Received 8 April 2023, accepted 4 May 2023, date of publication 23 May 2023, date of current version 1 June 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3279106

RESEARCH ARTICLE

Before Ethereum. The Origin and Evolution of Blockchain Oracles

GIULIO CALDARELLI
Department of Management, University of Turin, 10124 Turin, Italy
e-mail: giulio.caldarelli@unito.it

ABSTRACT Before the advent of alternative blockchains such as Ethereum, the future of decentralization was all in the hands of Bitcoin. Together with Nakamoto itself, early developers were trying to leverage Bitcoin's potential to decentralize traditionally centralized applications. However, because Bitcoin was a decentralized machine, the available non-trustless oracles were considered unsuitable. Therefore, strategies had to be elaborated to solve the so-called "oracle problem" in the newborn scenario. By interviewing early developers and crawling early forums and repositories, this paper aims to retrace and reconstruct the chain of events and contributions that gave birth to oracles on Bitcoin. The evolution of early protocols, along with the difficulties encountered in their development, are also outlined. Analyzing technical and social barriers to building oracles on Bitcoin, the transition to Ethereum will also be discussed.

INDEX TERMS Bitcoin, blockchain, contracts, oracles, extrinsic data, multi-signature, OP_return.

I. INTRODUCTION

"*That's cheating, though, isn't it?... but all of the really interesting complex contracts I can think of require data from outside the blockchain*" [1]. "Cheating" is how Gavin Andresen provocatively referred to utilizing oracles on the blockchain to run smart contracts. The idea is that in order to be able to finally utilize the blockchain for something above cryptocurrencies, renouncing to a degree of decentralization may be considered a fair take. Whether it is right to leave hard-achieved decentralization and the degree to which it has to be renounced in exchange for more interoperability have yet to be defined [2], [3], [4].

The literature on blockchain oracles is a small niche. Two recent studies show that the total number of academic papers concerning oracles barely reaches two hundred [5], [6]. Academic and practitioner interest in blockchain oracles rose, in fact, after the 2017 initial coin offering (ICO) hype, where hundreds of blockchain integration proposals were launched in almost every sector [7]. Because many have turned out to be fraudulent or unrealistic, studies have arisen on the motives for their infeasibility [8], [9], [10], [11]. An emergent stream of literature guided by the works of Egberts [12], Frankenreiter [13], and Damjan [14] also started to investigate the role of oracles, along with their uses, risks, and legal implications in real-world blockchains. As general awareness of the so-called "oracle problem" increased, many other papers concerning oracle technical structure and classification emerged [2], [15]. The paper by Al-Breiki et al. [16] is one of the first to classify trustworthy blockchain oracle protocols by evaluating their security and the foundations of their trust models. Eskandari et al. [17] and Liu et al. [18] instead focus on oracles used in DeFi. The first provides a theoretical framework to classify them, while the second outlines, by gathering on-chain data, the deviation rates of the different oracle designs. In contrast, recent research by Pasdar et al. [19] involves a consistent number of oracles. It investigates the data type they can provide, their resistance to Sybil attacks, and their exposure to the so-called "verifier dilemma." The dilemma concerns the preference of the verifier to vote for an outcome that guarantees himself a reward instead of performing work for correctness.

It has to be said that since the academic literature on oracles started in 2017–2018, the whole decentralized infrastructure had already shifted from Bitcoin to Ethereum and other alt chains by that time. Therefore, the studies undertaken mainly involved the infrastructure active and observable in that timeframe and onward, thus reflecting a specific philosophy

The associate editor coordinating the review of this manuscript and approving it for publication was Mehdi Sookhak.

and belief. However, the concept of decentralizing applications with blockchains and the use of oracles is intuitively much older [20]. Before the advent of Ethereum, Bitcoin was the leading ecosystem on which early smart contract developers and blockchain enthusiasts experimented with decentralized applications. As data from the real world require oracles, they had to be primarily theorized and built on Bitcoin. Although research in [12] and [20] hinted that oracles on Bitcoin worked with multisignature (multisig), to the best of the author's knowledge, no broader and further studies can be retrieved on how those protocols were theorized and built. Since Bitcoin is much older than Ethereum, it is reasonable to hypothesize that more than one oracle type was active on top of its chain.

The idea of this paper is that the oracle literature broadly misses the transition from Bitcoin to Ethereum, which should also include the blockchain oracle's origin. For this reason, the theoretical background and evolution of oracles may be biased by an investigation of projects already developed on Ethereum and alt chains. Research in [5] supports the view of excessive heterogeneity and confusion in oracle definitions and boundaries. As the oracle origin and underlying idea have yet to be defined, it is arguable that those aspects may be clarified by investigating early-stage proposals and improvements.

In the absence of dedicated academic or gray literature, the author opted for an exploratory study to shed light on the advent of blockchain oracles. The data collection was therefore guided by experts in the field who were among the first to theorize and develop oracles on the Bitcoin blockchain. The motives for creating their protocols and the protocol themselves will be outlined to better understand how blockchain oracle structures have evolved. Where possible, the data provided by the experts were double-checked with available online documents, repositories, emails, and forum posts.

The research questions of this study are as follows:

1) What is the exact origin of blockchain oracles, and how were they theorized?
2) How did the first oracle protocols work, and how did they evolve?
3) What were the difficulties faced by early oracle developers?
4) Which factors mainly drove the shift of oracles development to the Ethereum ecosystem?

The paper proceeds as follows. Section II introduces the methodology and data collection, while Section III outlines the findings. Section IV discusses the results and answers the research questions. Section V concludes the paper by providing hints for further research.

II. METHODOLOGY

Aware of the niceness of the research topic, the original idea was to investigate the origin of blockchain oracles by performing a multivocal literature review, thus blending academic material with practitioner posts and white papers. As a sector in rapid evolution, the blockchain literature includes several papers utilizing this methodological approach [17], [21], [22]. It was soon clear, however, that neither academic nor practitioner articles were written on the topic despite its importance. Conversely to Ethereum-based protocols, no systemized or generalized information could be found for Bitcoin-based protocols. The only available choice was to conduct an explorative study aimed at progressively gathering the required information when relevant data were discovered. The idea was to start with interviews with experts and then expand the data collected with retrieved official documents, emails, forum posts, or specific articles directly suggested by the experts. Therefore, the study's core idea is to identify early oracle developers and let them guide the research by offering additional research material. The author is aware that by not building on prior research methodologies, the study may raise concerns about the reliability and replicability of the results. For this reason, data collection is described as transparently as possible, and the material to which it refers is almost all available online. The only exceptions are the interviews, which, although recorded, are kept private, except for some sentences. Concerning the replicability of the results, given the scarcity of available material, it is arguable that a different research approach would have led to the same data, although probably obtained in a different order.

The involvement and guidance of experts also served to avoid biases from the author's personal background and beliefs. The field was haunted by false information, so their advice was also necessary to distinguish relevant and original contributions from those that were misleading. To limit redundancy and pursue a high quality of data, only people who founded or proposed specific oracle protocols were interviewed, avoiding, for this study, general Bitcoin experts.

When experts were asked for information, they were also asked to provide a link to a written resource as hard proof of the data provided. Interviews with experts were semi-structured. The proposed length was 45 minutes, but given the wideness of the content, the mean interview time was 73 minutes. Some interviews were also split to better accommodate the experts' schedules. Along with direct interviews done with Zoom software due to physical distance, there were communications by email, phone, and instant messages for clarification and further information. All the elaborated content was sent to each expert for final approval.

A. DATA COLLECTION

The research began on August 30, 2022, with the first interview with Edmund Edgar. Edgar is the founder of Reality Keys and, in this [23] online resource, was indicated as the developer who wrote the first lines of code for an oracle on Bitcoin. Above mentioning other experts and oracle protocols, he suggested Bitcointalk (as well as some specific threads) as a valuable resource for finding material and identifying early protocols. The interviews continued with Tomasz Kolinko, the founder of Orisi, on September 20, 2022, and with Mike Hearn on October 27, 2022. The interview with Mike Hearn pointed out two critical resources: Bitcoin Wiki (with the first wiki page discussing blockchain oracles) and the official Nakamoto emails shared with early Bitcoin developers. Given the impossibility of contacting Nakamoto, his emails and forum posts were the only resources on which to rely when speculating on his vision of oracles. Although Bitcoin.org (the first Bitcoin forum founded by Nakamoto) was no longer accessible, with Bitcoin at a very early stage by that time, it is arguable that no specific information could be found on this topic. Furthermore, Nakamoto's first posts on Bitcointalk concerning Bitcoin implementation in real-world applications came out much later. Therefore, it is reasonable to hypothesize that no critical data on the topic were missed. The interview rounds continued with representatives of the Truthcoin and Oraclize projects mentioned in the previous interviews. The research ended with the data collection on Counterparty since the online resources proposed and the projects mentioned became mainly redundant. Concerning the Counterparty section, it has to be said that in addition to the video interview with Adam Krellenstein (one of the founders), another phone interview, several emails, and direct messages were needed to clarify some concepts concerning the OP_Return war. This piece of Bitcoin history included many contradictory sources, and thorough research was required to find unbiased information on the topic. The description of the OP_Return debate made in this study was based on official posts by Bitcoin core developers and developers directly involved in the dispute. Therefore, it is potentially not biased by opinion posts written by third parties. All the materials and projects relevant to this study were retrieved. The only project mentioned by a couple of experts but excluded in this research is Early Temple. Although the original webpage was found on the web archive [24], the retrieved content was insufficient to contribute to this paper, and it was impossible to interact with any founders to further elaborate on the project. The protocol, however, never went into production and was abandoned, with any related material removed. Therefore, its absence should not alter the significance of the study. An overview of the data collected in this study is provided in Table 1.

III. FINDINGS

The present section summarizes the information collected from the interviews and other online material. Unlike the material concerning the origin of blockchain oracles, the other presented projects do not follow chronological order. They were arbitrarily organized based on how they differentiated from the first oracle proposal. The chronological ordering of oracles was seen as unsuitable since an arbitrary ordering criterion had to be chosen (project elaboration, first line of code, first announcement, website, or whitepaper release). Therefore, the author opted for an ordering that could at least improve content delivery. The quotations from each paragraph (unless stated otherwise) are from the experts indicated in Table 1. After each section that describes a specific protocol, a dedicated subparagraph outlines the constraints faced and difficulties encountered.

TABLE 1. Data collection.

Interviews			
Interviewed Expert	Date	Duration	Project/Topic
Edmund Edgar	31/08/2022	36 m	Oracle History
/	22/12/2022	1 h 04 m	Reality Keys
Tomasz Kolinko	20/09/2022	1 h 12 m	Orisi
Mike Hearn	27/10/2022	41 m	Oracle History
Paul Sztorc	27/10/2022	1 h 39 m	Truthcoin
Thomas Bertani	03/11/2022	45 m	Oraclize
Adam Krellenstein	28/11/2022	31 m	Counterparty Protocol
/	09/01/2023	51 m (Phone Call)	Protocol oracle Module and OP_Return war
Online material			
Source	Subject		Reference
Bitcoin.com/Satoshi Archive	Chargebacks, Contracts, Extrinsic data, trusted third parties.		[25], [26], [51], [62], [82]
Bitcointalk	Satoshi escrow proposal, bitDNS, oracle announcements/discussions, OP_Return		[36], [42], [52], [64], [83]
BitcoinWiki	Contracts, OP_Return, Multi-signature		[27], [28], [84], [85]
GitHub	Reality Keys, Orisi, Provable and Counterparty protocols, OP_Return		[31], [33], [38], [40], [41], [69], [75]
Whitepapers	Bitcoin, Bitmessage, Truthcoin, Mastercoin, Blockstream.		[32], [54], [56], [81], [86], [87]
Other	Bitcoin wallets, Oracle discussion, OP_Return war, Bitcoin API implementations, Meta chains.		[1], [23], [66], [68], [73], [88], [29], [37], [39], [43]–[47]
Academic Publications/Monographies			
Journal Papers	Non-standard transactions, OP_Return, LMSR.		[34], [57], [58], [67], [76]
Books	Bitcoin scripts, Smart Contracts, Focal points.		[35], [61], [70]

A. THE NAKAMOTO CONTRACTS

According to the data gathered in this study, the origin of blockchain oracles can be traced back to 2011 from some interaction between Satoshi Nakamoto and Mike Hearn, an early Bitcoin developer. As the young developer found some unusual piece of code in Bitcoin, he queried Nakamoto for clarification. The lines of code emerged as the base structure for something new called "contracts" [25]. Contracts were dynamic agreements based on the Bitcoin blockchain, for which a certain amount of Bitcoin was moved to predetermined addresses when specific conditions were met. The original example of a contract theorized by Nakamoto [25] concerns an unregistered and open transaction (Fig. 1) that can be updated multiple times until a specific deadline (nLockTime). It was supposed to work as follows:

- The contract contains payments from multiple parties and has a deadline for it to be broadcast.
- To rewrite the contract, each party must sign the transaction with a higher sequence number (#1, #2,…#n) before the deadline. The party adding resources to the contract (e.g., Bitcoin) should be the first to sign the

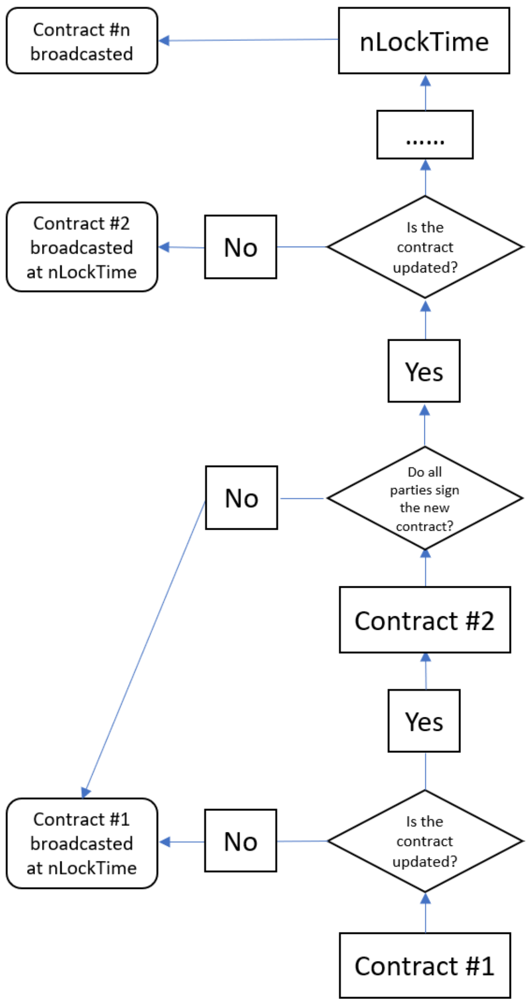


FIGURE 1. Graphical conceptualization of Nakamoto's contracts.

updated version. As per Nakamoto's idea, by signing, the input owner says, "I agree to put my money in if everyone puts their money in, and the outputs are this."

- Leveraging SIGHASH_SINGLE instead, a user can lock a specific output in the contract de facto, bowing out of negotiations concerning outputs except for a specific one. By using SIGHASH_NONE, a user declares to agree to any chosen output and therefore withdraws from any ongoing negotiations.
- A pre-agreed default option can also be created utilizing a higher sequence number and OP_CHECKMULTISIG. Since it is based on multisig and therefore validated by a subset of parties, it can be used as insurance if one of the parties is unresponsive or refuses to sign the transaction.
- At the deadline, the contract that will be broadcast is not necessarily the one with the higher sequence number but the last one on which all parties (or the valid subset) agreed.

It must be noted that there is an essential difference between using SIGHASH_NONE and refusing to sign a transaction. In the first case, the user opts out of negotiation and agrees to any condition, de facto contributing to the successful execution of the contract. In the latter case, by not signing the transaction, the user opts out of negotiations by rejecting any condition, de facto impeding contract execution.

The contracts theorized by Nakamoto had, then, as conditions, specific actions made by users (e.g., deciding if and how to sign a transaction) and a predetermined amount of time passed. In reply to Mike Hearn's requests for clarification, Nakamoto also shared his vision of the possibility of Bitcoin scripts being conditional on extrinsic data [26]. His opinion was that if Bitcoin had access to outside data, which may change between nodes, it could generate a fork in the chain. Arguably, he was not keen on that hypothesis. Furthermore, commenting on the chance of having contracts with known and trusted entities, such as Google, he suggested that those had to be executed in a trustless way by making trusted entities sign the contract prior to its creation. Therefore, no particular advantage should have been granted to trustworthy entities in the real world.

B. THE FIRST CONCEPTUALIZATION OF A BLOCKCHAIN ORACLE

By the time Nakamoto left the Bitcoin community, the code to support contracts was insufficient to execute them fully, and no indications were left on how to continue their development. In light of contributing to their development, Mike Hearn started a Bitcoin Wiki page concerning the so-called contracts on May 22, 2011, defining them as "a method of using Bitcoin to form agreements with people, anonymously and without the need for trust." Although he was the main contributor to the page, other developers (often using pseudonyms) also contributed. Hearn also had a famous talk concerning Bitcoin extensions and contracts in London in 2012, which inspired many developers to build on his and Nakamoto's ideas.

An initial example of the contract described on the page was the promise of later payment using data from the blockchain (e.g., timestamp) to determine the exact time to unlock the money (Timelocks). Later, Hearn also inserted an example of a "will." A will contract concerns the event of death and therefore introduces arbitrary data on the blockchain for the first time. Due to that new contract type, on July 25, 2011, Mike Hearn also added the concept of oracle to the wiki page, explaining that "as Bitcoin nodes cannot measure arbitrary conditions, we must rely on an 'oracle.'" In the same contribution, an oracle was defined as "a server that has a keypair, and signs transactions on request when a user-provided expression evaluates to true" [27].

In the will example described in the wiki, the oracle was the third key owner of an M-of-N multisig contract that signed the transaction when the condition death = true. The will contract (illustrated in Fig. 2) was meant to work as follows:

The creator of the will (e.g., grandfather for grandson) would create a transaction spending the output and setting the output to:

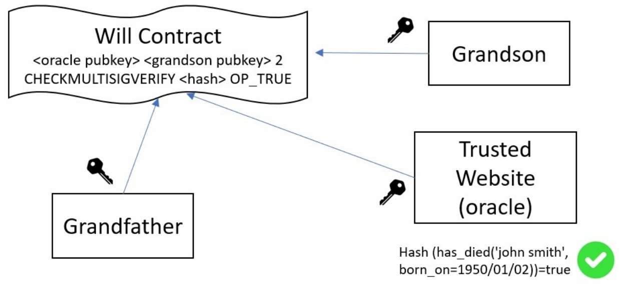

Will Contract
<oracle pubkey> <grandson pubkey> 2
CHECKMULTISIGVERIFY <hash> OP_TRUE
Grandson
Trusted
Website
(oracle)
Grandfather
Hash (has_died('john smith',
born_on=1950/01/02))=true

FIGURE 2. **Will contract relying on a trusted external data source.**

<oracle pubkey> <grandson pubkey> 2 CHECKMULTISIGVERIFY <hash> OP_TRUE

This means that the transaction is complete from the grandfather's side, but its expendability is conditional on the script's output mentioned above. The script requires two other key owners to sign the transaction when a specific hash is verified.

The oracle then accepts the request and receives the expression and a copy of the partially complete transaction along with the output script. The oracle pubkey would be published on the oracle website, which is meant to be a trusted data source (in this case, it concerns people's deaths). Then, the sentence about the grandfather's death should be in a form that the oracle can understand (e.g., a hash). Ideally, it may be a hashed form of the string:

has_died('john smith', born_on=1950/01/02)

The oracle then verifies whether the hash of the expression matches the hash of the output scripts, and if it does, it signs the transaction. Otherwise, it returns an error.

Assuming that the grandson has already signed his part of the script, when the oracle successfully signs its part, the grandson can broadcast the contract transaction and the money claim [27], [28].

A critical feature of this example is that the creator of the will contract decides the oracle that can unlock the funds.

- DIFFICULTIES AND LIMITATIONS

Mike Hearn's approach to oracles was purely theoretical, and he never developed an actual oracle. As oracles were known as black boxes, Hearn's efforts were toward "*how to make that black box a little more transparent*." As for the reason why he used the name oracle, he replied, "*The name itself (oracle) is a bit of everything, just like contracts; all these things are metaphors. I think I used it because there is a history of using that term in the field of cryptography, and what I was developing was similar to the concept of random oracle.*"

The main limitation of the development of newborn contracts was the absence of a Bitcoin wallet with contract support. In 2012–2013, the wallet market was, in fact, small and fragmented. In 2014, Bitcoin wallets were banned by Apple [29]. Aiming for a solution, Mike Hearn started the development of Bitcoin-J, a wallet that could successfully support contracts. Unfortunately, other difficulties were faced, of which Hearn mentioned as the most problematic:

- Development Hardness – Contracts, as a new type of programming, were not easy to develop. They involved cryptography, which is something programmers were not always very familiar with.
- Non-interoperable wallets – As developers were developing their own wallets, there was an evident lack of interoperability. Programs hardly worked on one wallet and rarely worked on others.
- Incomplete contributions – Many contracts were just proof of concept, not even integrated into a wallet, and often only executable from the command line. As the final integration into a wallet was difficult, the developers were not doing so.

As per Hearn's advice, apart from being an important experiment, oracle mechanisms are unlikely to be broadly used in the future. He sees higher potential in the "trusted computing" area he has dedicated to after leaving the Bitcoin community in 2015.

C. AN EARLY STANDARDIZATION OF THE ORACLE API (ORACLIZE)

The most direct extension of Hearn's proposal was Oraclize. The protocol was developed by Thomas Bertani, who came across both the Hearn wiki page and the London conference video and was "fascinated by those complex conditional transactions."

Bertani's main concern and purpose were to find a practical solution to the problem of feeding automated transactions with real-world data. The first version of what later became Oraclize, in fact, was based on the concept of creating a pre-authorized Bitcoin transaction by partially signing it and having the oracle put the second signature when a certain condition was met. What Oraclize offered was the possibility of allowing any application programming interface (API) to be leveraged as a blockchain oracle, also providing proof that the data fetched were not altered in the process.

When Ethereum was launched in May 2015, the Oraclize interface was adapted to run on the new blockchain, with the first tests done in August directly on the main net. It was soon actively used in smart contracts, as Bertani declared, "the oracle to get data from the APIs was something that got much traction. We had a peak of tens of thousands of transactions every month to get data about all different things" [30]. By then, both Bitcoin and Ethereum versions were live and available on the Oraclize website. For Bitcoin, there was an API with a point-and-click interface and a dedicated library. For Ethereum, however, there was solidity integration. To expand the use of the oracle, Oraclize implemented an authenticity proof to validate the fact that the oracle behaved honestly. It was called "honesty proof" at the beginning. Still, the name was soon rebranded, given the fact that as data source reliability was out of oracle's control, it could have created false expectations. This new

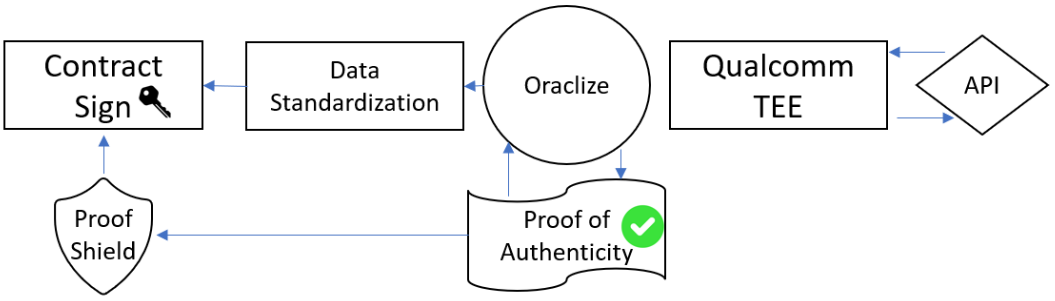


FIGURE 3. Oraclize data fetching and contract signing with proof shield on Bitcoin.

feature guaranteed that the data provided matched the data drawn from the source.

Oraclize in Bitcoin can be leveraged using conditional transactions and a pay-to-script-hash (P2SH). The Bitcoin script had to include the condition (or set of conditions), the required signature to be redeemed, the data source, the outcome (or set of outcomes), and possibly an expiration date so that in case of oracle malfunction or inability of parties to sign the transactions, the funds are returned to their owners.

The following example, taken from the protocol library and formalized in Fig. 3, outlines its use [31].

The two agents (Alice and Bob) establish that if the temperature in Milan (Italy) is above 10 degrees or if it rains from the contract is established until the next 24 hours, Bob can unlock the funds; otherwise, Alice can. They establish that the conditions are checked hourly via Wolfram Alpha until a condition is matched or the time limit elapses.

In this contract, we then have the following:

- A number of **agents**: Alice, Bob, and Oraclize. A fourth agent (e.g., Carol) can be the arbitrator if one of the others is unreachable or inactive. Otherwise, an nLockTime script can establish the refund of the money after a certain amount of time.
- Pre-established **conditions** and **outcomes**. The conditions are the temperature in Milan above 10 degrees and the event of rain. In both cases, the outcome is that Bob can unlock the funds. If both conditions are not verified within the timeframe, the outcome is that Alice takes the funds. Action outcomes can overlap, but conditions should not. This is done to avoid ambiguity in contracts and certain types of attacks. For example, if a condition is ambiguous, an oracle can select the most convenient result for selfish purposes.
- The data source in this case is **Wolfram Alp**ha, but the parties can agree on any other source.
- A **Timeframe** in which the contract is active.

The contract resolves when two of the three key owners sign the transaction.

Truthfully, the contract can be written and established without the help of Oraclize, as it can directly point to a web API. Similar to Hearn's solution, the data provider may offer its signature to the data and transaction. What Oraclize does, however, is standardize the data transfer so that any web API can be a data source for the blockchain without any adaptation from its end. Leveraging Qualcomm Trusted Execution Environment (TEE) technology, Oraclize can also guarantee that the data drawn from the web API have not been manipulated. Intuitively, only data sources with SSL encryption can be utilized with Oraclize since in the absence of this level of security, the protocol would be unable to guarantee the reliability of the data due to unforeseen man-in-the-middle attacks. An additional level of security was also developed called "ProofShield." With this feature, it could have been ensured that the proof of authenticity was already correctly verified if a transaction had been signed. Without "ProofShield" on a chain similar to Bitcoin, the verification could not have been enforced but only verified and audited manually at a later time.

- DIFFICULTIES AND LIMITATIONS

Conceptually, the idea of launching Oraclize relied on the fact that during his conferences, Bertani noticed a great interest in automated transactions (contracts) based on real-world events. As oracles were necessary to build contracts, he thought they would have been huge in the short term. On this early thought, however, he realized that "I was wrong. Oracles are still a very long-term problem; I don't think there is a convincing solution at the moment. It's a partially solved problem for very simple use cases such as price feeds."

It was also hard to find developers to work on the project because although they understood the potential of Oraclize, they were skeptical because of script length, high costs, and network congestion. Against skeptical opinions, Bertani managed to continue the development of Oraclize, thanks to some hackathon awards, the first of which was actually won by proposing a half-life insurance model based on Oraclize. The team continued the development of insurance, as well as other ideas, but they soon realized that every application needed a specific solution to the oracle problem. Therefore, they decided to drop all the side projects and focus exclusively on the oracle module.

Due to the already discussed problems of Bitcoin conditional transactions (length, costs, and congestion), Bitcoin integration was eventually dismissed. It must also be noted that it never went into actual production besides being used for testing purposes. The team also considered integration with Reality Keys and Amazon Mechanical Turk as data sources, but given the scarcity of requests, these features did not go into production.

To date, the project's rebranded "Provable" is still under development, and since its inception, it has processed millions of transactions on the Ethereum blockchain, making it one of the longest-running and most widely used oracles.

D. MULTIPLE INDEPENDENT ORACLES ON BITCOIN (ORISI)

A further extension of the Hearn proposal was the Orisi protocol, which considered the possibility of adding multiple trusted data feeds for the first time. Orisi, in principle, was meant to support the launch of a stablecoin on the Bitcoin protocol. To build a stablecoin, a data feed was necessary to constantly update its exchange rate, and by that time, the only available and known oracles were Reality Keys and Truthcoin. Reality Keys was an already operating and reliable oracle project. However, in its early versions, it did not offer the possibility of a data feed, and it was mainly oriented toward a one-off event, such as election results. The other oracle, Truthcoin, was still under development, and, similar to Reality Keys, it was more oriented to one-off events rather than data feeds.

Due to the lack of available data feeds, a new oracle (Orisi) was considered necessary. An entry for the decentralized data feed based on the Orisi protocol was also added to the contract Bitcoin Wiki page on June 9, 2014.

The key innovation of Orisi oracle, compared to previous proposals based on multisig, was to add a "set" of independent oracles (Fig. 4). On the basis of having multiple data providers, it was difficult to bribe more than half of the oracles. In addition, as different entities, they would have implemented various hardware and applications, thus reducing the chance of them all being hacked. A list of trusted oracles was proposed on the platform website, but users could also select other trusted nodes. Instead of direct IP communication, the protocol also implemented "Bitmessage" to protect the identity of oracle nodes and to prevent spam [32]. The following example, involving two agents (Alice and Bob), better clarifies how Orisi differentiates from previous oracles based on multisig.

- Alice promises Bob that if candidate A wins the election, she will give him 10 Bitcoins.
- Both agree that the condition for the payment would be that on a specific website, the election of candidate A is declared. Then, they both agree on a set of seven oracle nodes.
- Alice should then deposit 10 Bitcoins in a multisig address that is considered a "safe" until oracles decide to forward the funds to Bob or to return them to Alice (in case candidate A loses the election). To do so, Alice creates an "unlock" transaction to forward the funds from the safe and pays the fees to the oracles and the Orisi project.

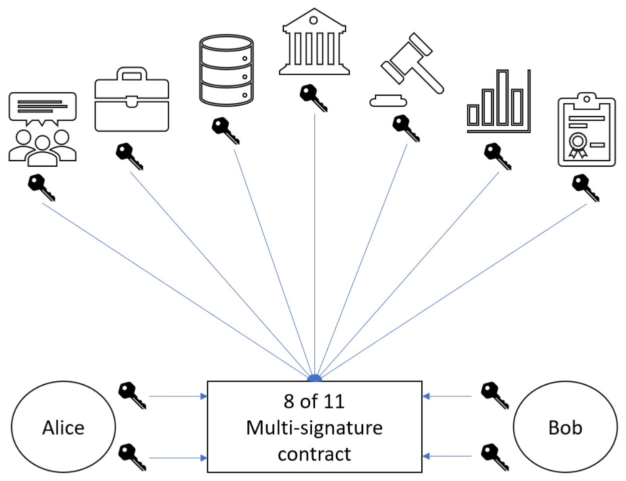


FIGURE 4. Multiple independent oracles with multi-signature.

- The oracles then verify the transaction and the validity of the request. If valid, they add the transaction and notify the agents. If all the oracles acknowledge the validity of the transaction, Alice transfers the 10 Bitcoins to the wallet for the contract to be finally active.
- If candidate A wins the election, then as soon as the oracle nodes notice the information on the website, they sign the transaction that is also broadcasted through Bitmessage. Once enough oracles sign the transaction, Bob can also sign the transaction with his keys and broadcast the transaction to the Bitcoin network to finally unlock the funds.

- DIFFICULTIES AND LIMITATIONS

In the Orisi underlying idea, oracles must be trusted entities from the financial world (e.g., banks and other financial institutions) sending data about real-world asset prices. Therefore, instead of adding only an oracle key, the protocol requires multiple keys to be as decentralized as possible.

However, relying on multisig, technical limits prevented the number of oracles from being large, as was initially planned. The M-of-N multisig protocol is, in fact, not entirely customizable, and there is a limited key combination from which users can choose. The standard multisig was, in fact, two out of three keys, and more complex ones allowed, in theory, a maximum of up to 15 keys. In practice, however, not all key holders can be oracles. For example, if a multisig is three out of five keys and three keys belong to oracles, then all the oracles may decide to send the funds to an arbitrary address. Therefore, if the maximum is 15 keys and a multisig wallet is 12 out of 15, then no more than 11 keys can be in control of the oracles, while users should control the rest. Thus, 1 + (m of n) is turned into (n + 1 of 2n - m + 1) [33]. This intrinsic multisig limitation made it impossible to add more reporters to the oracle, thus dramatically limiting the functionalities of Orisi.

Unfortunately, another issue was encountered in the development of Orisi. The scripts of the Orisi oracle were, in fact, hardly mined for two inherent reasons:

1) First, there was an economic disincentive to mine Orisi transactions because the scripts were larger than usual. As they occupied the weight of many simple Bitcoin transactions, miners could have collected more fees by selecting other transactions instead of Orisi's. Therefore, although a bit more rewarding than a simple Bitcoin transfer, it was unlikely for miners to mine the script voluntarily.
2) Second, and most importantly, by that time, scripts were not a common transaction type. P2SH was introduced in 2012, and part of the community was not keen on inserting scripts on Bitcoin. Many wanted to keep it as a payment system only. Miners feared that processing scripts on the Bitcoin network could have broken the chain and altered the payment system. Therefore, the miners did not relay the scripts, leaving them in the mempool. If, after some time, the transaction was not mined, it was automatically rejected.

Bitcoin miners can actually exert a sort of ''veto'' for which they can arbitrarily decide which transaction to put in their block. Censorship resistance is, however, guaranteed since for a miner who refuses to put a transaction in a block, others will agree to mine it. The chance for all miners to collude to reject a specific transaction is ideally remote. However, some transactions considered ''non-standard'' (e.g., multisig above three keys), although perfectly valid, are unlikely to be mined [34]. Even though the miners did not deliberately collude to reject those transactions, they were so unusual that they naturally decided not to include them. This was a sort of Schelling point [35].

Relying on Bitcoin scripts, unfortunately, despite being legitimate, Orisi transactions were generally not mined. The Orisi team had to search for a compliant mining pool to have their transaction mined. Providentially, they managed to involve Eligius Pool, which had around 7% of the Bitcoin hash rate (as of June 2014). With 7% of the hash rate, Orisi transactions had a 7% chance of being mined, which resulted in one transaction every 8–15 blocks on average.

Finally, due to the multisig limitations and the difficulties in including Orisi transactions in blocks, the Orisi project was abandoned. The multisig limits prevented honest and trusted oracles from joining the project, and the frequency of updates (every one or two hours) made it impossible for Orisi to serve as a price feed for a stablecoin.

E. BRIDGING REALITY TO THE BITCOIN BLOCKCHAIN (REALITY KEYS)

Reality Keys was a project that proposed a different alternative to the scheme outlined by Mike Hearn. The oracle protocol was developed by Edmund Edgar, who was, at that time, trying to implement Bitcoin as an official currency for OpenSim (an open alternative to Second Life). As he came across the Mike Hearn London talk, he started following many discussion threads on Bitcointalk about the need for a trusted oracle for Bitcoin to build real-world applications [36]. He noticed, however, that it had yet to be an official practical implementation of this idea, so he started working on his own. The first lines of code for Reality Keys were written in late 2013, and the project was released in early 2014. Its ecosystem was built in response to the need of that time to create a bridge from blockchain to the real world and strict Bitcoin technical constraints. Congesting and eventually breaking the chain with these new applications was the primary concern; therefore, Edmund thought about making its oracle ecosystem ultimately work off-chain. Furthermore, given the technical limitations of Bitcoin and adherence to the available scripts, the oracle was set to answer only binary (yes/no) questions.

The following example, conceptualized in Fig. 5, provides an overview of how Reality Keys could be used as an oracle on Bitcoin.

Consider having two agents, Alice and Bob, who wish to bet on Bitcoin prices. Alice bets that by June 1, 2014, the price of Bitcoin would reach or exceed \$400, while Bob, on the contrary, bets that by the same date, the price of Bitcoin would be lower than \$400.

The agents can solve the bet themselves or entrust it to a third-party oracle, such as Reality Keys. If they decide to use Reality Keys oracle, they must make a simple binary (yes/no) question on the oracle website asking whether, by June 1, 2014, the Bitcoin price is above/equal to \$400, which corresponds to Alice's bet. If the oracle replies no, then Bob wins the bet.

Both agents know how and from which source Reality Keys draws the answers, and they both trust the source and the Reality Keys project. Otherwise, they would freely opt for another contract resolution method.

Reality Keys creates two key pairs (public and private) for both yes and no answers. The two public keys are then published on their website. When the selected date comes (June 1, 2014), the Reality Keys system checks the price of Bitcoin on the proposed data source, but the results are published in two stages. First, the system automatically publishes the results (not the key) on its website and waits for an objection period. During this objection period, an agent can ask for a ''human check'' of the results, offering a tip of 10 millibitcoins [37]. Once the objection period has elapsed, the team then publishes the correct private key and deletes the private key corresponding to the false outcome.

From a technical point of view, the role of Reality Keys ends with the publication of the correct private key. However, it is vital to understand what happens or can happen between the publication of the public key and the private key in the Bitcoin network.

Although some examples were offered on the Reality Keys website, there was actually no specific or ''standard'' way of implementing their oracle service. The choice of implementing a standard or non-standard multisig transaction or using a script (P2SH), along with any specific conditions, was

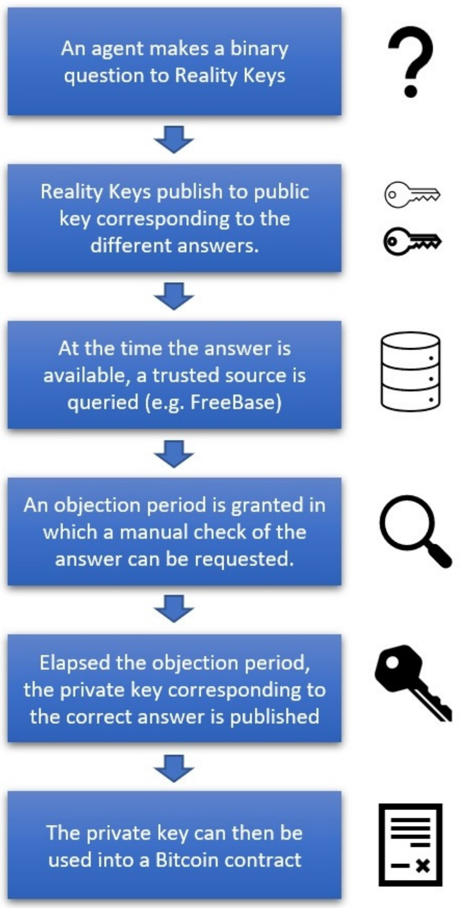

An agent makes a binary question to Reality Keys
Reality Keys publish to public key corresponding to the different answers.
At the time the answer is available, a trusted source is queried (e.g. FreeBase)
An objection period is granted in which a manual check of the answer can be requested.
Elapsed the objection period, the private key corresponding to the correct answer is published
The private key can then be used into a Bitcoin contract

FIGURE 5. **Reality Keys workflow.**

totally in the hands and responsibility of the users. Depending on the selected choices, different costs, technical difficulties, or security standards would have been obtained, for which Reality Keys was not responsible.

One of the few still available demo scripts (realitykeys-demo.py) implement Reality Keys, creating a conditional contract on the outcome of the oracle using pybitcoin-tools [38]. The mentioned commands refer explicitly to the script described in the repository. From the user's side, the steps are as follows:

1) Alice creates a key pair with the command below and sends the public key to Bob. She then funds her address using any Bitcoin client. Bob does the same.

./realitykeysdemo.py makekeys

2) Alice and Bob register a Reality Key and get the ID <reality_key_id> from the Uniform Resource Locator.
3) In case one of the two parties (Alice or Bob) disappeared before completing the transaction, the other party could get the money back from the temporary address with the command.

./realitykeysdemo.py pay <address> -a <amount> -f [<fee>]

4) Alice creates a P2SH address spendable by combining (Alice key + reality key-yes) or (Bob key + reality key-no). Afterward, she creates a transaction, spending the contents of both her and Bob's temporary address to the P2SH address, using her private key. The following output is then sent to Bob for him to sign and broadcast.

./realitykeysdemo.py setup <reality_key_id> <yes_winner_public_key> <yes_stake_amount> <no_winner_public_key> <no_stake_amount>

5) When Bob receives the partially signed transaction, he recreates it to check if the output is the same. If everything is as expected, he signs the transaction and broadcasts it.
6) When the result is issued, whoever wins the bet, Alice or Bob, can execute the following script to unlock the funds from the contract and send them to another address of their choice:

./realitykeysdemo.py claim <reality_key_id> <yes_winner_public_key> <no_winner_public_key> -f [<fee>] -d [<destination_address>]

• DIFFICULTIES AND LIMITATIONS

At the first implementation of Reality Keys, the research for the data and the publication of the correct answer were done directly by the project team and eventually by Edmund himself. However, the users who made the questions knew the data source from which the information was taken. Therefore, despite the system being relatively centralized and not automated, there was a certain degree of transparency. In this regard, however, Edmund specified that although his specific system design was centralized, he hoped the whole blockchain oracle ecosystem would eventually be decentralized. He expected, in fact, many other competitors to show up in the short term. Therefore, if Reality Keys was just one of the available oracles, users could freely select among the most trusted and reliable alternatives.

When the oracle system was ready and running, it served different kinds of requests, from Bitcoin prices to soccer scores. However, there was no specific application managed by Reality Keys until the team uilt a sponsorship integration. They used RunKeeper API to promote walks and marathon-related events. However, personal challenges concerning walks and runs with humanitarian aims were also sponsored. Someone could, for example, challenge himself that if he does not run a certain number of kilometers by a specific date, he has to send some Bitcoin to a charity. Thanks to a system of APIs, Reality Keys can provide information on whether the user has reached his goal.

With these new implementations, the team also faced new challenges. Integration with RunKeeper required, in fact, a dedicated website for the application. The user was then supposed to generate a key, and the website should have been able to perform a transaction with that key. Since a working wallet such as Metamask was not available for Bitcoin, as well as tools for coding, the whole implementation should have been written from scratch. In the end, they managed to complete the website, but as Edgar declared, "This is very hard, and this is very hard to do securely." Unfortunately, the absence of a wallet supporting contracts and developing tools remained almost unchanged on Bitcoin. Since the system was inflexible, there was not much demand for contracts, and since there was not much demand, interest in building those applications was eventually scarce.

The system then remained almost unchanged until the advent of Ethereum, with only binary questions available, but improvement attempts were made on the range of available data types. Freebase was then added as a data source for Reality Keys. Freebase allowed for a wide variety of queries to be made using the structured data system run by Google.

From that point onward, however, the development of Reality Keys switched to Ethereum, mainly for three reasons. First, Google shut down the Freebase website. Since it constituted one of the primary Reality Keys sources of data, it eventually affected its overall utility. Second, the outcome of the block-size war made Bitcoin more expensive to use for contracts, ultimately decreasing the demand for Reality Keys service. Lastly, the implementation of Ethereum could have allowed the switch from a yes/no based platform to a system of signed data of any type to be directly used on-chain.

Since the platform radically changed, the project was rebranded first to Realitio and finally to Reality.eth. Besides the technical differences, the new Ethereum version also had a different theoretical approach. What Reality Keys offered on Bitcoin was simply a bridge between real-world data and on-chain contracts. Therefore, it grabbed existing data from trusted sources (e.g., Freebase) and made it available.

However, the team realized that there was a need for data that did not exist anywhere. Therefore, Reality.eth on Ethereum was meant to provide data that could not be pulled from APIs or websites. The philosophy of the platform switched from delivering data to creating data. Two factors mainly drove the design change:

1) A trusted data source (API) could not be found for specific applications.

2) Other projects became specialized in the bridging process (e.g., Oraclize), and it was not helpful to provide a similar service.

The project, therefore, evolved into its current version, in which it can answer any human language question.

F. BITCOIN ORACLES THROUGH META-CHAINS (COUNTERPARTY)

The standardization of Bitcoin OP_Return in 2013 allowed a broad range of new applications to be built on top of the chain. Therefore, it also opened possibilities for new oracle mechanisms. Leveraging this new feature, Counterparty, a meta-chain with a built-in oracle, was launched in early 2014. A meta chain is a chain in which transaction data is contained on another chain called a master chain. Meta-chain transaction data run after master chain transactions are complete [39].

Counterparty transactions are Bitcoin transactions but with extra metadata. If a blockchain is a book, and blocks are pages, Counterparty software writes information in the margin of those pages. Intuitively, Bitcoin software ignores that extra data; therefore, specific software is needed to read it.

The Counterparty protocol's idea is that when someone signs a transaction with Bitcoin, they add some metadata to the transaction. The content of this metadata is then verified by all Counterparty users to ensure that the transaction is valid. The architectural pattern is called state machine replication.

Let's assume that Alice wishes to transfer five Counterparty tokens to Bob. She will then sign a Bitcoin transaction in which OP_Return (or OP_Multisig) declares a willingness to transfer five Counterparty tokens to Bob. Since Counterparty is a meta-chain, the related Bitcoin transaction will always be confirmed as long as Alice pays the necessary transaction fees. Therefore, if Alice decides to spend five Counterparty tokens and only owns two, the transaction on the Bitcoin blockchain will be confirmed anyway. The corresponding counterparty transaction will instead be marked invalid.

The Counterparty transaction data is retrievable with a block explorer, but it is encoded and appears in a format such as the following:

```
�������6U��A��ᑫ��V��:}!mq����
�m_j�-k?ee
```

The format is not human-readable and must be decoded by the Counterparty engine to be read and digested. The string, in fact, needs to be deobfuscated with the ARC4 Cypher and verified if it starts with CNTRPRTY (first eight bytes). From the 9th byte, information on the transaction type (send, broadcast, and issuance) should be retrieved, followed by the specific transaction data. Fig. 6 outlines the deciphered content of a Counterparty transaction data chunk [40].

Once deciphered, the transaction will be digested by the Counterparty engine, which also verifies its validity.

Being able to inject extrinsic data into the blockchain, the Counterparty engine may already be considered an oracle for Bitcoin. However, on top of Counterparty, applications that further require data from the outside, such as prediction markets or decentralized exchanges, can be built. Data, such as a price feed for a decentralized exchange, are injected into the protocol thanks to the "broadcast" transaction type. A broadcast message publishes textual and numerical information, along with a timestamp. A series of broadcasts from the same address is called a "feed." Intuitively, the

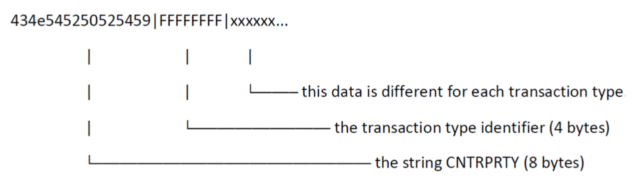
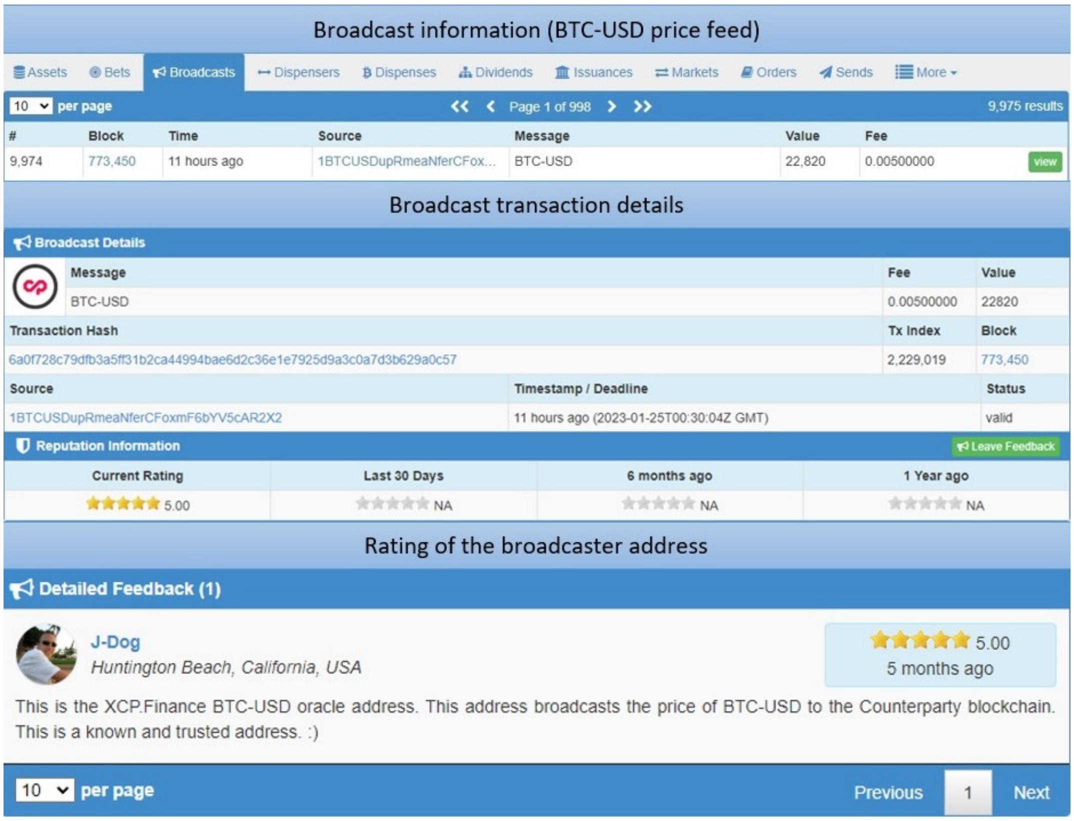

434e545250525459|FFFFFFFF|xxxxxx...
this data is different for each transaction type
the transaction type identifier (4 bytes)
the string CNTRPRTY (8 bytes)

FIGURE 6. Counterparty data chunk.


Broadcast information (BTC-USD price feed)

#	Block	Time	Source	Message	Value	Fee
9,974	773,450	11 hours ago	1BTCUSDupRmeaNferCFox...	BTC-USD	22,820	0.00500000

Broadcast transaction details

Broadcast Details

Message	Fee	Value
BTC-USD	0.00500000	22820

Transaction Hash	Tx Index	Block
6a0f728c79dfb3a5ff31b2ca44994bae6d2c36e1e7925d9a3c0a7d3b629a0c57	2,229,019	773,450

Source	Timestamp / Deadline	Status
1BTCUSDupRmeaNferCFoxmF6bYV5cAR2X2	11 hours ago (2023-01-25T00:30:04Z GMT)	valid

Reputation Information

Current Rating	Last 30 Days	6 months ago	1 Year ago
5.00	NA	NA	NA

Rating of the broadcaster address

Detailed Feedback (1)

J-Dog
Huntington Beach, California, USA

5.00
5 months ago

This is the XCP.Finance BTC-USD oracle address. This address broadcasts the price of BTC-USD to the Counterparty blockchain. This is a known and trusted address. :)

FIGURE 7. **Screenshots of a Counterparty broadcaster (oracle) address, with transaction details and rating mechanism.**

timestamps of a feed should increase monotonically [41]. On the Counterparty Explorer (xchain.io), users can leave feedback and comments on the address that publishes feeds. Fig. 7 provides an overview of what a broadcast price feed shows, the details of the transaction, and feedback. The oracle mechanism described above, developed and available from day one of Counterparty, is usable for a broad range of applications. Users could, for example, also wager on the outcome of a feed placing their bets into an escrow, which is settled when the feed that they rely on passes the chosen deadline [41].

- DIFFICULTIES AND LIMITATIONS

Counterparty was already launched with full functionalities on day one. Features were added quickly except for one, as the chief developer said: "The most difficult thing to add was the decentralized and trustless gaming in the form of rock paper scissors. But we had a decentralized exchange on day one, and it was already working when we launched it." In line with other projects at that time, Counterparty was announced on Bitcointalk [42]. Still, the developer team decided to stay anonymous at the beginning and opt for a proof-of-burn to launch their currency (Counterparty), de facto renouncing to raise any money for their project. The reasons for those choices were mainly the following:

1) The choice reflected what Nakamoto did with Bitcoin: staying anonymous and renouncing any reward for his project.
2) There were high concerns about the legal implications of raising capital with cryptocurrencies, "which turned out to be not very serious."
3) There were personal concerns about the project's development and how it could have turned out, as unforeseen events may have damaged personal reputation of developer team members.
4) Replicating bitcoin issuance (electricity consumption), they wanted to burn resources (bitcoins) instead of transferring resources.
5) They were against raising capital for a project in the alpha–beta stage. "We didn't want to raise money during the development as we thought it was dishonest."

Initially, the proof of burn was supposed to work by consuming bitcoin as fees for the miners, de facto not really destroying bitcoins. However, many community members on Bitcointalk argued that miners could have exploited their position to produce Counterparty tokens unlimitedly. Understanding that it was an actual threat, the developer team decided to change the burning mechanism and made it by transferring Bitcoin tokens to an address whose private keys were unknown (e.g., an impossible vanity address), de facto making them permanently unspendable.

Above the hardships of developing a project without funding, Counterparty suffered the effect of a dispute labeled the OP_Return war [43]. To add the required transaction data, Counterparty needed an OP_Return size greater than the 40 bytes the Bitcoin core developers set in the official v0.9.0 release [44]. Counterparty utilized the shrunk OP_Return feature, but the limited size also forced them to use others, such as multisig, to make their protocol work. Multisig was designed for features such as escrow payments, but the second signature could be leveraged to store data instead [45]. However, this workaround drew the attention of the opposing faction of the OP_Return war.

In March 2014, Luke Dashjr, a Bitcoin core developer and owner of a mining pool, started to filter all Counterparty transactions (eventually without success). As Luke declared, this censorship's motivation was to prevent the exploitation of network resources by Counterparty [42], [43]. However, although probably beneficial for Bitcoin nodes, this decision was criticized because Luke was also a co-founder of Blockstream, a major Counterparty competitor [46], [47].

Although OP_Return size was increased at a later date, the development of Counterparty was inevitably affected by this limit [43]. Furthermore, above the constraints resulting from the reduced payload size, the most significant consequence of this debate was the fear of censorship of the Counterparty protocol. The widespread climate of uncertainty prevented developers from building on Counterparty, further impacting its development and competitiveness. It should be noted that Ethereum, the second biggest network after Bitcoin, was negatively affected by OP_Return limit in its development, as Vitalik Buterin argued on social media. Although some considered it an overclaim, Vitalik declared that the original idea of Ethereum was a "counterparty-style metacoin on top of Primecoin. Not Bitcoin because the OP_RETURN wars were happening" [48], [49].

The launch of Ethereum also had an impact on the development of Counterparty. The main innovation of Ethereum was not smart contracts but the virtual machine, along with the language to write smart contracts. Counterparty had smart contracts, but only those written and supported by its developers. It was possible to code more smart contracts, but every application built had to be part of the protocol. Ethereum, on the other hand, had an extensible infrastructure that allowed anyone to write their own smart contracts. Only the language was part of the protocol, and applications could be deployed on top of it. As Krellenstein stated: "It is a more elegant and flexible system…but ultimately does the same thing."

Aware of the value of the Ethereum virtual machine (EVM), it was ported to Counterparty (EVMParty) so that Ethereum smart contracts could be run on Bitcoin via Counterparty [50]. However, an official version was never released due to multiple factors. In the beginning, there was the idea that the user base would have been minimal since few people were building on Ethereum, and most of the applications were still on Bitcoin. After, when developers started to move to Ethereum, it was clear that Bitcoin could not compete in the smart contract field. First, contracts would have been slow due to Bitcoin block time, even if they were more user friendly with the introduction of EVM. Second, due to the block-size war, the price of Bitcoin increased, while Ethereum was very cheap. Therefore, nobody would have preferred Bitcoin to build contracts. To date, Counterparty is still an active project on Bitcoin and wields the same structure and premises as when it was built.

G. ENABLING ORACLES ON BITCOIN SIDECHAINS (TRUTHCOIN)

Shortly before his last communication [26], Nakamoto commented on the possibility of broadening the range of applications built on Bitcoin. He was planning an "eBay style" marketplace built on top of Bitcoin but with the same mechanism of review and ratings of modern intermediary platforms. However, due to Bitcoin's "locked-in nature," he shared the idea of utilizing other chains (e.g., sidechains) with more developer-friendly rules but with the same miners as Bitcoins. He suggested that inputs for the other chains could have been data from Bitcoin blocks (e.g., the nonce) to achieve interconnectedness between the two chains. The alternative chain to which Nakamoto referred to in his example was "BitDNS," an early sidechain proposal apparently unrelated to current projects sharing the same name [51], [52]. As communications from Nakamoto halted thereafter, no information was retrieved on how he wished to transfer extrinsic data on sidechains.

Ideally, as chains with new and more developer-friendly rules, sidechains would also have allowed new oracle mechanisms. The first to theorize a sidechain as an oracle ecosystem was Paul Sztorc. Disappointed with the closure of InTrade [53], he resolved to find a way to leverage Bitcoin technology to launch an open and uncensorable prediction market. Other than the markets themselves, it would have included a new peer-to-peer "oracle" (Truthcoin) to resolve them without trusted third parties.

In its whitepaper, Truthcoin is described as a "proof-of-work sidechain that collects information on the creation and state of Prediction Markets (PMs)" [54]. Compared to other oracle mechanisms directly built on Bitcoin, Truthcoin shows a higher level of complexity that could never have been reached at that time without leveraging sidechains.

To gather reliable data, the Truthcoin protocol exploits the concept of "salience." The Oxford Dictionary defines salience as the quality of being particularly noticeable or important [55]. Salient information is something that should be well known by anyone. The solution proposed in Truthcoin to achieve salience is based on time. The idea is that information is certain and true after a certain amount of time. Instead of providing a piece of information as soon as it is known, the idea is to provide it at a point where it is undoubtedly certain. Technically, it is organized as follows:

Two coins are present on Truthcoin: CashCoins and Votecoins. CashCoin is pegged 1:1 to Bitcoin (via sidechain) and allows users to create, buy, and sell PM shares. VoteCoins represent each user's reputation, are tradable, and pay dividends over time. VoteCoin allows users to vote on PM decisions and collect PM fees. The ownership of VoteCoins can only change due to the effect of voting activity. In the whitepaper, the totality of voters is referred to as a "corporation." The concept behind this is that the reputation of the entire system is more relevant than the reputation of each individual. VoteCoins are not mined but are proportionally shared among voters in such a way that if someone acquires some VoteCoins, someone else has less, as its total amount remains constant.

Two types of decisions are supported on Truthcoin: binary (0,1) and scalar (Xmin, Xmax). A third state (0.5) identifies decisions that are non-resolvable or confusing. Four entities are present on the platform:

Authors: Users who create a prediction market and provide initial liquidity. The difficult work of an author lies in finding a market that may attract many users and identifying a decision that will be well-known to the voters after a specific time.

VoteCoin Owner: User who votes on a decision. Their main task is to maintain or increase their reputation.

Traders: Users who trade on any PMs; they are the customers of the platform.

Miners: Those who mine blocks on the sidechain. For a Bitcoin sidechain that allows merged mining (hash reuse), Bitcoin miners can mine sidechain blocks at negligible costs.

The resolution of a market and a decision on its "true" outcome, as described in Fig. 8, are as follows.

An author adds a decision specifying the topic and the time of resolution, and then waits for the transaction to be included in a block. When a decision is added, the author can also add a market that provides initial liquidity. Then, they wait for the transaction to be included in a block. When a market is added, trading begins. The market can be advertised so that users can buy and sell the shares of the different market states (such as "yes" and "no"). Eventually, the event occurs and becomes "observable." After this, when the time specified by the creator has passed, the decision is considered "mature." The set of all mature decisions is called a "ballot." Staking their tokens, owners of VoteCoins are called to vote for all the decisions in the ballot, and when votes are revealed, VoteCoins staked are frozen. The decision is then resolved according to the consensus algorithm, which also reallocates VoteCoins. After a decision is resolved, a waiting time of one week starts. Once the waiting time expires, another phase starts in which miners can veto the "resolved ballots." If more than 50% of the blocks of this period veto the ballot, all the decisions inside the ballot must be re-voted. When all the above-mentioned phases are concluded, the redemption phase starts, in which all the winning shares are given a price, and users can redeem them for CashCoins.

Crucial for the oracle's good outcome is the creator's role, which must choose an event whose outcome should be "salient" at a certain point in time, and voters that must correctly predict the answer of the majority of voters. If an event is not salient, voters will not be able to vote, leading to an unresolved market. Otherwise, if voters are incapable of coordinating, they will be slashed off their tokens. Failing to report or report outside the accepted value range results de facto in a slash. The number of tokens slashed is proportional to the distance of the reported value to the one identified as the true outcome.

Finally, the Truthcoin ecosystem can be broken if someone obtains 51% of the corporation. This means being able to control 51% of the system's economic value, which is unlikely to happen.

- DIFFICULTIES AND LIMITATIONS

When Sztorc had the idea of building a decentralized prediction market on Bitcoin, Reality Keys was a reliable oracle in development, but Sztorc was concerned that its system could have been manipulated for information of high value. Alternatives based on multisig were also not viable in the long run: "I was convinced that multisig was not the solution to the oracle problem. If the oracle problem is like sending a man to the moon, using a multisig is like trying to do it with a catapult." Emerging projects were also more oriented toward the idea of data feeds and, therefore, to a constant update of data to the blockchain. Sztorc's approach was, however, opposite: "We don't need a data feed, we don't need a frequent check... we only need to check if some information is true at a certain point."

For this reason, although in principle not interested in oracles, he developed his own Truthcoin by the end of 2013, publishing the first version of the whitepaper in early 2014. The Truthcoin whitepaper and the project itself were influenced by what was called the "blocksize war," a fierce dispute between Bitcoin developers on block size growth and the emergence of numerous alternatives to Bitcoin of dubious value. As the intention was to formalize the Truthcoin idea and then to have some other group do the actual development, Sztorc never launched the project. He could not take the risk of his ideas being manipulated by charlatans or of his project being erroneously labeled as a scam.

Therefore, the Truthcoin whitepaper was written in a highly scrupulous, detailed way so that debates such as the block-size war could never happen on his project. Furthermore, to alienate any association with alt/scam coins, he strictly adhered to Bitcoin and Nakamoto's ideas. In light of this, Truthcoin was planned to be developed as a Bitcoin sidechain.

In addition to being endorsed by Nakamoto, sidechains were also valuable because they allowed for the avoidance of using complex Bitcoin scripts. In the script design of prediction markets, "each bitcoin transaction would be like an enormous computer program." The script's length is due not only to the application's data but also to the information on how to digest that data. Although better programmable, the same limitations would have been encountered using an all-purpose blockchain such as Ethereum. A dedicated sidechain was then seen as a better solution since it already

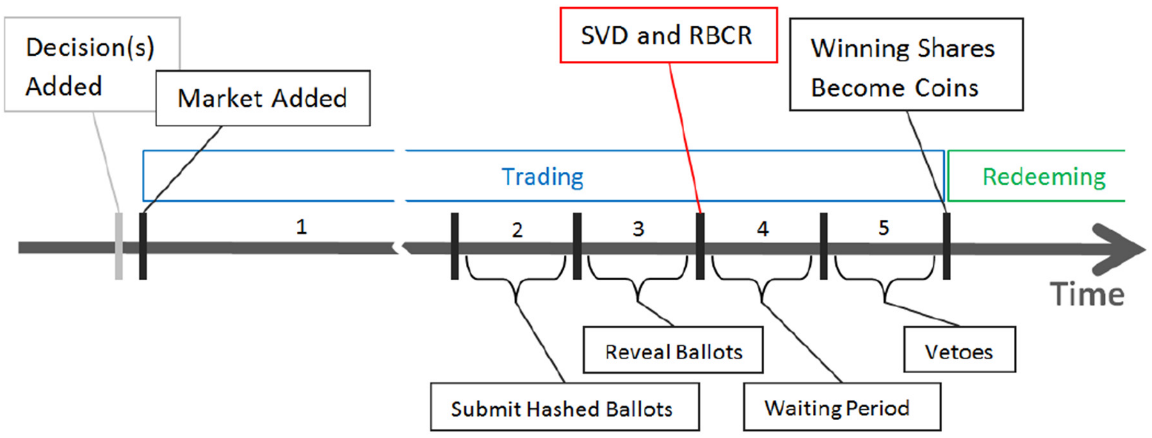


FIGURE 8. Truthcoin market resolution mechanism [54].

knows how to process the data and only needs minimal inputs to code each user action. All the required code is preloaded, and the full node already knows where to find and how to process the incoming data.

Although theorized and discussed in 2010, with Nakamoto's help, sidechains were still underdeveloped when Sztorc proposed Truthchoin [52]. The first practical idea of a two-way peg sidechain was discussed in December 2013 by Luke Dashjr. Along with others, they released the Blockstream whitepaper in October 2014, a system to enable blockchain innovation via pegged sidechains [56].

Sztorc started to develop a sidechain concept ("Drive Chain") in 2014. Still, being alone and aware of the work of Luke Dashjr, he decided to focus on other aspects of the Truthcoin project, hoping to use the Blockstream sidechain once completed. Therefore, inspired by Robin Hanson's work, he refined Truthcoin's logarithmic market scoring rules (LMSR) [57]. Concerning Hanson's contribution, Sztorc stated, "Each buy and sell happens unilaterally and atomically, so it was perfect for the blockchain." Apparently, LMSR also inspired what is now called automated market makers on Ethereum [58].

In 2015, the Truthcoin software was almost complete, but unfortunately, by that time, it was clear that the Blockstream sidechain project was not going to succeed in the short run. Therefore, Sztorc switched again to the development of a Bitcoin sidechain, "Drivechain" (spelled without a space this time), of which an advanced version was published in November 2015 so that it could have been beneficial not only for his now rebranded project Hivemind but also for the whole Bitcoin community. Due to the slow development of sidechains, the Truthcoin protocol is still under development.

IV. DISCUSSION

This section elaborates on the retrieved material to answer the research questions. The first part discusses the origin of the oracle idea, Nakamoto's view on extrinsic data on chains, and its influence on oracle development. The second part is concerned with the limitations of building oracles on Bitcoin and further elaborates on the passage to Ethereum and alt chains.

A. ORACLE ORIGIN AND NAKAMOTO'S VISION OF EXTRINSIC DATA

As per experts' experience, the oracle concept's first appearance came from the developer Mike Hearn, who had it formalized in a Bitcoin Wiki post. In his interview, Hearn stated that he was not inspired by the work of someone else, but he borrowed part of the idea directly from the computer science concept of the "random oracle." It also emerged that the name was meant to be provisional since oracles in computer science referred to something quite the opposite to what he wanted to elaborate on. Arguably, if an "oracle" is a black box that feeds a centralized machine with trusted data, Hearn's proposal of a transparent box that feeds a decentralized application with trustless data should have been addressed with a different name. However, the name is stuck to date, and the heterogeneity in blockchain oracles definitions found in [5] may also be due to this taxonomic overlap. Intuitively, suppose a developer is asked to write an "oracle" for a blockchain, and the principle of the white box is not explicitly explained. In that case, he will probably write the type of oracle learned from legacy computer science. Truthfully, both oracle types have the same finality but should work in a different way and under different logic.

Interestingly, it emerged that the word "smart contract" had also been improperly used, as Nakamoto named applications built on Bitcoin just "contracts." In its contract example, of a transaction that is executed as soon as enough conditions are met, no particular "smart" feature emerges.

According to Mike Hearn's reminiscences, the prefix "smart" started to be used a bit later—on the one hand—because it slightly resembled the concept of smart money or smart contract developed by Nick Szabo [59] and—on the other hand—it was seen as necessary to alienate the "legal aura" from the word contracts. Bitcoin contracts would probably have attracted unnecessary attention if erroneously perceived as related to legal contracts. However, similar to oracles, as smart contracts in origin referred to something else, their purpose and use could have been misinterpreted [59], [60], [61].

Concerning Nakamoto's idea of oracles, we cannot fully speculate on his opinion, given the limited number of available messages and posts certainly traceable to him. Thanks to an email from Mike Hearn on April 27, 2009, we can confirm that at some point, Nakamoto contemplated the idea of a third-party arbitrator to enable chargebacks on Bitcoin transactions [62]. In 2009, Bitcoin was at its earliest stage, and no scripts had yet been implemented. In that email, Nakamoto stated that if an agent required the possibility of a chargeback, an "escrow" transaction (which was still not implemented) should have been used. Therefore, a third party with the power to decide whether to return or release the money had to be designated. The idea was also to implement an expiration date to escrow for the funds to be automatically returned if no options were exerted within the time limit. Interestingly, the original Nakamoto proposal of escrow was slightly different. He proposed an escrow system in which Bitcoins were either released or burned. It was a sort of "kill switch" that prevented thieves from gaining benefits from cheating [63]. However, the community voted against the burning mechanism, opting for a chargeback mechanism instead. The kill switch was thought to penalize the buyer excessively (and also the whole community) by permanently removing Bitcoin from circulation [64].

Concerning extrinsic data, Nakamoto was arguably reluctant to have it on the Bitcoin mainchain. However, a marketplace, such as the one he proposed to launch on a sidechain, requires extrinsic data on products and feedback. Still, no explanation is given for how these data should have been securely fetched. It is arguable but not provable that Nakamoto was not planning any specific data transferring system for his sidechain, different from the traditional ones.

B. THE ON-CHAIN DATA DEBATE

Nakamoto's vision undoubtedly influenced early developers and enthusiasts. For Bitcoin core developers, extrinsic data injection into the blockchain was often seen as improper use of the ledger [43]. Following Nakamoto's idea, real-world applications utilizing extrinsic data should have been developed only through sidechains. This general mindset affected the development of oracles in many ways. Reality Keys was, in fact, developed entirely off-chain to avoid messing with Bitcoin. Truthcoin was created as a Bitcoin sidechain to adhere strictly to Nakamoto's ideas. Orisi transactions were considered non-standard and unlikely to be mined, although they were perfectly valid. Oraclize struggled to find developers due to skepticism about using Bitcoin scripts. Finally, Counterparty was dragged into the OP_Return war, which is also said to have impacted the Ethereum launch and development.

The OP_Return war is a matter that requires further elaboration. It was always "technically possible" to add data unrelated to Bitcoin transactions on the Bitcoin blockchain. Although achievable with other features, such as the one Nakamoto utilized to add the famous string "The Times 03/Jan/2009 Chancellor on brink of second bailout for banks" [65] on the genesis block, the OP_Return, was the easiest way to perform this operation [66], [67]. However, adding extrinsic data with OP_Return resulted in a transaction considered unusual or non-standard. As explained in the Orisi case, non-standard transactions are transactions that, despite being perfectly valid and minable, are not relayed by ordinary Bitcoin nodes and, therefore, are unlikely to be included in blocks. However, with a compliant miner (or by mining the transaction autonomously), it was possible to add any sort of data, such as hashes, pieces of articles, song lyrics, pieces of poetry, or pieces of whitepapers [43]. There are, in fact, online repositories, such as bitcoinstrings.com, that keep track of all this extrinsic data on Bitcoin. Fearing network bloat and discouraging widespread adoption of this practice, Bitcoin core developers, with the 2014 version v0.9.0., reduced the OP_RETURN payload size to 40 bytes (after a pre-release testing phase at 80 bytes), which made it practical for storing a hash plus some small metadata [68], [69]. Significant on-chain data were thought to negatively impact transaction fees and network performance. However, with this update, transactions with OP_Return of 40 bytes (or less) were considered standard and relayed by nodes with default settings.

This piece of Bitcoin history is exciting because, from this study, a discrepancy emerges in how the events are described and recalled by experts in the industry. According to the official Bitcoin Wiki, and reliable work of literature on the Bitcoin protocol, the OP_Return operator was inserted with version v0.9.0. and directly at 40 bytes [70], [71]. OP_Return at 80 bytes was described as an early hypothesis that was soon discarded and then accepted as an improvement in February 2015 with the v0.10.0 release. This view of history, however, clashes with some information found online and what some experts recall. The OP_Return operator, in fact, appears to already be part of the Bitcoin code developed by Nakamoto in 2009 [72]. As also discussed in official forums, it was leveraged as a "non-standard" feature long before 2014 [69].

Nonetheless, the main "trigger" of the OP_Return war appears to be an early release of v0.9.0, which de facto included a standardized OP_Return operator with a payload size of 80 bytes in the middle of 2013. According to what was declared by Bitcoin core developers, 80 bytes was a random value picked for testing purposes, and 40 bytes was then sought to be a fair amount to be finally included in the official release [69]. However, since the testing release was not widely announced and advertised to avoid overuse of the experimental features, other protocol developers building on Bitcoin in 2013 were unaware that the features they were using were meant to be "provisional." Therefore, when v0.9.0. was officially released with OP_Return payload at

40 bytes, they interpreted the ''slash'' as a deliberate censorship attempt, resulting in a fierce debate (therefore labeled as ''war'') within the community [43], [44].

Furthermore, from a technical point of view, the standardization of the OP_Return promoted with 2014 v0.9.0 was de facto inserted on Bitcoin in 2013 (for testing purposes) with pull request #2738 [73]. Therefore, for developers already leveraging OP_Return in 2013 as a standard feature, the 2014 standardization announcement was a ''lie,'' further fueling the harshness of the debate [74]. From a strictly technical perspective, the main change in OP_Return with the v0.9.0 2014 official release was just the halving of the payload size [75].

According to other views, however, the OP_Return debate was an exaggeration since they saw it just as an excuse for some people to promote their alt chains, blaming Bitcoin core developers for creating division in the community [43], [48].

Further details on the OP_Return debate are beyond the scope of this research. What clearly emerges is that the discussion of whether it is right to inject extrinsic data into Bitcoin is a conundrum of a difficult solution. If, on the one hand, Nakamoto's vision is evident in the fact that Bitcoin should remain pure (of data except for time), on the other hand, inventions in history are not always used as the inventor intended. In the case of Bitcoin, however, being decentralized and maintained by the community, any network ''misuse'' is paid by all the nodes regardless of their approval. As some objected, however, the exponential growth of the Bitcoin blockchain is also due to an increase in its use rather than just arbitrary data injection [74], [76].

Although still unsolved, nowadays, the debate is of less interest since real-world applications are mainly built on alt chains such as Ethereum. In the author's opinion, however, Nakamoto's idea to keep the mainchain pure and experiment on additional layers—alt chains or sidechains—could have been a reasonable, fair take at the end.

C. TRANSITION TO ETHEREUM

It is widely known that building on Bitcoin in the early days was a difficult task; therefore, oracle development was also problematic. The existence of applications, such as Satoshi Dice or Lighthouse, supports the view that it was actually possible to build on Bitcoin, but the development was not standardized, and every developer had to find the proper workaround for their application. Although Hearn developed Bitcoin-J as a wallet aimed at being like Metamask for Ethereum, it lacked the contributions of other developers to build further on top of it. The experience of Edmund Edgar with Runkeeper API also confirms that although it was possible to build with Bitcoin, the absence of a proper wallet and developing tools constituted a critical limitation. With developing tools and Metamask, EVM undoubtedly constituted an incentive for developers to migrate to Ethereum.

As discussed in the previous paragraph, concerns about net congestion and transaction costs were another element that further contributed to pushing real-world application development outside the Bitcoin domain. Despite the interest and prizes that Oraclize managed to obtain, he could never put the Bitcoin version into production due to low usage and the struggle to find developers. The Orisi project was abandoned due to skepticism about scripts and the inability to have their transactions mined. Their experience made it clear that no application could rely entirely on non-standard scripts for survival. Truthcoin was built following Nakamoto's advice, but to date, a working version is still unavailable due to the struggle to build a proper sidechain. Although not facing the issues of building directly on Bitcoin, it is suffering from the issue of not having an existing chain to be developed upon. In fact, other projects inspired by Truthcoin, such as Augur, could have been successfully launched on Ethereum in 2016 [77]. However, in complying with different standards, it is debatable whether that choice constitutes an improvement in the original design.

In the author's opinion, the actual limit of building on Bitcoin emerges through the experience of Counterparty. Regardless of whether or not they were misusing the OP_Return feature, their history shed light on the fact that building on Bitcoin was simply ''unwelcome'' [43]. The Ethereum gas system compromise allows anyone to program any type of application as long as the proper gas fees are paid. Ethereum was born with this system, and those who run nodes know their roles and purposes. Different philosophies and visions co-exist on Bitcoin; however, not all the nodes/miners share the same ideas. Although Bitcoin has a system of fees that varies according to net congestion and transaction type, it was not meant to run programs in the first place. Therefore, the payment of a fee is not necessarily a good compromise for those who wish for a light and mono-purpose chain. When Ethereum was launched, the environment was divided into two leading platforms, one of which was tormented by disputes on extrinsic data usage and block size and another one that was cheap and full of enthusiasts building and experimenting [78], [79]. Above all, much funding was also coming to the Ethereum platform, so it was understandable to expect a consistent migration of developers [80].

Nowadays, many improvements have been made to the Bitcoin network with the development of second layers, such as the Lightning network [81]. Ideally, they are capable of bringing the entire ecosystem built on Ethereum to the Bitcoin network. In addition, advancements have been made by Blockstream concerning sidechains. It is arguable, then, to expect a working version in the near future. Nonetheless, due to the shift from an electronic cash system to a safe-haven asset, as a consequence of the block-size war, many who own a significant amount of bitcoins share the philosophy of ''hold for dear life.'' Therefore, even if decentralized applications will eventually be built on Bitcoin, skepticism emerges about the existence of a solid user base willing to spend its precious assets on them.

V. CONCLUSION

This study provides an overview of the origin of blockchain oracles—from the first theoretical idea to the early practical applications on Bitcoin until the advent of Ethereum. In the absence of dedicated literature, experts who worked on oracles in the early days were interviewed, and the information provided was enriched with the available written material found online. From the research, it emerges that the idea of an oracle mechanism came from Mike Hearn, which was formalized on an early Bitcoin Wiki page. The concept was then further elaborated theoretically by other experts and then translated into actual software by a few enthusiasts. All these projects were developed in 2014. All approaches to solving the oracle problem bear peculiarities that are primarily due to the specific applications for which they were designed. Another aspect that emerges from this research is the difficulty of building oracles and, in general, applications on Bitcoin. According to experts' opinion, the main difficulties concerned the following:

- The absence of developing tools and wallets
- The large size and costs of Bitcoin scripts
- Concerns about net congestion.
- Skepticism in storing extrinsic data in Bitcoin

Interestingly, the hardest to overcome was not technical difficulties. A part of the Bitcoin community was, in fact, reluctant to introduce extrinsic data into the chain due to concerns about network growth/congestion and transaction fees. The same applies to some non-standard Bitcoin scripts. The passage to Ethereum was, therefore, inevitable.

The present research contributes to the academic literature filling the gap that exists from the origin of oracles on Bitcoin to modern oracles on Ethereum and alt chains. The original concepts of oracles, as well as smart contracts, are clarified. Therefore, the theoretical background of future academic papers could build on the findings of this research. Practitioners can also benefit from this research by understanding how oracles were theorized at early stages and how they were initially adapted to different applications.

The study also has limitations since, although data were double-checked and verified by the author, the history is described through the eyes of the experts interviewed. Therefore, it can be biased by their personal views and background. Furthermore, the impossibility of interviewing Nakamoto, and given the scarcity of retrieved material concerning his opinion/idea on oracles, the accuracy of the interpretation provided cannot be guaranteed.

Further studies can build on this by comparing the oracles analyzed in this paper with those developed afterward on Ethereum and other alt chains to outline their evolution in a broader timeframe.

ACKNOWLEDGMENT

The author would like to thank the anonymous reviewers and all the experts interviewed for their kind support and contributions to the research. Their cooperation in data collection was crucial to the successful outcome of the study. The author also wishes to thank all the colleagues who offered valuable feedback on the early versions of the paper.

REFERENCES

[1] G. Andresen. (Jun. 9, 2014). Bit-thereum | GavinTech. GavinTech. Accessed: Jan. 21, 2023. [Online]. Available: http://gavintech.blogspot.com/2014/06/bit-thereum.html

[2] G. Caldarelli, ''Understanding the blockchain Oracle problem: A call for action,'' *Information*, vol. 11, no. 11, p. 509, Oct. 2020, doi: 10.3390/info11110509.

[3] G. Caldarelli, *Blockchain Oracles and the Oracle Problem: A Practical Handbook to Discover the World of Blockchain, Smart Contracts, and Oracles—Exploring the Limits of Trust Decentralization*, 1st ed. Naples, Italy: Amazon, 2021.

[4] R. Belchior, A. Vasconcelos, S. Guerreiro, and M. Correia, ''A survey on blockchain interoperability: Past, present, and future trends,'' *ACM Comput. Surv.*, vol. 54, no. 8, pp. 1–41, Nov. 2022, doi: 10.1145/3471140.

[5] G. Caldarelli, ''Overview of blockchain Oracle research,'' *Future Internet*, vol. 14, no. 6, p. 175, Jun. 2022, doi: 10.3390/fi14060175.

[6] S. K. Ezzat, Y. N. M. Saleh, and A. A. Abdel-Hamid, ''Blockchain oracles: State-of-the-art and research directions,'' *IEEE Access*, vol. 10, pp. 67551–67572, 2022, doi: 10.1109/ACCESS.2022.3184726.

[7] S. Ibba, A. Pinna, G. Baralla, and M. Marchesi, ''ICOs overview: Should investors choose an ICO developed with the lean startup methodology?'' in *Proc. Int. Conf. Agile Softw. Develop.*, in Lecture Notes in Business Information Processing, vol. 314, 2018, pp. 293–308, doi: 10.1007/978-3-319-91602-6_21.

[8] S. Lahajnar and A. Rožanec, ''Initial coin offering (ICO) evaluation model,'' *Invest. Manag. Financial Innov.*, vol. 15, no. 4, pp. 169–182, 2018, doi: 10.21511/imfi.15(4).2018.14.

[9] B. Barraza, ''The worth of words: How technical white papers influence ICO blockchain funding,'' *MIS Quart. Executive*, vol. 18, no. 4, pp. 281–285, Dec. 2019, doi: 10.17705/2msqe.00021.

[10] H. Treiblmaier, ''The impact of the blockchain on the supply chain: A theory-based research framework and a call for action,'' *Supply Chain Manag., Int. J.*, vol. 23, no. 6, pp. 545–559, Sep. 2018, doi: 10.1108/SCM-01-2018-0029.

[11] A. Kumar, R. Liu, and Z. Shan, ''Is blockchain a silver bullet for supply chain management? Technical challenges and research opportunities,'' *Decis. Sci.*, vol. 51, no. 1, pp. 8–37, Feb. 2020, doi: 10.1111/deci.12396.

[12] A. Egberts, ''The Oracle problem—An analysis of how blockchain oracles undermine the advantages of decentralized ledger systems,'' *SSRN Electron. J.*, 2017. [Online]. Available: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3382343, doi: 10.2139/ssrn.3382343.

[13] J. Frankenreiter, ''The limits of smart contracts,'' *J. Inst. Theor. Econ.*, vol. 175, no. 1, pp. 149–162, 2019, doi: 10.1628/jite-2019-0021.

[14] M. Damjan, ''The interface between blockchain and the real world,'' *Ragion Prat.*, vol. 2018, no. 2, pp. 379–406, 2018, doi: 10.1415/91545.

[15] A. Pasdar, Z. Dong, and Y. C. Lee, ''Blockchain Oracle design patterns,'' Jun. 2021, *arXiv:2106.09349*.

[16] H. Al-Breiki, M. H. U. Rehman, K. Salah, and D. Svetinovic, ''Trustworthy blockchain oracles: Review, comparison, and open research challenges,'' *IEEE Access*, vol. 8, pp. 85675–85685, 2020, doi: 10.1109/ACCESS.2020.2992698.

[17] S. Eskandari, M. Salehi, W. C. Gu, and J. Clark, ''SoK: Oracles from the ground truth to market manipulation,'' *Proc. Under Rev.*, vol. 1, no. 1, Jun. 2021. [Online]. Available: https://dl.acm.org/doi/10.1145/3479722.3480994

[18] B. Liu, P. Szalachowski, and J. Zhou, ''A first look into DeFi oracles,'' in *Proc. IEEE Int. Conf. Decentralized Appl. Infrastruct. (DAPPS)*, Aug. 2021, pp. 39–48, doi: 10.1109/DAPPS52256.2021.00010.

[19] A. Pasdar, Y. C. Lee, and Z. Dong, ''Connect API with blockchain: A survey on blockchain Oracle implementation,'' *ACM Comput. Surv.*, vol. 55, no. 10, pp. 1–39, Oct. 2023, doi: 10.1145/3567582.

[20] A. Beniiche, ''A study of blockchain oracles,'' 2020, *arXiv:2004.07140*.

[21] G. Caldarelli and J. Ellul, ''The blockchain Oracle problem in decentralized finance—A multivocal approach,'' *Appl. Sci.*, vol. 11, no. 16, p. 7572, Aug. 2021, doi: 10.3390/app11167572.

[22] G. Caldarelli, "Wrapping trust for interoperability: A preliminary study of wrapped tokens," *Information*, vol. 13, no. 1, p. 6, Dec. 2021, doi: 10.3390/INFO13010006.

[23] Blockchain Oracle Summit. (Apr. 2022). *BOS Pre-Event Webinar: Oracles & Prediction Markets*. Accessed: Mar. 23, 2023. [Online]. Available: https://www.youtube.com/watch?v=GRNHwJNcmRM&t=210s

[24] (Jul. 9, 2019). *Early Temple | Smart Contracts for Next-Generation Business Models*. Accessed: Apr. 2, 2023. [Online]. Available: https://web.archive.org/web/20190709201247/http://earlytemple.com/

[25] S. Nakamoto. (Mar. 9, 2011). *Re: Open Sourced My Java SPV Impl | Satoshi's Archive*. Accessed: Nov. 25, 2022. [Online]. Available: https://www.bitcoin.com/satoshi-archive/emails/mike-hearn/13/

[26] S. Nakamoto. (Apr. 20, 2011). *Re: Holding Coins in an Unspendable State for a Rolling Time Window | Satoshi's Archive*. Accessed: Nov. 25, 2022. [Online]. Available: https://www.bitcoin.com/satoshi-archive/emails/mike-hearn/15/

[27] M. Hearn. (2011). Contracts. Bitcoin Wiki. Accessed: Dec. 2, 2022. [Online]. Available: https://en.bitcoin.it/w/index.php?title=Contract&oldid=13637

[28] Furunodo and Bitcoin Wiki. (2020). *Contracts*. Accessed: Dec. 12, 2022. [Online]. Available: https://en.bitcoin.it/w/index.php?title=Contract&oldid=67871

[29] J. Southurst. (Feb. 6, 2014). *Apple Removes Blockchain Bitcoin Wallet Apps From Its App Stores*. Accessed: Nov. 21, 2022. [Online]. Available: https://www.coindesk.com/markets/2014/02/06/apple-removes-blockchain-bitcoin-wallet-apps-from-its-app-stores/

[30] Etherscan. (2023). *Oraclize Address Transactions*. Accessed: Jan. 30, 2023. [Online]. Available: https://etherscan.io/address/0x26588a9301b0428d95e6fc3a5024fce8bec12d51#analytics

[31] D-Nice. (May 12, 2017). *GitHub—Provable-Things/Oraclize-Lib: Oraclize Node.js Library*. Accessed: Jan. 27, 2023. [Online]. Available: https://github.com/provable-things/oraclize-lib

[32] J. Warren. (2012). *Bitmessage: A Peer-to-Peer Message Authentication and Delivery System*. Accessed: Nov. 24, 2022. [Online]. Available: https://www.Bitmessage.org

[33] T. Kolinko, G. Pstrucha, and K. Kucharski. (2014). *Orisi Whitepaper*. Accessed: Sep. 6, 2022. [Online]. Available: https://github.com/orisi/wiki/wiki/Orisi-White-Paper

[34] S. Bistarelli, I. Mercanti, and F. Santini, "An analysis of non-standard transactions," *Frontiers Blockchain*, vol. 2, p. 7, Aug. 2019, doi: 10.3389/FBLOC.2019.00007.

[35] T. C. Schelling. (1960). *The Strategy of Conflict. Public Domain, Google-Digitized*. Accessed: Feb. 13, 2023. [Online]. Available: https://www.hup.harvard.edu/catalog.php?isbn=9780674840317

[36] A. Piscitello. (Jul. 23, 2013). *Implementing External State Contracts—Feedback Requested*. Accessed: Jan. 3, 2023. [Online]. Available: https://bitcointalk.org/index.php?topic=260898.0

[37] J. Southurst. (2014). *Reality Keys: Bitcoin's Third-Party Guarantor for Contracts and Deals*. Accessed: Jan. 6, 2023. [Online]. Available: https://www.coindesk.com/markets/2014/01/17/reality-keys-bitcoins-third-party-guarantor-for-contracts-and-deals/

[38] E. Edgar and C. Delrey. (2014). *Realitykeysdemo.py*. Accessed: Jan. 6, 2023. [Online]. Available: https://github.com/edmundedgar/realitykeys-examples/blob/master/realitykeysdemo.py

[39] K. Pani. (2014). *How to Create the Meta Chain*. Accessed: Jan. 16, 2023. [Online]. Available: https://blogs.sap.com/2014/01/06/how-to-create-the-meta-chain/

[40] D. Weller. (2016). *Decoding a Transaction*. Accessed: Jan. 24, 2023. [Online]. Available: https://github.com/tokenly/counterparty-spec/blob/master/spec/02-decoding.md

[41] D. Weller, I. Zuber, and Chiguiretor. (2019). *Protocol Specification | Counterparty*. Accessed: Jan. 25, 2023. [Online]. Available: https://github.com/CounterpartyXCP/Documentation/blob/master/Developers/protocol_specification.md

[42] A. Krellenstein, E. Wagner, and R. Dermody. (Mar. 21, 2014). *[ANN][XCP] Counterparty—Pioneering Peer-to-Peer Finance—Official Thread*. Accessed: Jan. 15, 2023. [Online]. Available: https://bitcointalk.org/index.php?topic=395761.msg5817170#msg5817170

[43] BitMex-Research. (2022). *The OP_Return Wars of 2014—Dapps vs Bitcoin Transactions*. Accessed: Jan. 12, 2023. [Online]. Available: https://blog.bitmex.com/dapps-or-only-bitcoin-transactions-the-2014-debate/

[44] BCH World Order. (2018). *A Few Months After the Counterparty Developers Started Using OP_RETURN, Bitcoin Developers Decreased the Size of OP_RETURN From 80 Bytes to 40 Bytes. The Sudden Decrease in the Size of the OP_RETURN Function Stopped Networks Launched on Top of Bitcoin From Operating Properly*. Accessed Jan. 19, 2023. [Online]. Available: https://www.reddit.com/r/btc/comments/80ycim/a_few_months_after_the_counterparty_developers/

[45] D. Bradbury. (2014). *Developers Battle Over Bitcoin Block Chain*. Accessed: Jan. 19, 2023. [Online]. Available: https://www.coindesk.com/markets/2014/03/25/developers-battle-over-bitcoin-block-chain/

[46] Historian1111. (2015). *Blockstream Co-Founder Luke-jr Banning Mastercoin and Counterparty Transactions, Adding Blacklists to Gentoo Bitcoind by Default*. Accessed: Jan. 17, 2023. [Online]. Available: https://www.reddit.com/r/Bitcoin/comments/2pfxak/blockstream_cofounder_lukejr_banning_mastercoin/

[47] Insette. (2018). *Your Best Pitch for Decred*. Accessed: Jan. 19, 2023. [Online]. Available: https://old.reddit.com/r/decred/comments/6wxueo/your_best_pitch_for_decred/dmcer4d/

[48] M. I. Moneyist. (2019). *The OP_Return War 'Debunked'*. Accessed: Jan. 14, 2023. [Online]. Available: https://twitter.com/notgrubles/status/1187470076833697794

[49] V. Buterin. (Nov. 12, 2017). *The Very Earliest Versions of ETH Protocol*. [Online]. Available: https://twitter.com

[50] E. Muratov. (2016). *Bitcoin Minimalism: Counterparty to Talk With Bitcoin in Ethereish*. Accessed: Jan. 20, 2023. [Online]. Available: https://web.archive.org/web/20170623073803/http:/forklog.net/bitcoin-minimalism-counterparty-to-talk-with-bitcoin-in-ethereish/

[51] S. Nakamoto. (Mar. 9, 2011). *Re: 2 Open Sourced My Java SPV Impl | Satoshi's Archive*. Accessed: Nov. 25, 2022. [Online]. Available: https://www.bitcoin.com/satoshi-archive/emails/mike-hearn/14/#selection-25.4239-25.4589

[52] Appamatto. (Nov. 15, 2010). *BitDNS and Generalizing Bitcoin*. Accessed: Feb. 20, 2023. [Online]. Available: https://bitcointalk.org/index.php?topic=1790.0

[53] C. Isidore. (Mar. 11, 2013). Intrade shut down due to financial probe. CNN Business. Accessed: Dec. 1, 2022. [Online]. Available: https://money.cnn.com/2013/03/11/investing/intrade-shutdown/index.html

[54] P. Sztorc. (Dec. 14, 2015). *Truthcoin Peer-to-Peer Oracle System and Prediction Marketplace*. Accessed: Feb. 15, 2023. [Online]. Available: https://bitcoinhivemind.com/papers/truthcoin-whitepaper.pdf

[55] *Salience Noun—Definition, Pictures, Pronunciation and Usage Notes | Oxford Advanced Learner's Dictionary*. Accessed: Feb. 15, 2023. [Online]. Available: https://www.oxfordlearnersdictionaries.com/definition/english/salience

[56] A. Back et al., "Enabling blockchain innovations with pegged sidechains," Tech. Rep., 2014. [Online]. Available: https://blockstream.com/sidechains.pdf

[57] R. Hanson, "Logarithmic markets coring rules for modular combinatorial information aggregation," *J. Predict. Markets*, vol. 1, no. 1, pp. 3–15, Dec. 2012, doi: 10.5750/jpm.v1i1.417.

[58] G. Angeris and T. Chitra, "Improved price Oracles: Constant function market makers," *SSRN Electron. J.*, pp. 80–91, 2020. [Online]. Available: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3636514, doi: 10.2139/ssrn.3636514.

[59] N. Szabo. (1994). *Smart contracts, Personal Blog*. Accessed: Feb. 8, 2023. [Online]. Available: https://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/smart.contracts.html

[60] S. Nick. (1997). *Formalizing and Securing Relationships on Public Networks*. Accessed: Feb. 15, 2020. [Online]. Available: https://journals.uic.edu/ojs/index.php/fm/article/view/548

[61] A. M. Antonopoulos and G. Woods, *Mastering Ethereum: Building Smart Contracts and DAPPS*. Sebastopol, CA, USA: O'Reilly Media, 2018.

[62] S. Nakamoto. (Apr. 27, 2009). *Re: Lack of Chargeback Support | Satoshi's Archive, Nakamoto Email*. Accessed: Nov. 25, 2022. [Online]. Available: https://www.bitcoin.com/satoshi-archive/emails/mike-hearn/8/

[63] S. Nakamoto. (Aug. 7, 2010). *Escrow*. Accessed: Feb. 20, 2023. [Online]. Available: https://bitcointalk.org/index.php?topic=750.0

[64] Sebastian. (Mar. 2011). *Bitcoin Secure Chargebacks (With Votes)?* Accessed: Feb. 20, 2023. [Online]. Available: https://bitcointalk.org/index.php?topic=4856.0

[65] C. Murray. (Dec. 7, 2021). *The Mystery of the Genesis Block—CoinGeek*. Accessed: Feb. 7, 2023. [Online]. Available: https://coingeek.com/the-mystery-of-the-genesis-block/

[66] N. O'Dell. (Dec. 29, 2016). *Op Return—What Was the Very Initial Value of OP_RETURN?—Bitcoin Stack Exchange*. Accessed: Feb. 6, 2023. [Online]. Available: https://bitcoin.stackexchange.com/questions/50414/what-was-the-very-initial-value-of-op-return

[67] M. Bartoletti and L. Pompianu, "An analysis of Bitcoin OP_RETURN metadata," in *Proc. Int. Conf. Financial Cryptogr. Data Secur.*, in Lecture Notes in Computer Science: Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics, vol. 10323, 2017, pp. 218–230, doi: 10.1007/978-3-319-70278-0_14.

[68] Seandotau. (2016). *OP_RETURN 40 TO 80 BYTES*. Accessed: Jan. 23, 2023. [Online]. Available: https://www.talkcrypto.org/blog/2016/12/30/op_return-40-to-80-bytes/

[69] G. Andresen. (2013). *Relay OP_RETURN Data TxOut as Standard Transaction Type*. Accessed: Jan. 23, 2023. [Online]. Available: https://github.com/bitcoin/bitcoin/pull/2738

[70] A. M. Antonopoulos, *Mastering Bitcoin: Programming the Open Blockchain*, 2nd ed. Sebastopol, CA, USA: O'Reilly Media, 2017.

[71] Furunodo. (2020). *OP_RETURN—Bitcoin Wiki*. Accessed: Feb. 7, 2023. [Online]. Available: https://en.bitcoin.it/wiki/OP_RETURN

[72] (Aug. 30, 2009). *2009 Bitcoin Code*. Accessed: Feb. 7, 2023. [Online]. Available: https://github.com/bitcoin/bitcoin/blob/4405b78d6059e536c36974088a8ed4d9f0f29898/script.cpp#L170

[73] G. Andresen. (Oct. 24, 2013). *Core Development Update #5*. Accessed: Feb. 9, 2023. [Online]. Available: https://web.archive.org/web/20131024212741/https://bitcoinfoundation.org/blog/?p=290

[74] T. Swanson, "Bitcoin hurdles: The public goods costs of securing a decentralized seigniorage network which incentivizes alternatives and centralization," *SSRN Electron. J.*, pp. 1–52, 2014.

[75] G. Andresen. (Feb. 26, 2014). *Script: Reduce OP_RETURN Standard Relay Bytes to 40 by Jgarzik · Pull Request #3737 · Bitcoin/Bitcoin · GitHub*. Accessed: Feb. 10, 2023. [Online]. Available: https://github.com/bitcoin/bitcoin/pull/3737/files

[76] E. Strehle and F. Steinmetz, "Dominating OP returns: The impact of Omni and veriblock on Bitcoin," *J. Grid Comput.*, vol. 18, no. 4, pp. 575–592, Dec. 2020, doi: 10.1007/S10723-020-09537-9.

[77] J. Peterson, J. Krug, M. Zoltu, A. K. Williams, and S. Alexander, "Augur: A decentralized Oracle and prediction market platform," Jan. 2015, *arXiv:1501.01042*.

[78] C. Russo. (Jul. 11, 2020). *Sale of the Century: The Inside Story of Ethereum's 2014 Premine—CoinDesk*. Accessed: Feb. 12, 2023. [Online]. Available: https://www.coindesk.com/markets/2020/07/11/sale-of-the-century-the-inside-story-of-ethereums-2014-premine/

[79] V. Buterin. (Jul. 22, 2014). *Launching the Ether Sale | Ethereum Foundation Blog*. [Online]. Available: https://blog.ethereum.org/2014/07/22/launching-the-ether-sale Accessed: Feb. 12, 2023.

[80] Cryptopedia Staff. (Mar. 10, 2022). *Initial Coin Offerings: The Ethereum ICO Boom | Gemini*. Accessed: Feb. 12, 2023. [Online]. Available: https://www.gemini.com/cryptopedia/initial-coin-offering-explained-ethereum-ico

[81] J. Poon and T. Dryja. (2016). *The Bitcoin Lightning Network: Scalable Off-Chain Instant Payments*. Accessed: Feb. 12, 2023. [Online]. Available: https://lightning.network/lightning-network-paper.pdf

[82] L. Gudgeon, D. Perez, D. Harz, B. Livshits, and A. Gervais, "The decentralized financial crisis," in *Proc. Crypto Valley Conf. Blockchain Technol. (CVCBT)*, Piscataway, NJ, USA, Jun. 2020, pp. 1–15, doi: 10.1109/CVCBT50464.2020.00005.

[83] E. Edgar. (Jan. 20, 2014). *[ANN] Reality Keys: An Oracle Letting You Use External State in Transactions*. Accessed: Apr. 7, 2023. [Online]. Available: https://bitcointalk.org/index.php?topic=423638

[84] Pacyrus. (2021). *Multi-Signature—Bitcoin Wiki*. Accessed: Apr. 7, 2023. [Online]. Available: https://en.bitcoin.it/wiki/Multi-signature

[85] BitcoinWiki. *OP_RETURN—Bitcoin Wiki*. Accessed: Apr. 7, 2023. [Online]. Available: https://en.bitcoin.it/wiki/OP_RETURN

[86] S. Nakamoto. (2008). *Bitcoin: A Peer-to-Peer Electronic Cash System*. Accessed: Jun. 11, 2019. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[87] J. Willett. (2013). *The Second Bitcoin Whitepaper*. [Online]. Available: https://github.com/bitsblocks/mastercoin-whitepaper/blob/master/index.md

[88] Redleader556. (2015). *Exposed: Luke-jr Plans on Forcing Blacklists on All Gentoo Bitcoin Users by Default, for the Second Time*. Accessed: Jan. 17, 2023. [Online]. Available: https://www.reddit.com/r/Bitcoin/comments/2pfgjg/exposed_lukejr_plans_on_forcing_blacklists_on_all/

GIULIO CALDARELLI received the M.S. degree in management and governance from the University of Siena, Italy, in 2013, and the Ph.D. degree in economics and management from the University of Verona, Italy, in 2022.

He is currently a Researcher with the University of Turin, Italy, and a Lecturer of financial accounting and decentralized finance. He is investigating oracular mechanisms and issues related to the use of oracles in real-world blockchains. Most of his work concerns sustainable supply chains and decentralized finance. He recently published a book titled *Blockchain Oracles and the Oracle Problem*.

Dr. Caldarelli was the Chairperson for the first Blockchain Oracle Summit, in 2022. Within the IEEE, he presented a paper on the Oracle problem at IEEE ICTMOD 2020 and served as the Session Chair for the IEEE Cybermatics 2020.

Open Access funding provided by 'Università degli Studi di Torino' within the CRUI CARE Agreement