July ‘23

dOra, Distributed Oracle technology whitepaper

Bring the blockchain robustness to enterprise applications

STEFANO DELLA VALLE – TELECONSYS

VERSION 1.0

DRAFT

Summary

1 Introduction

Blockchain has demonstrated for the first time in the history of information technology that it is possible to create an application that performs critical processing with zero operational costs.

Any computing platform (ERP, CRM, IoT, etc.) has marginal direct operating costs, proportional to the usage level, but high indirect operating costs, proportional to the relevance of the supporting processes. These indirect costs come from the need to manage the platform, where "manage" means ensure that the platform is secure and reliable. Platform management requires the use of numerous accessory technologies to monitor critical parameters, security events, etc., and requires a team of expert operators who can react promptly to any critical event by applying corrective actions.

Obviously, blockchains have an operational cost as well, but this marginally impacts users because the cost derived from the processes that produce the high security of the network is covered by ad hoc created rewards.

None of the organizations that support the development of the best-known blockchains, such as Ethereum Foundation, IOTA Foundation, Algorand Foundation, etc., spends a single euro to protect network nodes with firewalls and VPNs or adopt verification procedures aimed at detecting attempts to breach nodes.

This is not simply because foundations do not control the network's nodes, but mainly because if someone decided to attack one or more nodes, they would get no benefit.

The indirect operational costs of a blockchain are zero because there is no need for a team of system administrators, security managers, and auditors, no need for a NOC, and no reason to activate a SOC. This is even more evident when one considers that Bitcoin, the network that contains the largest capital of crypto finances, does not even have a foundation that governs the project.

1.1 No blockchain for enterprise applications

The robustness of blockchain has motivated many teams to try to use it to bring these advantages to industrial and enterprise applications, or alternatively, to bring industrial and enterprise applications into the blockchain space.

Both approaches have failed, and the reason is simple: the robustness of blockchains comes from rewards obtained from validator nodes that justifies their direct operational costs and thus the activation of many nodes which concretely produces the resilience of the distributed ledger to many types of attacks.

This is a game changer: obtaining the cooperation of many in securing a common asset without direct remuneration has never happened before.

Unfortunately, enterprise data is not a common asset and indeed often must be covered by a high degree of confidentiality even though it is archived in a public container.

Moreover, voluntary participation to the blockchain network is justified by an indirect return on investment in the mining's systems and in the increased token price: the more powerful the node

that produces a block, the greater the probability of getting a reward that largely covers costs and the more secure is the network and more people will invest in the token, moving its price.

Bringing this organizational model into a traditional application architecture is extraordinarily complex if not impossible, and using a public blockchain is inconvenient due to the impossibility of obtaining service level agreements. This comes again from the fact that the leger is a shared common resource that of course can support unlimited operation requests, so it provides a best effort service.

The attempt to use private or dedicated blockchains for an application is also unsuccessful, because obviously in a private blockchain there is a predetermined authority and thus the decentralized nature of the blockchain is lost and with it the security it can guarantee, including the risk of the platform operator acts maliciously.

1.2 Authoritativeness and anonymity

Blockchain uses authoritativeness of nodes to create common consensus on the validity of the ledger state. Authoritativeness is an abstract concept: it cannot be acquired but is awarded by the community.

In mining, authoritativeness is awarded to nodes that demonstrate to control the most hash power; in stacking, it is awarded to nodes that allocate guarantee capital.

So, in theory, it is possible to create a secure blockchain by creating a new consensus model that rewards the authority of its validators, with methods other than mining and stacking.

Unfortunately, in practice it is not, because anonymity and authoritativeness are two conflicting characteristics: how can a subject be authoritative and anonymous at the same time?

This is achieved in mining and stacking because the authoritativeness of the node is not persistent. In mining it is implicitly proved, while in stacking it is verified every time the node publishes a new block. The prior behavior of the validator nodes is irrelevant.

This is one of the design choices that enabled the creation of blockchain, because anonymity is a functional requirement for a public ledger that manage funds: if anyone can analyze the contents of the ledger and detect addresses containing enormous amounts of assets, then, for obvious security reasons, the owners of these addresses must be anonymous.

This design choice, however, gives rise to some limitations of blockchains: a block cannot be confirmed by a single validator because the single node may be malicious. Therefore, to obtain final transaction confirmation, that is, that a transaction is not later removed from the ledger because it is included in an invalid block, users must wait for numerous partial confirmations produced in succession by different validators.

In summary, anonymity is a strong limitation of the consensus-building system in a blockchain and is even more limiting for enterprise applications, because the validity of application data cannot be tested by a node that does not know the application workflow.

The only solution to validate business data is to have validator nodes that know how to interpret them and are recognized by the community as authoritative validators.

Therefore, enterprise data validators must be trusted, i.e., provided with an inviolable identity and have a tracklog of successful validations, characteristics that allow us to have a high degree of confidence that their output is reliable.

Joining many opinions of trustable enterprise validator nodes allows to create strong consensus on any kind of data.

1.3 Exchanges, oracles and bridges

Because blockchains are remarkably successful while not supporting enterprise applications, someone came up with the idea of creating "oracles," moving (in part) the problem of validating enterprise data outside the blockchain's security domain.

Oracles, in honor of the famous Oracle of Delphi, are applications that provide data to smartcontracts by inscribing them in the ledger, thus making it possible to use smartcontracts to perform sophisticated analyses on application data. Bridges are oracles that move data from between two blockchains. Exchanges are platforms where clients can connect to swap tokens (and eventually data).

Most exchange transactions are completed off-chain, that is, supported by a cloud platform that in most cases is managed by a corporation, so it is not a decentralized system.

The problem with oracles and bridges is that, very often, they are platforms whose authority comes from the team that runs them. They are often small teams, often the operational processes are not transparent, and in general people can make mistakes even unintentionally.

Moreover, the use of the consensus produced by the network on the status of the smartcontract provides no guarantee at all that the data produced by the smartcontract is reliable.

In fact, in addition to depending on the inputs produced by the oracles, the output of a smartcontract depends on the algorithm for processing the input data, and this algorithm is controlled by someone.

2 The idea behind dOra

Our goal is to design a decentralized computing system that can run any algorithm on any type of data and be free of single points of control and failure.

In short, instead of creating consensus on a finite state machine's state, as is done in a blockchain, dOra creates consensus on the proper execution of a sequence of tasks by a committee of nodes, no matter which is the tasks.

As in a blockchain, the sequence of operations can be executed incorrectly only if the majority of nodes (or a specified percentage of nodes) were operating maliciously.

This approach has many advantages. The most obvious is that if it is impossible to violate committee security, then the data it produces is implicitly correct. As a result, there is no need for a sequential validation process, and thus the data produced is immediately usable for further processing or as input to operational processes.

In practice, the correctness of the data produced by the committee depends solely on the task sequence designer.

This is exactly what the classical smartcontract model does, where the logical correctness of the smartcontract output is the responsibility of the developer who wrote it, not the blockchain nodes that execute it.

So dOra produces the same effects as classical smartcontract, but without suffering its limitations.

Nodes that make up the committee do not compete because they will share the reward given to the entire committee by the users.

This could lead to the creation of node cartels that could be persuaded to operate maliciously, compromising the quality of data produced by the committee.

To avoid this risk, each node is provided with its own identity. The identity allows the node to be assigned a degree of authoritativeness to which, for instance, a proportional share of the reward might correspond.

The node's authoritativeness can be derived from various indicators, such as the track log of successfully executed tasks by committees in which the node has actively and correctly contributed.

The authority of the node is therefore verifiable by anyone, which allows the number of nodes that make up the committee to be kept to very small quantities.

In fact, the security of the committee is proportional to the number of nodes, and it can be intuitively understood, that just as for Bitcoin it is statistically unlikely that someone would attempt an attack by linking a block to one with more than 6 confirmations, so it is statistically unlikely that all 4 out of 7 nodes in a committee would be corrupted.

In addition, if necessary, the client can request a higher minimum participation threshold, such as 6 out of 7 or even 7 out of 7.

So, we can say that:

- A committee of 3 nodes can perform non-critical operations.
- A committee of 5 nodes is down sufficiently robust to handle medium-critical tasks.
- A committee composed of 7, 9 or more nodes is virtually inviolable.

If nodes, instead of competing in speed, cooperate, then they can produce consensus on data acquired from external sources and processed with an algorithm provided by the client.

3 The dOra building blocks

Having always worked on extremely critical business projects and having in-depth knowledge of how blockchain works, Teleconsys' DLT competence center team designed dOra, which stands for Distributed Oracle, a distributed computing, validation, and archiving system.

dOra answers these questions:

- Who processes an algorithm in a decentralized way?
- Who certifies that it was executed correctly?
- Who stores the data produced?
- Who publishes it with assurance that it is the correct one?

In addition, dOra provides a solution to a problem common to every cloud-based application, i.e., the risk of being the target of attacks simply because the application's IP address or URL must be public.

The dOra system consists of a committee of nodes that execute scripts containing a list of tasks.

The architecture of dOra nodes is similar to that of blockchain nodes, but the building blocks are designed very differently to avoid any performance limitations and security risks previously discussed.

The architecture of the dOra node is very similar to that of a blockchain node. Each building block may be implemented differently but it exists in every blockchain because it plays a critical role in the security of the ledger and the operation of the network.

Below we analyze the characteristics of each block in the classical blockchain model, and then highlight the different implementation in the dOra system.

3.1 Identity block

3.1.1 Node Identity in Blockchain

In blockchains node identification is not a requirement. For instance, when a node receives a reward for creating a valid block, it is assigned to an address specified by the node manager that is not necessarily representative of the node identity or even that of its manager.

Distributed ledger addresses are typically used as pseudo-anonymous identifiers of blockchain users and therefore could be used to represent various logical subjects of an application context. However, blockchains do not use any specific standards for creating Digital Identities, so this option must be implemented at the application level.

3.1.2 Node Identity in dOra

It would have been simple enough to define that a dOra node must have a Digital Identity issued by a Certificate Authority, however, the responsibility of a distributed computational system that manages business information goes far beyond ensuring the identity of its operational components.

We want to ensure that no one can tamper with data collected by third parties or controlled equipment and that the same equipment cannot be replaced with others that impersonate them.

It is essential that data sent from the system to a recipient be safeguarded from unauthorized observers and therefore recipients of information must also be identified with certainty.

To solve this problem in dOra every entity (data source, data destination, validator, processors, etc.) is provided with a digital identity.

However, commonly used digital identities based on Certification Authorities are a risk factor as well: by attacking the Certificate Authority, certificates of important application entities can be deleted, shutting down the entire application platform.

To solve this issue in dOra we use a new Digital Identity standard, defined by W3C, named “Distributed Identity” or DID.

Among many interesting features this standard guarantees:

- Each subject can create one or more of its own identities.
- Each identity has a DID Document that contains methods and public keys that allow to interact with the subject for various purposes, including verifying that it holds the private key that controls a method or the DID document itself.
- An identity can be assigned to humans, digital and logical subjects.
- An identity can be assigned verifiable credentials by a second identity that gives it rights and powers. This tool makes it possible to eliminate operational directories, the critical security point of any enterprise system.

But more importantly, the DID standard defines that DID documents do not have to be published by an authority but can be published by the subject they represent on a distributed registry.

This standard eliminates any type of single point of failure and control in the Identity application space.

3.2 The communication block

3.2.1 The blockchain communication block

Blockchain are normally designed to use TCP/IP as Transport Layer.

In most cases, nodes communicate with a TCP based protocol, called "gossip protocol", that allows information to be propagated progressively to nodes throughout the network.

The topology by which nodes are connected is either statically or dynamically constructed through protocols to discover the IP addresses of public nodes to which each new node can connect.

TCP/IP is a reliable and efficient protocol, but the need to publish its network address exposes the node to potential attacks aimed at putting it off-line.

3.2.2 The Dora communication block

dOra nodes have more complexity in the communication block than a classic blockchain node. This is partly intended to increase the node security and reduce the operational costs required to protect them, and partly due to the need for nodes to communicate with the real world.

The dOra communication block has two distinct subcomponents. One realizes a secure transport layer that can be used by nodes for their interactions and by advanced applications to exchange data reliably. The other ensures interoperability with applications that are only able to communicate via REST API.

3.2.2.1 The dOra secure transport layer

A system that must handle enterprise data needs a reliable transport system suitable for handling several use cases, from data produced by Web servers to data produced by IoT devices.

TCP/IP is the most widely used protocol for this purpose, but not always. In many applications HTTPS is preferred, specifically to create encrypted peer-to-peer channels. To transmit data produced by connected devices (IoT), the MQTT protocol is often preferred, which builds on TCP/IP, extending its functionality.

MQTT delivers two services not provided by TCP/IP and HTTP:

- sending a single piece of data to multiple recipients
- Creating an "unconnected communication" ensures the transfer of data from multiple senders to the same recipient while preventing it from being overloaded.

In general, these services can be useful for many use cases, but MQTT is not suitable for supporting applications that require sending large volumes of data.

TCP/IP and HTTP are reliable because they implement the communication mode called "connected" where the two parties constantly confirm to each other that messages have been received correctly and are retransmitted if needed. This mode requires both peers to be active at the same time, which, in some use cases, is not convenient.

Blockchains were invented exactly to transfer data (representing value) between two parties, with no direct interaction.

So, the idea is to use a blockchain as dOra application data transport layer, gaining many advantages over TCP/IP and MQTT:

- persistence in the data in the ledger, thus connectionless communication.
- Maximum reliability because the data is replicated on all blockchain nodes.
- Unlimited number of receivers of the same data.

Unfortunately, blockchains do not provide service level agreements on costs and waiting times to permanently store data in the ledger. Both these parameters are impacted by network congestion level.

This feature makes it unfeasible to use a blockchain as a transport layer, but there is a distributed ledger designed differently: IOTA.

3.2.2.1.1 IOTA, a blockchain with predictable performance

IOTA has been chosen because it will ensure predictable performance and cost of writing data into the ledger. This feature derives from the network functional model in which each node can create and publish blocks autonomously without needing to be perfectly synchronized with all others to create a single long chain of blocks.

Instead, IOTA node creates a Direct Acyclic Graph of blocks, called Tangle, where each block is linked to more previous blocks. This allow nodes to freely choose where to link a new block and immediately publish it to the network.

Like regular blockchains, IOTA's consensus is also based on the authoritativeness of nodes. However, in IOTA[1] the authoritativeness evaluation is not based on node's computing power (mining) or from capital placed as a guarantee (stacking). It derives from the actual contribution each node provides to the security of the ledger.

To contribute to the security of the ledger, each node simply creates valid blocks. In short, a block is valid only if it contains a valid transaction and is cryptographically linked to multiple previous valid

[1] IOTA 2.0 https://www.iota.org/foundation/research-papers

blocks. Each valid block validates direct and indirect connected previous blocks and when the sequence of validations reaches a required length, the block is definitively considered confirmed.

As a side effect of this consensus model, in IOTA the greater the number of active nodes creating blocks, the faster the confirmation of blocks and the more secure the network becomes.

Of course, if every node can create blocks, the network can become congested very quickly.

To avoid this potential issue and democratically distribute the power of creating blocks, IOTA defines that the maximum block creation frequency allowed to each node is proportional to its authoritativeness.

One of the methods used to award authoritativeness to each node is the number of tokens transferred by the transaction contained in a valid block it publishes. Authoritativeness is consumed for each new block published in proportion to the moved tokens, then is gained over time, reach a maximum value proportional to the moved tokens, and then reduces over time. This mechanism ensures that the more authoritativeness a node owns, the higher is the block creation frequency it can leverage.

This is why IOTA grants predictable performance: an application dedicated node will grant performance proportionally to its authoritativeness, not the network usage level.

This system is a major step forward in building consensus on the status of the ledger in an efficient way that is only partly related to the amount of assets controlled and not related to energy consumption.

Thanks to this consensus model, nodes do not need to obtain rewards and fees.

This is why IOTA guarantees predictable costs[2], because a node dedicated to an application, will not incur additional costs beyond those derived from its own activity

3.2.2.1.2 How to use IOTA as Secure Transport Layer

Differently from other blockchain, IOTA can manage many kinds of blocks. One of them is the "tagged block" where block space (up to 8 KB) can be filled with arbitrary data and the tag included in the block header, can be compiled with a 32 byte of arbitrary data. A client can ask the node to provide blocks with a specific tag.

We use tagged blocks to publish data on the IOTA network.

The label value is set with the DID of the recipient who can easily find them in the ledger.

[2] In IOTA 2.0, to gain authoritativeness a node must control an address and move contained tokens creating valid blocks. This is not a cost, but a capital immobilization similar to the stacking.

This addressing method, besides being very efficient, supports the possibility of creating one-to-many communication channels, either by creating arbitrary tags or by using DIDs representing groups of recipients.

This communication method perfectly overcomes the TCP/IP and MQTT limitations and provides a strong, secure, reliable transport layer.

This method of communication, in addition, avoids the need for the two (or more) peers to share their public IP addresses, information that allows a malicious actor to geolocate a potential victim and implement a series of attacks via the Internet.

3.2.2.2 dOra interoperability

Blockchain's nodes, in addition to interacting with each other, provide services to its clients. These are usually applications for sending and receiving tokens, but they may have other purposes and functionalities.

To support the client, nodes expose REST APIs that allow applications to interact with the blockchain.

dOra nodes can do the same, but to increase the level of security while simultaneously reducing operational costs, we decided to use the IOTA network as the preferred communication transport layer, both between nodes and with clients.

This choice is severely limiting for those applications that could easily interact with REST servers, but not so easily with the IOTA network.

Therefore, we realized a functional component that performs the function of a communication gateway. The dOra Communication Gateway, or DCG for short, exposes REST APIs that allow an application to send a script with a list of tasks to a committee and get the responses produced.

The DCG is a stateless system that does not require its own identity. It is activated using a preconfigured Docker image and requires very limited resources.

The DCG is designed to be completely irrelevant in the security domain of a dOra committee: if it is attacked at most it could make the committee's services inaccessible to the applications that used it. Attacking the DCG has no practical effect because activating another one with a different public IP address immediately restores the intercommunication service.

Moreover, everything exchanged between applications and DCG is visible to anyone on the IOTA transport layer, so the attacker cannot get anything useful from the DCG.

3.3 Consensus building

3.3.1 The blockchain consensus

Blockchain consensus production is one of the most important features of a node.

The node does not need to be a miner or a validator, i.e., a block producer. All nodes validate blocks that are published, reporting any anomalies, or simply ignoring the invalid block and all following ones linked to it.

Obviously, nodes that produce blocks have a more complex consensus production function that depends on the individual blockchain protocol.

In all blockchain consensus is achieved by a progressive process.

This was one of the innovative ideas that enabled the emergence of a system where consensus is achieved by the collaboration of thousands of nodes. This would not be possible using assembly voting where it is necessary to send the votes of each node to all the others.

In the progressive voting model, nodes produce blocks and link them to the previous one to cast a positive vote or link them to a previous one they consider valid to cast a negative vote on the following ones.

The longest chain contains blocks with votes that keep increasing. After a certain number of votes have been received, a block is considered definitively valid because statistically it is unlikely that anyone would be able to link a new block before it.

Although the progressive validation system sounds efficient, it is not. In fact, for many nodes to validate new blocks, the amount of transactions contained and the frequency with which blocks are produced, must be limited. In practice, the system must be synchronous and every synchronous system has a performance limit beyond which it fails.

To prevent the system from going critical, a queue of transactions waiting to be entered into a block is created and competition among users to get their transactions entered as soon as possible in the ledger is triggered. Usable space in the block becomes a scarce resource and therefore competition among users induces an increase in fees paid per transaction.

This synchronous consensus production model does not fit the needs of business applications where the time and cost to have a data entered definitively into the ledger must be predeterminable.

3.3.2 The dOra consensus

In dOra, consensus is achieved in a single step and made evident by an asymmetric signature produced by the committee of nodes. Like in an assembly voting, the committee creates consensus by a majority of its members, but without using an explicit voting procedure.

This solves one of the main disadvantages of assembly voting, namely the need to validate votes before counting them, a procedure whose cost and execution time increases as the number of voters increases.

Instead of an explicit vote on a data item, dOra nodes create a joint signature of the data item that will be valid only if the majority of the committee's nodes contribute correctly.

3.3.2.1 The dOra committee creation

A committee is created by a Governor who has the only authority to create it and to request the replacement of a node if necessary.

The governor invites each candidate node to join the committee by sending a message via IOTA using a tagged block. The message includes a list of all the DIDs of the invited nodes.

The nodes are then able to communicate with each other via IOTA network.

The first task of the committee is to create the DID identity of the committee.

The process begins with the creation of a distributed secret using a cryptographic procedure called DKG (Distributed Key Generation).

In the DKG process, nodes share secrets in with Verifiable Secret Sharing or VSS in short. VSS scheme protects against malicious operators by enabling participants to verify that their shares are consistent with those dealt to other nodes, ensuring that the shared secret can be correctly reconstructed later.

At the end of the process, each can produce the same public key, but neither of them can produce the corresponding private key.

Having the public key, each node can compute the same new DID and its related DID document that includes the produced public key and supported methods.

Nodes use now the Threshold Signature procedure to produce the DID document signature, valid for the common public key.

Finally, the node with the smallest DID (in numerical value) adds the signature to the committee DID document and publishes it to the IOTA ledger. The publishing procedure must be completed within a specified timeout. If the node fails, the next one in arithmetic order assumes the publisher role.

3.3.2.2 The threshold signatures

The signature method used by the committee is called "threshold signature."

The threshold defines the number of nodes required to correctly collaborate to produce a signature valid for the common public key.

The signature procedure in short is:

- Each node knows the data item the committee is required to sign. The data item is normally not shared by the nodes but acquired independently. This grant that each node is producing consensus on the data item it trust, not simply sing on a provided data item.
- Each node shares with the other nodes its partial signature produced with its personal shared secret produced during the DKG process.
- When nodes receive the required minimum (threshold) number of partial signatures from other nodes, they become able to produce the valid common signature by integrating the received partials (Lagrange integration).

This signature method is very flexible and reliable:

- No one knows the private key and cannot obtain it with the data at hand.
- A node can't be replaced by others committee's node.
- The signature process last for a predefined time, due to a timeout
- There is no need for all the nodes to mandatory participate in the production of the signature.
- Nodes that sign and those that do not sign, or sign incorrectly are highlighted.
- Nodes do not have to share to in advance the data they want to sign. This prevents a malicious node from signing data that it did not produce or received from an associate.
- Partial signature of invalid data does not allow integration, so it is immediately discovered by the other nodes.

The use of threshold signature allows the dOra node committee to produce consensus in one single step on any data obtained individually and independently, without the risk that someone (internal or external to the committee) could override the security of the process.

3.4 The programmability block

3.4.1 Blockchain smartcontracts

The most smartcontract model was firstly designed by the Ethereum blockchain development team. It uses a specific programming language to implement a non-complete Turing machine.

The code is published to the ledger and executed by nodes in a virtual machine that isolates the execution of the smartcontract. This avoid any risk of node attack using a smartcontract and variable environmental conditions that can lead to indeterministic results.

The state of the smartcontract (the set of its variables) is read from the blockchain at the time of activation and written in an updated version at the end of its execution.

This operating model is very efficient in handling deterministic processes, but it is not at all efficient for validating data from business applications.

This is why oracles are needed to collect data, publish it in blockchains, and trigger smartcontracts that process it.

Unfortunately, the execution time of the smartcontract depends not only on its complexity but is closely related to the load level of the network. This results from the fact that the node executes smartcontracts in the context of the mining process and includes its new state in a block exactly like a new valid transaction. Therefore, the cost of execution of a smartcontract depends on its complexity, the level of network congestion, and the willingness of the client to pay to activate it as soon as possible.

Clearly this service model is not suitable for enterprise applications.

3.4.2 The dOra script

Unlike blockchain nodes, there is no impediment of dOra nodes to access real world data sources.

This degree of freedom comes from the noncompetitive nature of the dOra consensus model based on the threshold signature.

A customer can ask a committee of dOra node to execute a Job, made of many Tasks, by sending a Tagged block over the IOTA network, where the Tag is set to the committee DID. The Job is part of a script that describes the Job configuration.

A client can ask a committee of dOra nodes to perform a job, consisting of many tasks, by sending a tagged block on the IOTA network, where the tag is set to the DID of the committee.

The committee nodes are listening on the network to receive blocks tagged with their own DID and the DID of the committees to which they belong. In this way, all committee nodes receive directly and at the same time, the same request.

Therefore, all nodes can immediately start executing the Job, which is usually aimed at collecting data from various sources via the Internet.

A job can contain the several types of tasks:

- tasks executed by each node independently:
  - acquire data from a URL.
  - acquire data from the node local storage.
  - acquire data from a text string provided by the client.
  - obtain and activate a Docker image that processes the acquired data.
  - store acquired data on its own storage.

- tasks executed collaboratively by all nodes.
  - sign acquired data po products.
  - share processed data.

- tasks executed by one node, replaced by another in case of failure:
  - publish data to a URL.

Normally, nodes do not share the data they acquire and process. This ensures that the consensus produced has significant value. But in some cases, nodes may need to synchronize the status of their tasks. This is where the activity called "sharing processed data" comes in.

3.4.2.1 Processing indeterministic data source

One of the constraints of smartcontracts executed by blockchain nodes is that their output be deterministic, or in other words, the smartcontract must be a "finite state machines."

State machines can be classified into two main types: finite-state machines and infinite-state machines. A finite-state machine is a state machine with a finite number of states and transitions

between these states. An infinite-state machine, on the other hand, is a state machine with an infinite number of states and transitions.

In practice, in an infinite-state machine, the next state can be independent of the previous state, giving rise to infinite possible next states, and can even be completely random.

In contrast, in a finite state machine, the next state can never be random.

This feature prevents smartcontracts from processing data from the real world because any data acquired from a real source, understood in its broadest definition, is always unique and not strictly depending by a previous state.

In example, if three observers ask a temperature sensor for the value it is reading, since node cannot receive a single response in parallel, the three answers are surely different because the data is composed not only of the temperature value, which in a brief period may be identical, but also includes the timestamp of the measurement, which is inevitably different for each receiver and can't be exactly predicted.

dORa allows these cases to be handled very easily. Knowing the nature of the data to be processed, the customer can provide a task sequence that performs a normalization of the raw data, creating derived dataset in a trusted environment that added to the acquired dataset empower the committee to produce consensus.

In the case of temperature sensor described above, nodes may run a task that removes the timestamp and rounds the temperature measured by the sensor to the nearest tenth of a degree. It is likely that this derived data will be identical for most of the committee nodes. Node can now produce consensus and then publish the new dataset containing both the raw data collected by each node and the derived, common, dataset on which the consensus has been reached. Any observer can test the process and its results.

To obtain the same result using smartcontract, we have two options:

- Use an oracle that basically execute the data normalization procedure.
- Modify the sensor to produce data that can be compared with a specific logic (that, in other words means to ask the sensor to execute the data normalization process.

Both approaches have a number of limitations and can lead to incorrect results.

The dOra computational model, instead, provide a standard method applicable to any data source without need to adapt existing enterprise applications and data sources.

Producing consensus on the correct task list execution makes dOra suitable to process and validate any type of data with any applications logic.

3.4.2.2 Distributed script auditing

The dOra consensus is effective because it grants the correct execution of a data manipulation script. This means that the responsibility for producing a meaningful result falls not on the committee (which cannot make mistakes) but on the person who wrote the script.

This also happens to classic smartcontracts. In fact, the network only certifies their correct execution without getting into the merits of the correctness of the new state from an application point of view. Being certain that the executed script is correct is therefore critical.

Classic smartcontracts are not equipped with any tool that provides this proof. The only tool that can be used is user-community validation, which is valid, however, until someone notices a bug, perhaps introduced voluntarily in a new version of the smartcontract.

In dOra, on the other hand, the script can itself be the subject of a validation procedure This can be achieved by creating a script that "asks" independent validators for their opinion. The committee will produce a signature on the script hash that can be verified by the nodes before executing it, and then only execute it if it is the validated one

The validators who certify the quality of the script, are in this case people who will have to audit the code and publish their evaluation on an IOTA block that will then be used by the committee to produce the certification signature of the script.

3.5 The archive block

3.5.1 The blockchain archive

To support blocks and transaction validation process, nodes keep a copy of the ledger in their local storage where they save new blocks received by the network. Normally the storage is a relational database able to support complex queries.

However, the possibility to use blockchains to store any application information, forever, is a false myth.

Infact, keeping a copy of the entire ledger can be a very demanding requirement, especially for state-of-the-art blockchains able to achieve very high performance (more than 1.000 transactions per second written in the ledger), leading to a daily increase in ledger size of several GBytes,

The result is that the number of nodes willing to voluntarily participate in the network by keeping a full copy of the ledger may decrease over time, with a progressive reduction of the ledger security.

Blockchain designers addressed this problem by allowing nodes to delete the oldest part of the ledger. Then, to validate transactions that spend funds untouched for long periods of time, nodes called "archive" nodes were activated that instead retain all the ledger history.

Unfortunately, this approach, while it has some limitations, may be suitable for handling economic transaction validation, but it does not support the conservation of application data that no node would keep. In addition, it cannot be guaranteed at all that an archive node will grant public access to its copy of the ledger

In practice, an application that needs to store data reliably and permanently on a blockchain, must activate its own archive node. This not only makes the service provided by this node fully equivalent to that of one regular storage, but it is also certainly more expensive since, in addition to the data produced by the application, it must store all the full blockchain history.

3.5.2 The dOra archive

One of the tasks that a committee of dOra nodes can be asked to execute is the data archiving in a storage controlled every single node. dOra nodes can enable S3-compliant storage (local or remote) and archive data that the client has provided, or the committee has produced.

This feature provides the same service of a blockchain archive node, but:

- the nodes store only the required data
- It is a decentralized service, because each node independently keeps its copy of the archive.

The archive service provided by dOra nodes solves the most dangerous of risks that threatens any storage: being tampered with or simply can be stopped by its operator.

However, some might argue that an archive node of a blockchain, in addition to preserving the ledger, provides cryptographic proof that it has not been altered.

dOra committee provide an equivalent proof, called "proof of conservation" and an additional grant, called “proof of authenticity”.

3.5.2.1 Proof of conservation

Any storage, including that of dOra nodes, can be tampered with because there is always someone that manages the storage who can delete its contents or simply disable it.

"Proof of preservation" is evidence that each node of a dOra committee has properly and independently preserved the data entrusted to it, so the dOra archive service is immune from any storage mismanagement.

The proof consists of the threshold digital signature produced by the committee on the data requested by a client. Each node produces a hash of the data it has kept, signs the hash with its private key producing and shares it with the other committee’s nodes. Once the threshold is reached, each node will be able to produce the common signature, which in this case takes on the meaning of "proof of conservation."

The preservation test is very effective:

- if a node has altered the data it was supposed to preserve, it cannot participate in the production of the signature and get a possible reward.

- a node cannot obtain from another node the data it was supposed to preserve and instead deleted. Not only would the request be evident in the committee activity log, but more importantly, it would be fulfilled only after the current request is completed, thus too late to avoid detection.

However, proof of preservation does not represent proof of authenticity of the data.

To certify the authenticity of the data, it must include the signature of its producer and a reliable timestamp-this is what the "proof of authenticity" provides.

3.5.2.2 Proof of authenticity

Proof of authenticity, or PoA for short, is produced by the dOra committee using a particular function of IOTA network nodes, called "proof of inclusion" or POI in short.

When a client requests the PoA of a data item, the committee signs the data and publishes it in a tagged IOTA block, then asks an IOTA node for the PoI of the block.

The POI consists of data structure (Merkle Signature) that proves that the specified block has been referenced in the Tangle by a specific milestone (from which the timestamp is derived).

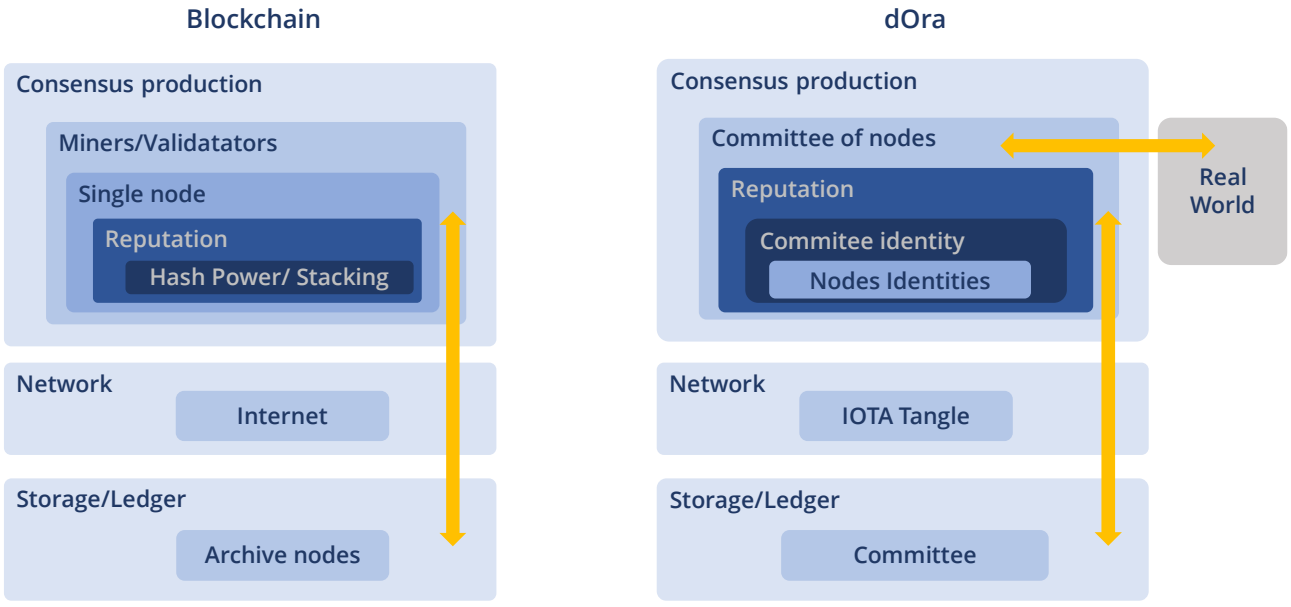
Any node, without needing a full copy of the Tangle, can verify that a PoI is valid for a given block.

Data provided with a PoA can be archived by the dOra archive or by any other reliable storage. When needed the application will always be able to demonstrate its authenticity.

Proof of conservation along with Proof of Authenticity provide the dOra archive the same theoretical blockchain security level, but with costs dependent only on the amount of data produced by the application and the number of required replicas.

4 Why dOra is a game changer

The following picture allows us to visually compare a classic blockchain node architecture with a dOra node architecture.



A different dependence of consensus-producing functional components is easily seen, but the most striking difference is the ability for dOra nodes to interact directly with the outside world.

Below we look in detail at the innovative features of the dOra node and its positive effects.

4.1 Identity based communication

One of the most critical aspects of the Internet and cloud-based platforms is that they must be reachable by clients but protected from multiple risks. Hence the need to use VPNs, virtual networks, tunnels, firewalls, and many other technologies that attempt to protect the platform but are not strictly tied to it.

dOra solves this problem in an elegant and economical way: all communication between dOra nodes and clients is done using the identity of the committee, nodes, and clients. Their TCP/IP address does not have to be published, so dOra nodes and clients cannot become targets of any cyber-attack, nor can they be geo-localized.

An attacker can only attempt to hack a DID identity, but the only way to succeed is to gain access to the DID document controller's private key. This is already normally challenging and simply impossible in the case of the identity of a dOra node committee because committee's DID private key does not exist.

There is no way to impersonate the committee and publish false, but properly signed, data.

Delegating a critical process to a committee of dOra nodes is therefore more secure as having it handled by a VPS super-protected by state-of-the-art cyber security technologies. This obviously implies that using dOra is significantly more cost-effective.

The distribution of confidential data to specific recipients is another use case that the dOra system supports efficiently. The use of the IOTA network as the transport layer ensures data delivery, and the use of the recipient's DID as the addressing method also allows the message to be encrypted. Although the data will be temporarily available on every node in the IOTA network, only the recipient can decrypt it.

4.2 Collaborative consensus

dOra's innovative consensus-building model is another advantage over any blockchain. As we have seen, in a blockchain, consensus is achieved incrementally, with varying costs and performance, and to validate enterprise application data we must rely on oracles.

The ability to collect data from any source, process it with any logic, securely sign it, and publish it anywhere allows dOra to achieve the same functionality as a blockchain, but on any type of data in a cost-effective and efficient way suitable for supporting enterprise applications.

This was the dream of blockchain designers, which, however, cannot be realized in a system (the blockchain) where consensus is based on a competitive organizational model. dOra succeeds where blockchain failed because the organizational model is collaborative.

4.3 Distributed storage

Finally, dOra realizes another unfulfilled promise of blockchain, the ability to store any data for as long as necessary, resilient to attack, and with the assurance that the returned data is consistent with the original.

4.4 dOra scalability

Scalability is one of the weak points of blockchains but is not a problem for the dOra system.

In fact, it is normal for each committee to have a performance limitation given by the characteristics of its nodes but there is nothing to prevent attaching multiple committees that operate independently.

As happens on the Internet, the first decentralized network ever built, traffic handled by one group of nodes does not impact on all others, allowing overall performance to scale simply by increasing the number of nodes and committees.

4.5 Private or public committees

The demand for private blockchains stems from a desire to leverage a distributed ledger in enterprise application contexts that, as we have discussed, are not compatible with the operational models of public blockchains.

In dOra this problem does not exist, however, an organization may find it necessary to manage nodes on a committee.

This is the mode of operation with which dOra will initially be available. In fact, choosing nodes based on their authority requires a minimum of history on which to base an objective evaluation. So initially dOra committees will be based only on nodes chosen by clients.

The security of dOra committees is not impacted by this possibility. It simply meets the needs of organizations that by their nature cannot base their operational processes on services offered by third parties, without formal contracts and well-defined legal jurisprudence.

When dOra is up and running, nodes can be selected by a committee Governor, or automatically, on the Assembly.sc platform, which is created precisely to connect those who want to make a profit from providing their own nodes and those who want to activate a committee and need reliable nodes.

5 dOra economics

When the development and test roadmap is completed, the dOra nodes will be chosen via the Assembly.sc platform.

Therefore, the remuneration of the nodes will also be via ASMB tokens.

Remuneration will be linked to the tasks constituting each job submitted to the committee. Remuneration will then be divided between all nodes that have completed tasks producing a threshold signature.

In the output of this task, in addition to the signature, there is the list of nodes that contributed to producing it and the list of nodes that did not contribute.

Periodically, the same nodes perform a task in which they forward the tokens received from the committee to the addresses of the individual nodes that make it up, in proportion to the signatures in which each node has correctly participated.

Another task that provides direct remuneration is the storage task in the node storage.

This task has a maximum retention duration of one month.

The customer requesting data storage will send tokens proportional to the space requested along with the job request. At the end of the retention period (one month) the customer has five days to extend the retention period by an additional month. To submit the request, he will send additional tokens referring to the ID of the object he wants to retain. The quantity of tokens requested will depend on the space and price list in effect at that time.

If sufficient tokens are not sent by the deadline, the data will be deleted from the nodes' storages.

When the retention period expires, the nodes produce a joint signature that proves which among them has correctly retained the data and is therefore entitled to receive the portion of tokens sent by the customer. Nodes that are not able to sign, thus have not preserved the data, will not receive the reward.

The service price list will be expressed in euros (or dollars) and will therefore be variable in ASMB according to the market price. This is why the number of ASMB required by each task can vary in time.

5.1 dOra business model

The business model of the dOra node is a classic service-oriented business model, in which costs are covered linearly by prices, also similar to the remuneration model of validators/miners of a blockchain. In fact, being remunerated in ASMB tokens that have a limited supply, in addition to the price-cost revenue, it is possible to obtain additional remuneration in USD or EURO as the value of the token obtained and stored increases.

An additional business opportunity and price model come from the possibility to run a committee of nodes dedicated to a customer’s application.

In this case the price will be defined according to the required SLA.

Finally, dOra technology can be licensed to qualified third parties.

6 Use cases

All dOra application scenarios can be traced to three use case categories:

- Trusted data source
- Trustable bridging
- Multi-party trustable data exchange

6.1 Trusted data source

In this type of service, an authority certifies the origin and quality of a dataset.

Services that provide timestamps, market price information services, both stock markets and consumable markets, belong to this category. Applications that produce synthetic reports from raw data also fall into this type of use case.

Also belonging to this category of use cases are all news validation systems, where multiple sources must agree on the news itself. In this, committee nodes can search for multiple sources that provide the same news item and only if they find more than an expected amount, make the publication.

Finally, this use case can be used to automate the validation of digitally signed documents that nevertheless require the countersignature of an authority. In this case, the authority can be a dOra committee itself.

The certified data is kept by the committee in a storage/database and provided at the request of users or other applications that will be assured of the execution of a quality verification protocol applied by the committee to the original data.

The identity of the committee that produced the data signature is well known, so anyone can validate the signature.

6.2 Trustable bridging

In these usage scenarios, the committee publishes collected data to a third-party platform specified by the client.

This can be a simple database, storage, or blockchain to transfer value between different networks.

In each scenario, the committee signature meets the requirements of the target platform so that whatever type of application needs to read the data produced, it will be able to verify the committee signature.

Of course, in addition to transferring data, the dOra committee can create logically derived data.

This is the operational model of a "sense - response" system where the committee, having acquired data and verified its quality, can produce, and publish data to other platforms that will "react."

The response may consist of triggering an alarm, executing a procedure, or activating an actuator.

For example, an urban logistics management system may activate variable signage upon receipt of committee-certified information, which in turn is derived from data collection from sensors that monitor city traffic.

A similar case may involve the collection of air pollution level data. When certain thresholds are exceeded, pre-programmed decisions can be triggered.

6.3 Multi-party trustable data exchange

In this usage scenario data sources and recipients are multiple, flows intersect, and data can be layered producing both synthetic data and raw data.

This is the typical context that occurs in a supply chain where competing players may exist and where the various sub-processes require reliable information as input to produce equally reliable output, used, however, by multiple recipients.

The dOra committee in this case provides not only certification of data, but also the guarantee of neutrality, ensuring that all parties have access to the correct data without anyone being advantaged or disadvantaged.

6.4 Use case in the crypto market space

In addition to use cases in the traditional market, of course dOra may have many applications in the blockchain and cryptocurrency-based solutions market.

As examples, we present two applications commonly used in this area.

6.4.1 Distributed blockchain bridges

Blockchain bridges are applications that virtually interconnect two blockchains allowing value to be moved bidirectionally.

A user with native tokens on blockchain A can virtually move them to blockchain B with the help of the bridge.

The virtual move occurs with this task sequence:

- The bridge creates an ADD1 address dedicated to the client on blockchain A. This address is controlled only by the bridge.
- The user sends his own tokens from blockchain A to address ADD1 and in the message includes the address ADD2 of blockchain B where he wants to receive the equivalent tokens.
- The bridge, prints native tokens from blockchain B in same quantity that has been sent by the client to address ADD1
- Finally, the bridge sends the created tokens to the address ADD2 controlled by the client.

The reverse operation involves burning the tokens on blockchain B created for the client, returned to the bridge.

The criticality of this application lies in the security of the method by which the private keys of the controlled addresses are stored. Many approaches can be taken, but each has a side effect.

For example, an HSM, which is a secure hardware device capable of generating signatures and storing keys within it, can be used. However, someone has configured it and manages it, and if they wish, they can disable it, making all the funds stored in the controlled addresses inaccessible.

Another approach involves the use of multisign addresses, where keys are split and delivered to several independent parties. In this case, the main risk is related to the minimum number of signers that must necessarily participate in the operation. It is usually required to be more than 90 percent, and therefore it is sufficient to attack a minority of nodes and take them offline to crash the system.

Assigning the dOra committee to generate the addresses and produce the signatures is the safest feasible method. No one knows a valid private key for the addresses, and all that is needed to obtain a valid signature is for most nodes to cooperate.

Since they do not communicate via the Internet it is impossible to locate them, attempt to tamper with them physically or with cyber-attacks, and threaten those who manage them.

6.4.2 Distributed wallet

7 Node requirements and technical specifications

The software executed by dOra nodes is written in RUST.

The server configuration (physical or virtual) needed depends on the task execution requests.

For example, if it is necessary to support complex data processing, then the server must be appropriately sized to allow the node to participate in consensus production.

Similar considerations apply to storage. dOra nodes expect to use S3-compliant storage, so node managers can opt to enable local object storage (e.g., https://min.io/) or use a compliant external provider.

Processing tasks are performed in a Docker compliant virtual environment, so the client must build a Docker image and upload it to a repository reachable by the node.

The node software is written to support parallel requests from multiple clients. Consistency validation of parallel requests is not the responsibility of the nodes.

8 Roadmap

July 2023

The first private release of dOra is released.

Features included:

- Government
  - Committee creation by the Governor.
  - Create a committee public key.
  - Create a committee W3C DID and its DID Document.
  - Sing the DID Document.
  - Publish the DID Document onto IOTA Tangle.
- Network
  - TCP/IP (for testing)
  - IOTA tagged blocks.
- Tasks
  - Creation of a job (script), as a sequence of tasks
  - Collect data from a provided string of characters.
  - Collect data from a provided URL (HTTP GET).
  - Collect data from an IOTA.
    - From tagged blocks.
    - From transaction blocks
    - From alias output
  - Download of a Docker image form a provided URL
  - Execution of the selected Docker images in a restricted environment
  - Publish data to a provided URL (HTTP PUT).
  - Publish data to IOTA.
    - In a tagger block.
    - As signed valid transaction.
  - Sing a data with threshold signature.
  - Share data among the committee nodes
  - Store data in nodes' S3 compliant storages
  - Publish data from node' S3 compliant storages
- User interface
  - Receive task requests/scripts via IOTA.
- Integration
  - Stateless REST – IOTA gateway that allow to interact with a committee of node via REST API (TCP/IP) (middleware)

Provided demo tests:

- Stock exchange price Oracle
    - The committee of nodes collects the current price of a required Stock on the NYSE marketplace, sing and publishes it on the IOTA Tangle in a tagged block

- Test of a committee signature of a published block containing a NYSE Stock price
    - The committee of nodes collects the selected block using the provided tag, sing the test result and publishes it on the IOTA Tangle in a new tagged block

- Stock exchange price on Tweet
    - The committee of nodes collects the current price of a required Stock on the NYSE marketplace, sing and publishes it on the Tweeter

September 2023

The first public release of dOra based service demo (web page)

Added features:

- Government
    - Communicate to the Governor the committees DID.

- Network
    - TCP/IP
- Tasks
    - Generate proof of conservation
- User interface
    - Web page that allows to interact with a committee of node.
    - Committee status web page

November 2023

The first public release of dOra code base

Added features:

- User interface
    - Mobile app suitable to manage controlled identity.