# Putting Distributed Ledgers Together (Extended Abstract)

**Antonio Fernández Anta**
IMDEA Networks Institute, Madrid, Spain
antonio.fernandez@imdea.org

**Chryssis Georgiou**
Dept. of Computer Science, University of Cyprus, Nicosia, Cyprus
chryssis@cs.ucy.ac.cy

**Nicolas Nicolaou**
KIOS Research and Innovation CoE, University of Cyprus & Algolysis Ltd, Cyprus
nicolasn@ucy.ac.cy

## Abstract

The various applications using Distributed Ledger Technologies (DLT) or blockchains, have led to the introduction of a new "marketplace" where multiple types of digital assets may be exchanged. As each blockchain is designed to support specific types of assets and transactions, and no blockchain will prevail, the need to perform *interblockchain* transactions is already pressing.

In this work we examine the fundamental problem of interoperable and interconnected blockchains. In particular, we begin by introducing the *Multi-Distributed Ledger Objects* (MDLO), which is the result of aggregating multiple *Distributed Ledger Objects* – DLO (a DLO is a formalization of the blockchain) and that supports append and get operations of records (e.g., transactions) in them from multiple clients concurrently. Next we define the *AtomicAppends* problem, which emerges when the exchange of digital assets between multiple clients may involve appending records in more than one DLO. Specifically, AtomicAppend requires that either *all* records will be appended on the involved DLOs or *none*. We examine the solvability of this problem assuming *rational and risk-averse* clients that may *fail by crashing*, and under different client *utility* and *append* models, *timing models*, and client *failure scenarios*. We show that for some cases the existence of an intermediary is *necessary* for the problem solution. We propose the implementation of such intermediary over a specialized blockchain, we term *Smart DLO* (SDLO), and we show how this can be used to solve the AtomicAppends problem even in an asynchronous, client competitive environment, where all the clients may crash.

**2012 ACM Subject Classification** Theory of computation → Design and analysis of algorithms → Distributed algorithms

**Keywords and phrases** Distributed Ledgers, Interoperability, Atomic Appends, Rational Clients, Fault-tolerance

**Digital Object Identifier** 10.4230/LIPIcs.SCNDS.2018.x

**Category** Keynote talk

**Funding** Partially supported by the Spanish Ministry of Science, Innovation and Universities grant DiscoEdge (TIN2017-88749-R), the Regional Government of Madrid (CM) grant Cloud4BigData (S2013/ICE-2894) co-funded by FSE & FEDER, the NSF of China grant 61520106005, and by funds for the promotion of research at the University of Cyprus.

**Acknowledgements** We would like to thank Kishori Konwar, Michel Raynal, and Gregory Chockler for insightful discussions.





## 1 Introduction

Blockchain systems, cryptocurrencies, and distributed ledger technology (DLT) in general, are becoming very popular and are expected to have a high impact in multiple aspects of our everyday life. In fact, there is a growing number of applications that use DLT to support their operations [20]. However, there are many different blockchain systems, and new ones are proposed almost everyday. Hence, it is extremely unlikely that one single DLT or blockchain system will prevail. This is forcing the DLT community to accept that it is inevitable to come up with ways to make blockchains interconnect and interoperate.

The work in [6] proposed a formal definition of a reliable concurrent object, termed Distributed Ledger Object (DLO), which tries to convey the essential elements of blockchains. In particular, a DLO is a sequence of records, and has only two operations, append and get. The append operation is used to attach a new record at the end of the sequence, while the get operation returns the sequence.

In this work we initiate the study of systems formed by multiple DLOs that interact among each other. To do so, we define a basic problem involving two DLOs, that we call *the Atomic Append problem*. In this problem, two clients want to append new records in two DLOs, so that either both records are appended or none. The clients are assumed to be selfish, but rational and risk-averse [17], and may have different incentives for the different outcomes. Additionally, we assume that they may fail by crashing, which makes solving the problem more challenging. We observe that the problem cannot be solved in some system models and propose algorithms that solve it in others.

## 2 Related Work

The Atomic Append problem we describe above is very related to the multi-party fair exchange problem [8], in which several parties exchange commodities so that everyone gives an item away and receives an item in return. The proposed solutions for this problem rely on cryptographic techniques [13,15] and are not designed for distributed ledgers. In this paper, as much as possible, we want to solve Atomic Appends on DLOs via their two operations append and get, without having to rely on cryptography or smart contracts.

Among the first problems identified involving the interconnection of blockchains was Atomic Cross-chain Swaps [12], which can also be seen as a version of the fair exchange problem. In this case, two or more users want to exchange assets (usually cryptocurrency) in multiple blockchains. This problem can be solved by using escrows, hashlocks and timelocks: all assets are put in escrow until a value $x$ with a special hash $y = hash(x)$ is revealed or a certain time has passed. Only one of the users knows $x$, but as soon as she reveals it to claim her assets, everyone can use it to claim theirs. Observe that this solution assumes synchrony in the system.

This technique was originally proposed in on-line fora for two users [1], and it has been extensively adapted and used [16]. For instance, the Interledger system [10] will use a generalization of atomic swaps to transfer (and exchange) currency in a network of blockchains and connectors, allowing any client of the system to interact with any other client. The Lighting network [14, 18] also allows transfers between any two clients via a network of micro-payment channels using a generalized atomic swap. Both Interledger and Lighting route and create one-to-one transfer paths in their respective networks. Herlihy [12] has formalized and generalized atomic cross-chain swaps beyond one-to-one paths, and shows how multiple cross-chain swaps can be achieved if the transfers form a strongly connected directed graph.

Unlike in most blockchain systems, in Hyperledger Fabric [4, 5] it is possible to have transactions that span several blockchains (blockchains are called *channels* in Hyperledger Fabric). This allows solving the atomic cross-chain swap problem using a third trusted channel or a mechanism similar to a two-phase commit [5]. Additionally, these solutions do not require synchrony from the system. The ability of channels to access each other's state and interact is a very interesting feature of Hyperledger Fabric, very in line with the techniques we assume from advanced distributed ledgers in this paper. Unfortunately, they seem to be limited to the channels of a given Hyperledger Fabric deployment.

There are other blockchain systems under development that, like Hyperledger Fabric, will allow interactions between the different chains, presumably with many more operations than atomic swaps. Examples are Cosmos [2] or PolkaDot [3]. These systems will have their own multi-chain technology, so only chains in a given deployment can initially interact, and other blockchain will be connected via gateways. Another proposal for interconnection of blockchains is Tradecoin [11], whose target is to interconnect all blockchains by means of gateways, trying to reproduce the way Internet works. Since the gateways will be clients of the blockchains, the functionality of the global interledger system will be limited by what can be done from the edge of the blockchains (i.e., by the blockchains' clients).

## 3 Contributions

As mentioned above, in this paper we extend the study of the distributed ledger reliable concurrent object DLO started in [6] to systems formed of several such objects. Hence, the first contribution is the definition of the Multiple DLO (MDLO) system, as the aggregation of several DLOs (in similar way as a Distributed Shared Memory is the aggregation of multiple registers [19]). The second contribution is the definition of a simple basic problem in MDLO systems: the *2-AtomicAppends problem.* In this problem, the objective is that two records belonging to two different clients are appended to two different DLOs atomically. Hence, either both records are appended or none is. Of course, this problem can be generalized in a natural way to the $k$-Atomic Appends problem, involving $k$ clients with $k$ records and up to $k$ DLOs.

Another contribution, in our view, is the introduction of a crash-prone risk-averse rational client model, which we believe is natural and practical, especially in the context of blockchains. In this model, clients act selfishly trying to maximize their utility, but minimizing the risk of reducing it. We consider that this behavior is not a failure, but the nature of the client, and any algorithm proposed under this model (e.g., to solve the 2-AtomicAppends problem) must guarantee that clients will follow it, because their utility will be maximized without any risk. For a complete specification of the clients' rationality their utility function has to be provided. Two utility models are proposed. In the *collaborative utility model*, both clients want the records to be appended over any other alternative. This resembles the *Coordinated Attack* problem [9], in which two armies need to agree on attacking a common enemy, and the desired outcome is obtain when both attack. In the *competitive utility model* a client still wants both records appended, but she prefers that only the other client appends. This resembles the *Atomic Swaps* problem [12] discussed above. This client model is complemented with the possibility that clients can fail by crashing.

We explore hence the solvability of 2-AtomicAppends in MDLO systems in which the DLOs are reliable but may be asynchronous, and the clients are rational but may fail by crashing. The first results we present consider a system model in which clients do not crash, and show that Collaborative 2-AtomicAppends can be solved even under asynchrony, while

Competitive 2-AtomicAppends cannot be solved. Then, we further study Collaborative 2-AtomicAppends if clients can crash. In the case that at most one of the two clients can crash, we show that, if each client must append its own record (what we call *no delegation*), Collaborative 2-AtomicAppends cannot be solved even under synchrony. This justifies exploring the possibility of *delegation:* any client can append any record, if she knows it. We show that in this case Collaborative 2-AtomicAppends can be solved, even if the system is asynchronous (termination is only guarantee under synchrony, though). However, delegation is not enough if both clients can crash, even under synchrony.

The negative results (for Competitive 2-AtomicAppends even without crash failures and for Collaborative 2-AtomicAppends with up to 2 crashes) justifies exploring alternatives to appending directly or delegating among clients. Hence, we propose the use of an entity, external to the clients, that coordinates the appends of the two records. In fact, this entity is a special DLO with some level of intelligence, which we hence call *Smart DLO* (SDLO). The SDLO is a reliable entity to which clients can delegate (via appending in the SDLO) the responsibility of appending their records to their respective DLOs when convenient. The SDLO hence collects all the records from the clients and appends them. Since the SDLO is reliable, all the appends will complete. If some record is missing, the SDLO issues no append, to guarantee the properties of the 2-AtomicAppends problem. Thus, the SDLO can be used to solve Competitive and Collaborative $k$-AtomicAppends even when all clients can crash. Full details can be found in [7].

We believe that SDLO opens the door to a new type of interconnection and interoperability among DLOs and blockchains. While the use of oracles to access external information in a smart contract (maybe from another blockchain) is widely known, we are not familiar with blockchain systems in which one blochchain (i.e., possibly a smart contract) issues transactions in another blockchain. We believe this is a concept worth to be explored further.

## References


1 Atomic swap. `https://en.bitcoin.it/wiki/Atomic_cross-chain_trading`.

2 Cosmos. `https://cosmos.network`.

3 PolkaDot. `https://polkadot.network`.

4 Elli Androulaki, Artem Barger, Vita Bortnikov, Christian Cachin, Konstantinos Christidis, Angelo De Caro, David Enyeart, Christopher Ferris, Gennady Laventman, Yacov Manevich, Srinivasan Muralidharan, Chet Murthy, Binh Nguyen, Manish Sethi, Gari Singh, Keith Smith, Alessandro Sorniotti, Chrysoula Stathakopoulou, Marko Vukolic, Sharon Weed Cocco, and Jason Yellick. Hyperledger fabric: a distributed operating system for permissioned blockchains. In *Proc. of the 13th EuroSys Conference (EuroSys)*, pp. 30:1–30:15, 2018.

5 Elli Androulaki, Christian Cachin, Angelo De Caro, and Eleftherios Kokoris-Kogias. Channels: Horizontal scaling and confidentiality on permissioned blockchains. In *Proc. of the 23rd European Symposium on Research in Computer Security (ESORICS)*, pp. 111-131, 2018.

6 Antonio Fernandez Anta, Chryssis Georgiou, Kishori Konwar, and Nicolas Nicolaou. Formalizing and implementing distributed ledger objects. In *Proc. of the 7th International Conference on Networked Systems (NETYS)*, 2018. Also, in SIGACT News, 49(2):58-76, June 2018.

7 Antonio Fernandez Anta, Chryssis Georgiou, and Nicolas Nicolaou. Atomic Appends: Selling Cars and Coordinating Armies with Multiple Distributed Ledgers. In *arXiv: 1812.08446*, 2018.

8 Matthew K. Franklin and Gene Tsudik. Secure group barter: Multi-party fair exchange with semi-trusted neutral parties. In *Proc. of the 2nd International Conference on Financial Cryptography*, pp. 90–102, 1998.

9 Piotr J. Gmytrasiewicz and Edmund H. Durfee. Decision-theoretic recursive modeling and the coordinated attack problem. In *Proc. of the 1st International Conference on AI Planning Systems*, pp. 88–95, 1992.

10 Interledger W3C Community Group. Interledger. `https://interledger.org/`.

11 Thomas Hardjono, Alexander Lipton, and Alex Pentland. Towards a design philosophy for interoperable blockchain systems. In *arXiv: 1805.05934*, 2018.

12 Maurice Herlihy. Atomic cross-chain swaps. In *Proc. of the 2018 ACM Symposium on Principles of Distributed Computing (PODC)*, pp. 245–254, 2018.

13 Silvio Micali, Michael O. Rabin, and Joe Kilian. Zero-knowledge sets. In *Proc. of the 44th Symposium on Foundations of Computer Science (FOCS)*, pp. 80–91, 2003.

14 Andrew Miller, Iddo Bentov, Ranjit Kumaresan, Christopher Cordi, and Patrick McCorry. Sprites and state channels: Payment networks that go faster than lightning. In *arXiv:1702.05812*, 2017.

15 Aybek Mukhamedov, Steve Kremer, and Eike Ritter. Analysis of a multi-party fair exchange protocol and formal proof of correctness in the strand space model. In *Proc. of the 9th International Conference on Financial Cryptography and Data Security*, pp. 255–269, 2005.

16 Arvind Narayanan, Joseph Bonneau, Edward W. Felten, Andrew Miller, and Steven Goldfeder. *Bitcoin and Cryptocurrency Technologies - A Comprehensive Introduction*. Princeton University Press, 2016.

17 Martin J Osborne et al. *An introduction to Game Theory*. Oxford university press, 2004.

18 Joseph Poon and Thaddeus Dryja. The bitcoin lightning network: Scalable off-chain instant payments. `https://lightning.network/lightning-network-paper.pdf`, 2016.

19 Michel Raynal. *Distributed Algorithms for Message-Passing Systems*. Springer, 2013.

20 Matteo Gianpietro Zago. 50+ Examples of How Blockchains are Taking Over the World. *Medium*, 2018.