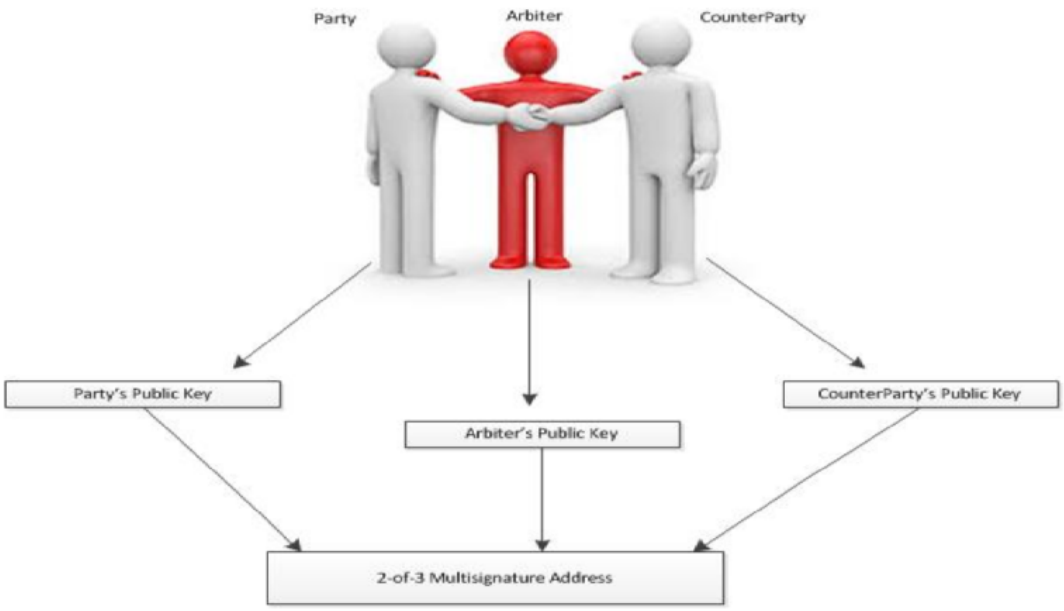
Additionally, it would provide income opportunities for traders who desire to lend idle funds and earn interest in the process. This interest income opportunity would attract additional participants to the decentralized exchange and provide a new economic sector within a decentralized P2P trading exchange model.

Poloniex is an example of a centralized exchange that provides margin trading and allows exchange users to earn interest by lending cryptocurrency to margin traders. Providing similar capability through a decentralized (P2P) exchange is the focus of this paper.

The key value-add of our model is the decentralized autonomous consensus facility which consists of a workflow model, a secure and tamper-proof Merkle DAG data structure, and a rule-based decentralized consensus algorithm. This facility has been designed for the automation of rule-based escrow workflows. It can be reused to transform existing centralized services to autonomous decentralized service offerings.

2 High-Level Description of the Decentralized Lending Protocol

The decentralized lending protocol utilizes *smart contracts*. There are three parties to a *Lending Smart Contract:*



Borrower

The borrower is typically the trader that trades on margin and desires to borrow an amount of cryptocurrency to sell short.

Lender

The lender has cryptocurrency available to lend to margin traders. The lender desires to earn interest income through the lending of cryptocurrency at a specified rate of interest for a specified duration.

Arbiter

The arbiter is a trusted third-party employed to make certain that the parties to the contract follow the rules and properly execute the terms and conditions of the contract.

In our model, the borrower, is a trader who borrows cryptocurrency from a lender with the expectation of selling the borrowed cryptocurrency at a specified price, wait for the price of that cryptocurrency to decline, so that the trader can buy it back at a lower price. This practice is commonly referred to as *short-selling*. The goal is that the margin trader makes a profit doing this. The difference between (sell price - buy price) is large

enough to cover the repayment of the loan principal + interest + fees while leaving a profit for the trader.

The lender is an individual that has an amount of cryptocurrency in his/her possession and would like to earn income by lending it to margin traders.

In our model, the arbiter is an implementation of a rule-based decentralized consensus algorithm that governs the proper execution of the terms specified in lending smart contracts. In our model, the borrower and lender do not know each other. They do not even know the identity of their counterparty. Therefore, they can not, and should not trust one another. They need to trust the arbiter which is an open-source software implementation of a rule-based decentralized consensus algorithm. Since the arbiter functionality is a decentralized consensus algorithm, there are actually many arbiters.

Our model is a P2P system. Where each participant in the decentralized lending network is running our open-source software package on their own hardware. This is referred to as a system node. Each system node has the capability of performing the arbiter function if the user opts-in and turns on the arbiter functionality switch. Users who opt-in and allow their node to participate as an arbiter are able to earn bitcoin as a reward for performing arbiter work. *Each P2P node has an arbiter bitcoin wallet for accumulating rewards (BTC).*

Many participating arbiter nodes operating concurrently in the system executing the rule-based decentralized consensus algorithm, provides the trust mechanism for ensuring that smart lending contract terms and conditions are executed properly.

In our model, we have provided an API interface to further levels of automation which would allow borrowers and lenders the ability to programmatically specify rules that can be followed in the lending workflow. This will allow borrowers and lenders to employ “bots” to automate their lending practices.

Since the rule-based decentralized consensus algorithm is executed in code on a participating system node, the arbiter function is completely automated. The evidence for the state and execution of the lending workflow is persisted in a secure and tamper-proof Merkle DAG data structure resident on a distributed P2P file system. The Merkle DAG provides a transparent, tamper-proof chain of evidence of the decentralized lending workflow.

The entire software package, including the rule-based decentralized consensus algorithm is open-source and provides military-grade cryptographic algorithms for security. So users of the system can gain a high-level of trust in the integrity of the automated arbiter facility, and the overall system itself. Our goal is to have users place their trust in a secure and transparent mathematical model rather than a human being playing the role of a trusted arbiter. Providing a high-level of automation in a

decentralized lending system will provide faster cycle times resembling those of centralized systems.

3 High-Level Description of the Decentralized Lending Workflow

The workflow starts with the submission of offers. Borrowers write offers to borrow cryptocurrency. Lenders write offers to lend cryptocurrency. *There is a fee (bitcoin) required to write an offer. This fee is refunded to the offerer once the lending smart contract is successfully executed. This fee is intended to incent proper behavior amongst participants*. The arbiters constantly scan submitted offers with the objective of finding a *match*. When offer matching criteria is met, the arbiter takes the two matching offers and creates a lending smart contract and records it in the Merkle DAG.

When an arbiter creates a lending smart contract, the borrower and lender are notified via a workflow event message. That message is also sent on the *arbiter message channel* (a message channel to communicate with all arbiters throughout the system), so that a decentralized consensus can be obtained for the newly created smart contract. Other arbiters will examine the offers, validate they in fact match, then digitally sign the smart contract to formulate a decentralized consensus. The evidence is written to the Merkle DAG to record the chain of evidence.

The message sent to the parties regarding the new lending smart contract informs all parties that a new contract has been created and that the borrower needs to deposit collateral (in bitcoin) to secure the loan. The arbiter, makes an external call to obtain data in order to determine the amount of collateral the borrower needs to deposit. Additional arbiters confirm the necessary amount of collateral so that a decentralized consensus regarding the amount of collateral needed to secure the loan is attained. The arbiter decentralized consensus regarding the amount of collateral needed to secure the loan is written as evidence in the Merkle DAG to verify the documented smart contract.

Upon receiving the notification to deposit collateral, the borrower (and all other parties to the contract) are provided the 2-of-3 multisignature Bitcoin address that the collateral needs to be deposited into. The 2-of-3 multisignature Bitcoin address is inspected to ensure that the borrower and lender public key specifications recorded in the original offers are in fact correct. Additional arbiters confirm this and digitally sign the confirmation in the Merkle DAG to record the decentralized consensus. An arbiter public key is also necessary for the 2-of-3 Bitcoin multisignature address. The acquisition of an arbiter public key is obtained via the *Arbiter Public Key Request* facility (described in the Decentralized Lending Collateral Facility and Multi-Signature Security section of this document). Other arbiters validate the arbiter public key and digitally sign the smart contract in the Merkle DAG to record this fact via decentralized consensus.

The borrower then has a specified time frame (ignoring network latency) in which the collateral must be deposited to the 2-of-3 Bitcoin multisignature address. *If the borrower*

does not deposit the required collateral to secure the loan within the specified time interval, the contract is breached and terminated. The lender's offer remains in effect and can be used for another lending smart contract. The borrower's offer is removed from the system, and the borrower's offer fee is donated to the arbiter reward pool.

Once the borrower has deposited the necessary collateral, a *Collateral Deposit* event is generated and a message is sent to all parties to the contract (borrower, lender, and arbiter channel). The arbiters validate that the confirmed Bitcoin transaction for the collateral deposit occurred successfully (by inspecting the blockchain explorer) and digitally sign the evidence of this fact in the Merkle DAG. The arbiter then sends a *Transfer Loan Principal* message to all parties of the contract, to signal that the borrower has deposited the necessary collateral and that the lender now needs to transfer the loan principal amount to the borrower.

The lender has a specified time frame (ignoring network latency) to transfer the loan principal to the specified borrower deposit address. *If the lender does not transfer the loan principal within the time period, the contract is breached and terminated. The collateral is returned to the borrower, the borrower's offer remains in effect, however the lender's offer is removed from the system, and the lender's offer fee is donated to the arbiter reward pool.*

Once the lender has transferred the loan principal, the arbiter validates the transfer took place by inspecting the appropriate blockchain explorer to verify this fact. Once verification is confirmed this is written and digitally signed in the Merkle DAG. Additional arbiters perform the work necessary to provide a decentralized consensus in the Merkle DAG and notification that this step has been successfully completed is sent to all parties of the lending smart contract. The loan clock starts and interest begins accruing. *The borrower can stop the loan clock at any time without penalty.*

When the loan clock stops (either by counting down the loan duration specified in the smart contract, or the borrower decides to terminate the loan early), an event notification is sent to all parties to the contract. Arbiters confirm the event and write the decentralized consensus to the Merkle DAG. Arbiters also write the interest amount due into the lending smart contract in the Merkle DAG and do so providing a decentralized consensus.

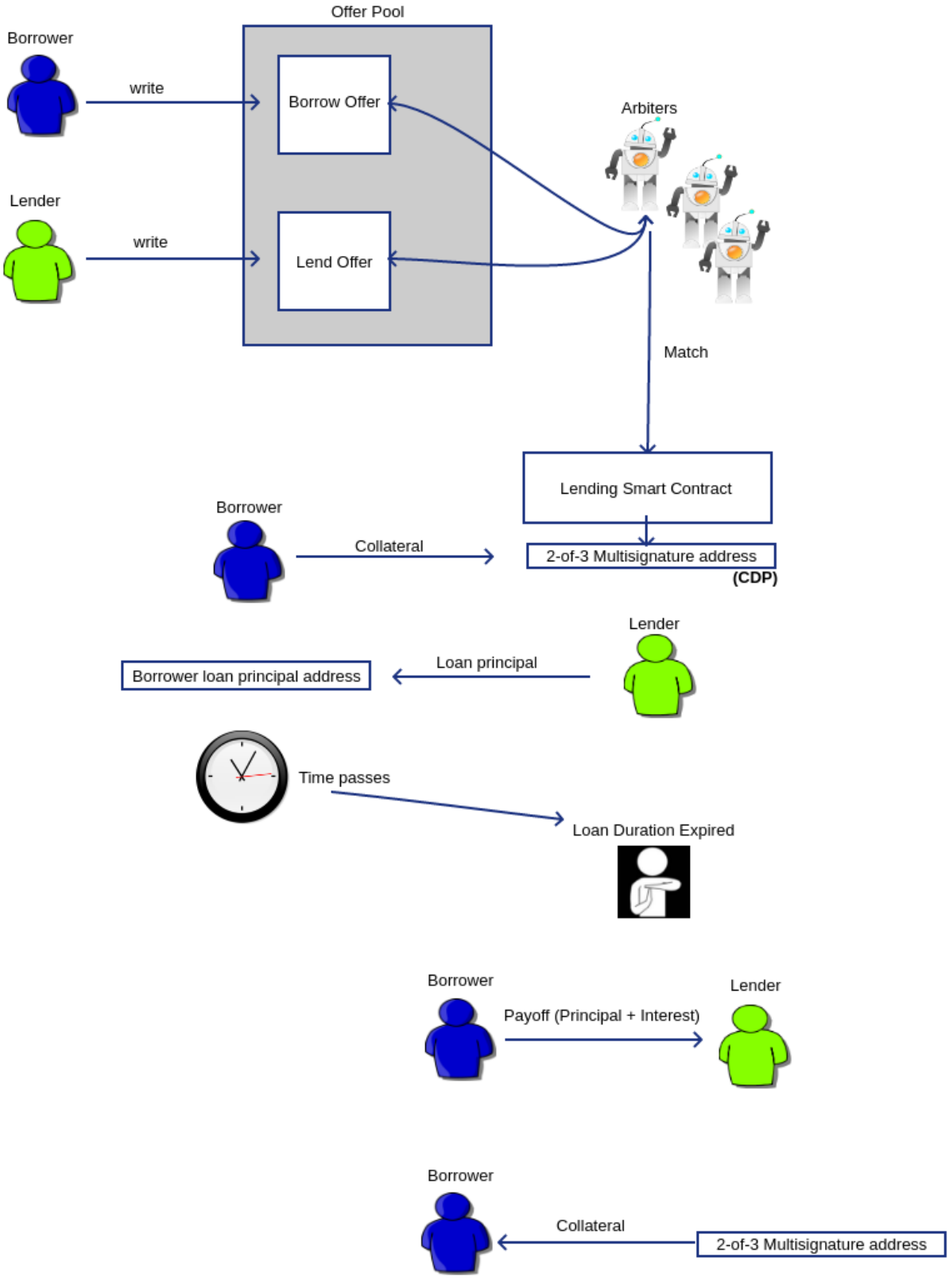
The borrower has a specified time frame (ignoring network latency) to pay off the loan (principal + interest) to the lender. *If the borrower does not transfer the payoff amount for the loan within the specified time interval, the contract is breached and terminated. The lender's offer is removed from the system and the offer fee returned to the lender. The borrower's offer is removed from the system, the borrower's collateral is forfeited to the lender, and the borrower's offer fee is donated to the arbiter reward pool.*

Once the borrower has transferred the proper payoff amount to the lender, The arbiter verifies the transfer took place by inspecting the appropriate blockchain explorer. Once

verified, the arbiter digitally signs the Merkle DAG to record the evidence of this fact. This verification is provided via decentralized consensus.

The collateral is returned to the borrower, the offer fees are returned, and the lending smart contract is marked complete. All of the arbiters who were involved in the decentralized consensus work for this contract are rewarded with the *Decentralized Consensus Fee* (a fraction of the loan interest) attributed to the lending smart contract.

The Decentralized Lending workflow just described is presented in the figure below.


Offer Pool
Borrower
write
Borrow Offer
Arbiters
Lender
write
Lend Offer
Match
Lending Smart Contract
Borrower
Collateral
2-of-3 Multisignature address
(CDP)
Lender
Borrower loan principal address
Loan principal
Time passes
Loan Duration Expired
Borrower
Payoff (Principal + Interest)
Lender
Borrower
Collateral
2-of-3 Multisignature address

4 Decentralized Lending Collateral Facility and Multi-Signature Security

Because the borrower and lender do not know one another, the lender needs some assurances that he/she will be repaid. Without such assurances, lenders would not assume the risks of making loans to complete strangers. For this reason, all borrowers are required to collateralize their loans.

The amount of collateral necessary to secure a loan is determined by the terms of the lending smart contract. The collateral must be sufficient to cover the value of the loan principal plus the interest accrued through the duration of the loan. Additionally, a margin amount of collateral is required to compensate for any volatility in the value of the involved currencies (collateral currency and loan principal currency).

We are making provisions for dynamic margin adjustments within the life cycle of active loans. However, this capability is absent in the prototype and we expect it will be absent in the initial release due to network latency factors and the high levels of volatility currently associated with cryptocurrency prices.

Our decentralized consensus algorithm is designed to terminate all active lending smart contracts once extreme volatility limits are breached. *For example, if after a lending smart contracts loan clock has started, the value of the loan principal increases significantly, such that the value of the loan principal borrowed exceeds the market value of the secured collateral, the loan is immediately terminated and the proceeds are distributed equitably.*

Because of the current volatility potential of cryptocurrencies, the required margin level is higher than what one would find in other lending markets. While a specific requirement is not set in stone we are thinking that a static margin requirement would not exceed 150% of the loan principal. As improvements are made and dynamic margin adjustments are available to the system, the static margin requirement percentage would drop accordingly.

We expect loan durations to be short (on Poloniex the default loan duration is currently two days) so locking up higher amounts of capital for collateral purposes for short durations would seem acceptable.

Collateral deposits are secured through cryptocurrency multi-signature technology. Currently this is only available via the Bitcoin protocol. So currently only bitcoin is accepted for collateral purposes. We expect Monero (XMR) to support n-of-n multisignature transaction capability soon and we would make provisions to accommodate additional collateral cryptocurrencies as soon as acceptable multisignature capability is available for such cryptocurrencies.

Our model currently supports 2-of-3 multisignature transactions for collateral transactions involving Bitcoin protocol. There are 3 key sets involved in securing collateral:

1. Borrower
2. Lender
3. Arbiter

In a given lending smart contract there is only one borrower and only one lender. However, to provide decentralized consensus of arbitration there are many arbiters and providing a single arbiter key to secure collateral would effectively corrupt the decentralized consensus algorithm.

To solve this key issue we have derived the *Arbiter Key Request* facility. This facility (which will be improved upon moving forward) essentially provides a ring multisignature like facility for a group of arbiters which are able to manage the arbiter key set for the Bitcoin 2-of-3 multisignature collateral transactions. The details are quite technical, involve complicated mathematics, and will be published in a separate document.

Essentially what happens is, once an arbiter matches the offers and creates the lending smart contract, the arbiter will need to create the 2-of-3 multisignature address for the collateral deposit. The arbiter needs 3 public keys to create the 2-of-3 multisignature address. The borrower provided his public key in the borrow offer, the lender provided his public key in the lend offer. What is missing at this point is the arbiter public key.

So the arbiter that is creating the 2-of-3 multisignature address, creates a new key set (private key and public key) for this purpose. The public key is then used to create the 2-of-3 multisignature address. The corresponding private key is encrypted in such a way that a group of arbiters is later able to decrypt the private key to manage collateral transactions.

A separate research brief describing this facility will accompany the source code for the initial open source project. This facility is technically the weakest security link in the model so we intend to have many experts focus on strengthening the approach or certifying that it is indeed strong enough for the intended purpose.

5 Decentralized Consensus Algorithm and the Automation of Arbitration

The decentralized consensus algorithm consists of the lending workflow (described in summary form above) and the rules that govern that workflow (described in summary form above). The state of the workflow for every lending smart contract is persisted in the Merkle DAG which is a secure and transparent, tamper-proof record of lending workflow events and state.*The Merkle DAG is maintained on a P2P distributed file system (our prototype uses the InterPlanetary File System - IPFS.)*

A P2P network of arbiters are coded to process the lending workflow according to the governing rules. Every arbiter in the system is identical in respect to the lending workflow and governing rules. Coordination of decentralized consensus between arbiters is performed via the contents of the Merkle DAG.

For example, if a lending offer is already linked to an active lending smart contract in the Merkle DAG, an arbiter knows that it should not attempt to match that offer with any other offer. The rules that govern offer matching specify that a match can be made if the offer is not tied to an active lending smart contract.

Additional rules describe criteria for matching offers. For example if a borrower desires to borrow 100 ETH for two days at 0.010 % daily interest rate, that offer can be matched with a loan offer having the following terms:

- 200 ETH
- 0.005% daily interest rate
- 5-day loan duration

The resulting (matched) lending smart contract would look similar to:

- 100 ETH
- 0.005% daily interest rate
- 2-days loan duration

However the borrow offer could not be matched up with a loan offer of:

- 100 ETH
- 0.099% daily interest rate
- 60-days loan duration

(The offer matching rules will be completely defined in the documentation along with all the governing rules of the lending workflow).

If a lending smart contract has not yet attained decentralized consensus on a collateral deposit, and a given arbiter has not already signed off on that collateral deposit, that arbiter would then sign the collateral deposit in order to attain decentralized consensus.

Similar rules apply to any state transition within the lending workflow that calls for decentralized consensus. Arbiters compete to do such work to obtain the financial reward.

6 Prototyping and Proof-of-Concept

Our prototype and Proof of Concept source code and documentation will be made a part of the open source project to be hosted on the Internet (most likely github).

7 Foundation of Open Source Project

The source code for the prototype, Proof-of-Concept, and the follow on alpha, beta, and subsequent release code will be made open-source.

We are currently deciding which of the numerous open-source licenses would be most appropriate.

We are also considering obtaining defensive software patents for the decentralized autonomous consensus facility.

We look forward to establishing an open source community to evolve this project and move it forward.