



# blockchain technology whitepaper

**2019**

**The key technology  
of tomorrow**



@BCTechConf #BCC19

**[www.blockchainconf.net](http://www.blockchainconf.net)**

# Index



## Blockchain Concepts

---

- Blockchain glossary: Asking the right questions** 4  
Seven experts on the fundamentals of blockchain
- Blockchain – between hype and world improvement** 8  
by Ingo Rammer
- „Decentralization is about freedom, flexibility, and choice. Blockchain is just one more tool“** 14  
Interview with Ricardo J. Méndez
- Blockchain and Web 3.0: The future of private data protection** 16  
by Premjith Purushotham



## Blockchain Impact & Strategy

---

- „2019 will be a continued reiteration of how people are using blockchain“** 18  
Interview with Brian Behlendorf
- Why the development of next-generation blockchain platforms must be led by the community** 22  
by Vladislav Dramaliev
- Looking forward to a golden future for the blockchain industry** 25  
by Dr. Demetrios Zamboglou
- IoT and blockchain are ready to drive a manufacturing revolution** 28  
by Ilya Pupko
- Forget about the new internet! Blockchains are the latest ‘Linux alike’ revolution** 30  
by Phil Zamani

# Index



## Blockchain Development

---

**How to build a Blockchain with Hyperledger Fabric and Composer** 32

by Thorsten Deelmann and Jannik Hüls

**Tutorial: Develop your own Ethereum Consortium Chain** 38

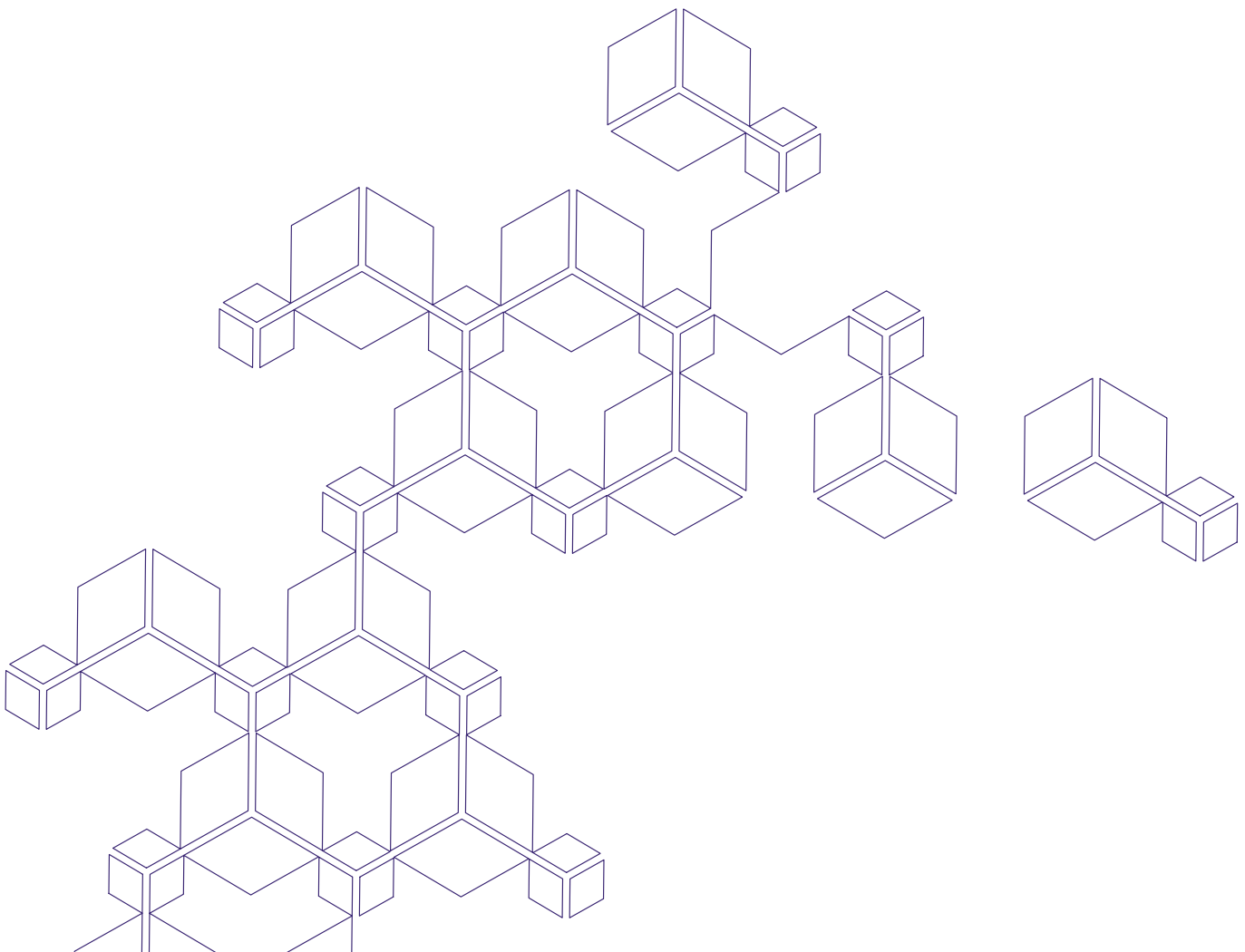
by Christian Scharr

**Reliable blockchain oracle for financial data** 42

by Samuel Brack

**Corda, the open source enterprise blockchain** 45

by Christian Koller



Seven blockchain experts weigh in on the fundamentals of blockchain

# Blockchain glossary: Asking the right questions

Can blockchain transform the world? Before we can answer this question, we need to understand what this technology can and can't do, what it is and what it isn't. Therefore, we have to go back to basics and explore the fundamentals in order to eliminate all misconceptions about blockchain.

by Gabriela Motroc

We invited seven blockchain experts and Blockchain Technology Conference speakers to weigh in on the fundamentals of the Blockchain Technology. What is the difference between blockchain and distributed ledger technology? What are the most important questions you should ask yourself before using Blockchain technology? Can anyone create their own blockchain?

## What is the difference between blockchain and distributed ledger technology?

**Ingo Rammer:** A blockchain is essentially just one kind of implementation of a distributed ledger. I am personally guilty, however, of sometimes using 'blockchain' in a conversation as the initial moniker to refer to DLTs as a whole – simply because this is the term most non-technical people are familiar with when referring to decentralized technology.

I then try to steer the conversations to 'DLT' over time as it is the more correct term for the phenomenon I care most about: decentralization, not the particular implementation.

**Peter Lawrey:** A distributed ledger is a highly redundant, decentralized ledger of transactions. The purpose is to achieve high availability. Such a ledger can perform at high throughputs, as the nodes trust each

other and don't need to validate each other's work. This works well within a single organizational unit, which is a very common use case.

A blockchain is a distributed ledger where the nodes don't trust each other. By trusting a protocol, instead of the nodes running a service, they can be Byzantine Fault Tolerant, where up to one-third of nodes are actively trying to defraud the system. However, if more than half the nodes are compromised the whole chain is. This works best when you have many individuals or organizations running a service, where there is an advantage in not having to trust each other, e.g. a collection of service providers adding liquidity to a market with equal access to that service.

**Arnaud Le Hors:** This is a question on which not everybody agrees today, but generally blockchain is considered to be one type of distributed ledger technology in which the ledger is stored as a chain of blocks. This is not the only possible form of storage though, and there are distributed ledger technologies that have similar characteristics while using a different form of storage.

**Vinita Rathi:** A distributed ledger is a database that is spread across several nodes or computing devices. Each node replicates and saves an identical copy of the ledger. Each participant node of the network updates itself independently. The most unique feature of distributed ledger technology is that the ledger is not maintained by a central authority. Updates to the ledger are independently

## *The structure of the blockchain makes it distinct from other kinds of distributed ledgers.*

constructed and recorded by each node. The nodes then vote on these updates to ensure that the majority agrees with the conclusion reached. This voting and agreement on one copy of the ledger is called consensus and is conducted automatically by a consensus algorithm.

Blockchain, on the other hand, is one form of distributed ledger technology. Not all distributed ledgers employ a chain of blocks to provide a secure and valid distributed consensus. A blockchain is distributed across and managed by peer-to-peer networks.

Since it is a distributed ledger, it can exist without a centralized authority or server managing it, and its data quality can be maintained by database replication and computational trust. However, the structure of the blockchain makes it distinct from other kinds of distributed ledgers. Data on a blockchain is grouped together and organized in blocks. The blocks are then linked to one another and secured using cryptography.

**Svetlin Nakov:** Distributed ledger technologies (DLT) are decentralized systems, which hold a sequence of transactions (ledger). The ledger is maintained by a peer-to-peer network (the DLT network) of nodes, which use a consensus algorithm to synchronize the data between the nodes in a reliable and attack-resistant manner. DLT systems can be implemented as public (permissionless), private (permissioned), or hybrid systems. The underlying structure of the DLP can be blockchain, DAG (directed acyclic graph), hash-graph or other structure and the consensus algorithm can be proof-of-work, proof-of-stake, practical byzantine fault tolerance, asynchronous byzantine agreement or other. The DLT technologies include a large class of distributed systems and the blockchain is just a subset of them.

Blockchain technologies use a “chain of blocks” (sequence of blocks). It is a linear structure (linked list). Each block cryptographically refers to the previous and holds a set of transactions (signed documents). Transactions may hold asset transfers (in the financial sector), plain documents (in a document management system) or results from a smart contract execution. Most blockchains use a public-key cryptosystem, typically elliptic curve cryptography (ECC), to sign transactions. Transactions are signed by the blockchain users who control their blockchain addresses through a private key. The blockchain is maintained by a peer-to-peer network that synchronizes the chain and constantly adds the users’ new transactions through a consensus algorithm.

In short, blockchains are a technical design and implementation of DLT, but not the only implementation.

**Jana Petkanic:** Distributed ledger technology could be considered as a broader term than blockchain. Not every DLT is organized in a “chain of blocks” encrypted by means of the latest cryptography tools. DLT is basically a database that is spread across several nodes.

Blockchain is capable of creating digital assets in an environment where stakeholders do not trust each other.

**Christian Junger:** In essence, blockchain and DLT are very similar, though there are some glaring differences between the two.

From a technical perspective, blockchain is a specific implementation of a DLT, so in a way blockchain is an evolution stemming out of the DLT technology. Each of these concepts requires decentralization and consensus among nodes. However, the blockchain organizes data in blocks. As far as cryptocurrencies are concerned, most of the known ones (Bitcoin, ETH, Litecoin etc.) are blockchain implementations – meaning all of them are DLTs. However, not all cryptos are blockchain implementations, such as IOTA. As such, IOTA is a good example of a DLT that is not a blockchain.

### **What are the most important questions you should ask yourself before using Blockchain technology?**

**Ingo Rammer:** I think the most important question is: Does my use case support or require decentralization between peers? If it doesn’t (for example if you have a commercially powerful entity who controls, say, a supply chain as a buyer and forces suppliers into certain ways of handling data), you might be better off using a central database with an HTTPS API in front.

Yes, you can implement centralized use cases with a blockchain – but doing so usually just increases complexity without providing tangible benefits. Yes, today, companies might also see a marketing/sales/recruiting-benefit when implementing blockchains, but I truly believe that this shouldn’t be the main driver.

**Peter Lawrey:** Even if you need a blockchain, you will still need a database to support queries. Once you have a database, does the blockchain add anything technically? If so, what do you need it to do and will this help you choose the best blockchain solution for your needs?

**Arnaud Le Hors:** Who are the participants in my network? Blockchain isn’t a one-player game. It is a system which only makes sense when there are several participants, so it is crucial to determine who the participants are.

One doesn’t necessarily need to have them all be part of the development process, but it is wise to have at least one representative of every type of participant you expect to have in the network.

**Vinita Rathi:** Blockchain, despite its potential, is not an ideal fit for all problems. The most important question that I always ask my clients is: why do you want to use Blockchain, what is the purpose and what you are looking to achieve by using Blockchain? If there are advantages to the use of technology, it also brings a lot of unknown risks to the system.

Some of the key questions I ask:

Does the system need to be immutable? Immutability

## *Blockchain isn't a one-player game. It is a system which only makes sense when there are several participants.*

ensures that no particular individual in the ecosystem changes/modifies historical data (records). If the solution we are looking for has no need for immutability, using blockchain could actually complicate things. Similarly, blockchain is by nature append-only, hence will not allow to remove/delete a record or correct them. This feature is directly in conflict with GDPR and its “Right To Be Forgotten” principle. Hence, blockchain (within the EU) will make it impossible to store personal data, at least in its current form.

Is the solution for global reach or controlled reach? If you are looking to implement a solution for a particular organization or pre-defined user base (closed group), there is no need to implement blockchain as trust already exists within the organization.

Does your system have real-time needs?

By virtue of what blockchain technology is, it can never be good enough for real-time needs at least in its current shape and form. If the system expectations are to have sub-second performance throughput, blockchain is in most cases not going to fit in the architecture.

**Svetlin Nakov:** I always ask whether the system can be implemented without blockchain and why the blockchain is valuable for a certain project. If I get a valid answer, I ask about why decentralization is important for this project. If it is not important, blockchain should not be used.

Next, I ask if the project needs to have its own token economy and why. If I get a valid answer, I continue asking. If I don't get a valid answer, then the project should not be based on blockchain, but on traditional centralized software architecture.

Blockchain should not be used in every project. Many people try to use DLT and blockchain technologies just because they are “modern,” but soon fail. The true application of blockchain is building a decentralized system, serving a decentralized organization with decentralized processes, where no single point of control exists. Most businesses do not match this scenario.

Some businesses may partially use the blockchain, e.g. for immutable storage of document hashes or asset tracking. Others use the cryptocurrencies as an additional payment method. Others use smart contracts to ensure that some logic is executed in a fair and decentralized way. Such scenarios are valid and are emerging along with the entire blockchain ecosystem. These organizations have good reasons to use blockchain and most probably will be successful.

**Jana Petkanic:** What benefit does blockchain bring me and am I capable of involving an entire ecosystem of stakeholders? Simply put: if you cannot invite parties outside of your organization to participate in blockchain solution and/or you use blockchain as a gimmick with no cost reduction or efficiency improvement – you probably don't need blockchain in the long-term.

Also, when you struggle with specifying which assets can you digitize (due to regulatory issues), tokenized blockchain might be too early or too redundant for you.

**Christian Junger:** This is a rather difficult question since there are plenty of questions a business should ask themselves before making use of blockchain technology. However, in my opinion, the most important question is to what extent your business model/product can actually make use of blockchain. For example, if your company doesn't rely on software or a computer network, you most definitely do not need blockchain. In other words, don't just try to implement blockchain out of FOMO.

### Can anyone create their own blockchain?

**Ingo Rammer:** If we're looking at private blockchains (as they are mainly used in business-to-business scenarios), then, yes. Absolutely. The complexity is also not necessarily in the technology. Open source platforms like Hyperledger Fabric provide relatively easy ways to implement private and permissioned blockchain networks between multiple participants. (You can think of it as similar to installing a database product on top of which you run stored procedures – but this time it's a blockchain network that runs smart contracts.)

The hard part is convincing other independent organizations to join your network, as this requires a clear set of benefits and a solid organizational and technical governance process. I believe that in the long run, the networks that will succeed are the ones which have a truly decentralized governance and revenue model that is understood by all participants. Or to formulate it differently: for me, the litmus test of a blockchain is whether or not it would survive when its initiators eventually decide to abandon it. I think today, the jury is still out whether or not blockchains that don't fulfill this requirement will provide enough value in the long run.

**Peter Lawrey:** To create a blockchain, you need to record transactions in a block and have each block include the hash of the previous block creating chain, thus a blockchain or chain of blocks.

This is pretty easy. The hard part is making it decentralized with a reliable, trustworthy consensus strategy that is more useful than a database alone.

The approach we take with Chronicle Decentralized is to have four stages:

- replication of blocks
- gossiping of which blocks every node now has
- vote on which blocks should be included in the next round
- when a majority vote is detected, announce what blocks have been added in what order so they can be processed.

## *Building your very own private blockchain is no longer as tedious or challenging as it was a couple of years back.*

**NOTE:** This makes a number of basic assumptions in our design:

- We want to have nodes adding blocks concurrently to increase throughput.
- We know which nodes are running the service so we can determine when we have a majority.
- A majority of our nodes are running in sync.

**Arnaud Le Hors:** Yes, anyone can create their own blockchain. Although, as pointed out earlier, this technology only makes sense when more than one participant is at play. There are many technologies now available for people to develop their own blockchain network.

Several of them are open source, such as the ones produced by Hyperledger, and Hyperledger Fabric is now supported by many vendors and cloud providers, including IBM.

**Vinita Rathi:** The short answer is yes; anyone can create their very own blockchain. As daunting as it may sound, building your very own private blockchain is no longer as tedious or challenging as it was a couple of years back. The easiest and most common way of setting up a private blockchain is by using the existing Ethereum network.

Ethereum blockchain is simply lots of EVMs or nodes connected to one another to create a mesh. Each node runs a copy of the entire blockchain and competes to validate a transaction (mining). Whenever a new node is added, the entire network is updated and is propagated in order to ensure that each node is in sync.

Ethereum provides tools like Geth, Eth, and Pyethapp that would cause you to become a node in the Ethereum network and download/update a copy of the entire Ethereum blockchain. These tools allow users to set up a private blockchain.

The blockchain is nothing but a distributed ledger in which transactions are recorded chronologically and publicly. These transactions are recorded in blocks, and all the nodes in the network compete to find the next valid block.

The very first block in the chain is what is referred to as genesis block. To create a private blockchain, one needs to create a genesis block. The above-listed tools like Geth can be used to create genesis block, which in turn will be the start of a custom private blockchain.

**Svetlin Nakov:** An experienced developer can create their own blockchain network, of course. This can be done by forking an open source blockchain system that already exists (e.g. Bitcoin or Ethereum) or writing their own solution from scratch. Usually, people don't write a blockchain system from scratch, because this is too complicated. Instead, they start from an existing project, clone it, and modify its code to add new features and change the consensus rules or other logic.

I am part of a blockchain network development project (protocol development), which started from forking the QTUM blockchain and later took its own path of development. This is a good example of how such protocol development project can work.

**Jana Petkanic:** Yes, they can, anyone can – but why? A sufficient technology is already out there, but it's crucial to have the incentive to bring stakeholders outside of the organization on board and reach consensus with them first. Do we need blockchain? Yes/no. What blockchain? Perhaps we try a pilot on a public chain (almost zero costs) and then evaluate? Do not reinvent the wheel when it's already there. Think of end users and benefits first. Tech will follow your drive and needs.

**Christian Junger:** Yes. There are two methods. You either fork an existing one or you create your own blockchain. Both ways will require extensive technical knowledge.

## MEET THE EXPERTS



**Ingo Rammer** (@ingorammer) is co-founder and managing director of Thinktecture AG.



**Vinita Rathi** (@VinitaKRathi) is the CEO and Founder of Systango.



**Peter Lawrey** (@PeterLawrey) is a Java Champion with the most Java & JVM answers on Stackoverflow.



**Svetlin Nakov** (@svetlinnakov) is a passionate blockchain engineer, trainer and experienced developer in a broad Jana Petkanic (@JeanneDeBit) is a blockchain consultant and founder of Blockchain Talks.



**Arnaud Le Hors** (@lehors) is Senior Technical Staff Member of Web & Blockchain Open Technologies at IBM.



**Christian Junger** (@ChristianJunger) is CEO and co-founder of the German blockchain start-up MADANA.

## On the Technical Background of Smart Contracts

# Between hype and world improvement

Anyone who has been dealing with blockchains in the business environment in recent months knows this: Smart Contracts are almost like magical problem solvers, from the automated billing of solar power and other goods deliveries, to the independent triggering of contract penalties in closely monitored areas such as the agreed maximum temperatures of a cold chain. The potential applications are versatile, and they even seem endless. But how much of it is true and how much is just hype? And above all: What is the actual technology behind it?

by [Ingo Rammer](#)

To understand the role of smart contracts, we must first briefly take a look into what a blockchain actually is. On the lowest content-related level, it is a list of signed statements by individual participants. These statements cannot be technically changed or deleted at a later date. In practice, machine-readable transactions are used for this, which, much like a transaction log of a database, determine the current actual state of all data throughout the entire system. Just like a database, each individual node in a blockchain network saves the current actual state in a second format, which makes querying easier.

So far so good. But why would we use a blockchain instead of a replicated database? The short answer lies in the timing of the rule check (both on a technical and business level) and in the error handling. Compared to a database, this timing is pushed forward in a block chain and is carried out early on when the transaction is being recorded, rather than later during replication. That is

precisely the job of smart contracts, which run distributed in a blockchain network.

An application suited for blockchain would be the porting of telephone numbers between mobile service providers. An excerpt from this process provides us deeper insight into the underlying processes: Put in simple terms, a central database in Germany maps the assignment of telephone numbers to the respective underlying mobile network operator. What would you need to do if you wanted to manage this information and the underlying processes locally in a blockchain network instead?

## Example telephone number porting: the network

For the present case application, it makes sense to establish a private block chain network as an implementation platform. Unlike in a public network, such as Ethereum, access to a private network is allowed only to selected participants. In our example, both the German Federal Network Agency and the larger mobile operators would



be participants in the private blockchain network. Each participant operates a complete node in the blockchain network. Smaller providers who do not want incur this expense would use the service by connecting to one of these network participants.

The technical platform for such a private blockchain network could be built on open source tools like Hyperledger Fabric [1] or Parity [2] (an Ethereum derivative), for example. Both infrastructure variants can run in a proof-of-authority configuration. This consensus algorithm works differently than the well-known CPU-intensive proof-of-work mining algorithms that Bitcoin uses, for example. For more information about these algorithms, refer to the „Proof-of-Work, Mining, Consensus and Co.” box.

The authorizations to participate in our example blockchain could ideally be managed by a legitimated body. In the case of telephone number porting, the German Federal Network Agency would be the best option. In other setups where cooperation in a network is voluntary, often democratic or economically motivated decision-making processes play a role in the admission of new participants. The initiators of a closed block chain can independently determine which method is preferred for the granting or deletion of access authorizations. In the case of the tools presented, such decisions can be incorporated into code or configuration information.

### Transactions and Smart Contracts

In the case of the telephone number assignment example, three types of transactions can be identified in terms of smart contracts (in a very simplified perspective):

- The German Federal Network Agency makes an initial assignment to a provider for a telephone number assigned for the first time.
- In the process of porting, the new mobile network operator requests a transfer of a number from the previous provider.
- The previous mobile operator confirms the porting of a number to the new provider or rejects it following information and confirmation by the contracting party.

To solve this process in a classic sense using a replicated database, the operator of the database would typically separate the read and write logic. The former would run on replicated read-only duplicates at the respective mobile service provider. All change processes would be handled centrally (e.g. via a web service), which would then write and distribute corresponding transaction logs so that each participant can update its read-only duplicate.

If we were to try - even without blockchains - to reduce the dependency on this central location by allowing multiple sites (or even all participants involved!) to write directly into the database, the question of securing data integrity would quickly surface. How is compliance with the agreed rules checked? Finally, it would be fatal if the code run by a participant would (intentionally or unintentionally), for example, remap the phone numbers of all participants without further security checks.

### Smart Contracts: the acceptance rules of a blockchain

Therefore, most blockchain implementations do not allow the creation and replication of the chain of sig-

## Proof-of-Work, Mining, Consensus and Co.

All blockchain platforms have a so-called consensus procedure. This technical procedure determines how a network participant recognizes whether or not a transaction (or several such transactions combined in a block) is valid. The most important criterion typically defined there is the selection of the node responsible for generating the next block. This cannot and should not always be the same node, as otherwise the node could assume sole control over the blockchain. Therefore, a rule must be defined that determines a node from a pool of participants.

The best-known consensus procedures are based on a so-called proof-of-work mechanism. It is currently used by Bitcoin or public Ethereum networks, for example. All participating nodes (often also referred to as „miners”) try to solve a cryptographic puzzle. The idea behind this is that all participants are randomly and alternately selected in proportion to their investment (hardware and power consumption) as creators of a block. Since these networks have horrendous (and ecologically barely viable) energy consumption, many projects have been developing alternative procedures in parallel for several years. A promising procedure for public networks is currently the so-called proof-of-stake procedure. Each node deposits monetary collateral that determines the likelihood that it will be selected as the generator of the next block. If a node attempts to misuse its current position to gain control over the blockchain data (for example, through defective blocks), a part of its collateral will be automatically collected (or destroyed, which will also increase the value of the remaining currency for all participants increase in the same ratio).

These two approaches to consensus are often used in public networks where participants do not know each other. Private blockchain networks in the form most commonly encountered in the business environment usually do not need these, because the participants know each other and are typically subject to „real” contracts. Blockchain participants prove their identity to each other by using accepted cryptographic signatures. This type of procedure is often called the proof-of-authority. If a participant behaves improperly (and tries to attack the network), they can automatically be excluded from the network by removing their certificate of trust. Conversely we can say that in the business use of blockchains, there is usually no need to waste CPU cycles and power through mining.

ned statements discussed above so easily, but rather link their execution to specific code in advance of a transaction, which is automatically executed on each node. This code is the smart contract. In most blockchains, the smart contracts has three responsibilities: (1) the examination of a transaction before execution, (2) the construction of the actual view of the data (or of the „world states“) after the transaction and (3) the possible triggering of further transactions. Simply put, the smart contract implements that part of the business logic that is shared among the participants.

In our example of telephone number porting, the smart contract could offer four functions that match the use cases defined above. Each of these functions would be called by the user through a corresponding block chain transaction.

Depending on the blockchain base technology chosen, the smart contracts also enclose the read logic in the respective world state of the smart contract. It should be noted that these methods should only be conducive for convenient read access. Under no circumstances should these read logics include the authorization check for individual reading rights of individual participants. This data has already been replicated by all blockchain participants and therefore basically visible. If such visibility is not desired, encryption or the partial use of point-to-point transmission methods can restrict data distribution accordingly. Some blockchains offer basic tools for this.

### Step by step

But let's go back to the smart contracts. For the examples presented in this article, I will use Solidity - the contracts are therefore executable on any Ethereum derivative for both public and private blockchains. However, the concepts presented can also be applied to most other blockchains. The reason I chose Solidity is the availability of easy-to-use development tools: For example, the Remix [3] open-source development environment provides a way to locally develop, test, and debug Solidity code without the need to connect to a real blockchain network.

As a first step, we need to define a data model that defines the actual state of the smart contract data. Expressed in simplified terms, this basic state in our example case includes the following three aspects:

- Determining the identity of the regulator - in this case the German Federal Network Agency. Transactions sent from there also have the authorization to associate a telephone number with an operator for the first time.
- Definition of a key/value assignment, comparable to a dictionary, This association allows the identification of the current operator's identity for each registered telephone number.
- Definition of another key/value assignment that records the identity of the porting destination (the new

mobile operator) for each telephone number currently in a porting.

(by the way: the data structure described above is, of course, a great simplification. To make this article more readable, I have allowed myself to disregard some aspects that would also be indispensable in practice. Thus, for example, I have not defined a list of permitted identities (each participant in this network could therefore make up as many mobile operators as they want), nor are escalation processes or time limits taken into account during porting. However, the underlying considerations and technologies allow for easy extension using these parameters.)

The addressed „identities“ are each technology-specific sender addresses. In practice, these are all values that correspond to the public key from an asymmetric key pair. This means that by examining the cryptographic signature of a transaction, you can determine which identity this transaction was signed for. This allows us to make sure that, for example, only the mobile phone operator currently responsible for a telephone number may change their data record.

So let's first take a look at Listing 1 for the base code of our Smart Contract, which implements the data model outlined above.

The conceptual persistence model of Solidity is relatively simple: All member fields of a smart contract are automatically stored in their persistent world state. In this case, this affects the fields *allPhoneNumbers*, *requestedTransfer* and *regulator*,

The constructor in this code snippet uses another peculiarity of Ethereum derivatives: A smart contract is installed on the network by making its compiled bytecode known to the network with a special block chain transaction. This blockchain transaction (like any other transaction) has a sender, which can be evaluated and stored using *msg.sender*. As a first step, before a participant of our number porting network could use the smart contract, the network agency would have to send this installation transaction to the network once. In response, the sender then receives an address that references the instance of the smart contract just generated. All further transactions (from all participants) are sent in succession to this concrete instance address.

As I mentioned above, this is a technical peculiarity of blockchain networks based on Ethereum. In networks

### Listing 1

```
contract PhoneNumberTransfer {
    mapping(string => address) allPhoneNumbers;
    mapping(string => address) requestedTransfers;
    address regulator;

    constructor() public {regulator = msg.sender;}
}
```

based on other basic technologies, the installation of smart contracts can look a bit differently. Depending on the platform selected, such as Hyperledger Fabric, for example - the identity of the regulator may also need to be explicitly communicated to the contract, with a one-time function call after installation. (an Ethereum implementation such as Parity behaves similarly to Hyperledger Fabric, much like Microsoft SQL Server behaves like MongoDB: both are manifestations of a technology concept - „database“ or „blockchain“ - but are interpreted and implemented in completely different ways).

The function of initially assigning a telephone number to a provider could be implemented as shown in Listing 2.

The `createAndAssignNumbers` function specified in this fragment receives two parameters: the telephone number and the identity of a mobile service provider (in the form of an Ethereum address, as described above) to which the telephone number is to be assigned. The code first checks whether the sender of the transaction is the same as the regulator, because in our example, only the regulator has the authorization to initially create and assign a telephone number. The `require` function in Solidity leads to the equivalent of an *Exception*, such that if the condition is not met, the system immediately stops processing the rest of the transaction code and marks it as bad. Then by accessing `allPhoneNumbers[phone number]`, we check whether the desired telephone number has already been assigned. This access to a

mapping is defined in Solidity such that unfilled entries are returned as `0`. Otherwise, in the last line, we simply assign the address passed as a parameter to the transferred telephone number. Since in transaction processing in a block chain, an absolute time sequence of the individual transactions is fixed by their position in a block, we do not have to pay attention here to simultaneous write access by other transactions.

For the second transaction type - the desire to port a number - we proceed analogously. Such a call would be sent to the network by the new mobile operator and could be performed as shown in Listing 3.

Again, the code of the smart contract first checks the two entry conditions and, after a positive check, changes the data in the world state to record who had requested the porting of the given number. Typically, the last line of the Smart Contract would now have a notification event (for example, a *event* in solidity) to which each mobile operator is subscribed using its own program code in its blockchain node. Thus, the smart contract starts the local (blockchain-independent) processing of the porting in the existing backend systems of a participant as soon as another operator requests the porting of a number from its area of influence.

After the mobile service provider to which this number is currently assigned completes its internal check of the porting and has validated the correctness of the porting itself (e.g. through the informed consent of the customer by SMS), it can respond to the query with another transaction. The smart contract code of this transaction could in simplified form look like Listing 4.

You can design the rejection of a transfer in a similar manner. Here the difference lies in the fact that the assignment of the telephone number is not updated, but only the porting request will be removed from *requestedTransfers*. In either case, at the end of the function you would trigger an event again so that the affected counterpart is informed without having to constantly monitor every single transaction of the blockchain.

### Listing 2

```
contract PhoneNumberTransfer {

    /* ... as above ... */

    function createAndAssignNumber(uint phoneNumber, address to) public {
        require(regulator == msg.sender);
        require(allPhoneNumbers[phoneNumber] == 0);
        allPhoneNumbers[phoneNumber] = to;
    }
}
```

### Listing 3

```
function requestTransfer(uint phoneNumber) public {
    // Number must exist and be assigned
    require(allPhoneNumbers[phoneNumber] != 0);
    // Number may not already be assigned to the requestor
    require(allPhoneNumbers[phoneNumber] != msg.sender);
    // number may not already be in a porting process
    require(requestedTransfers[phoneNumber] == 0);

    requestedTransfers[phoneNumber] = msg.sender;
}
```

### Listing 4

```
function confirmTransfer(uint phoneNumber) public {
    // Make sure the number should be ported at all
    address destination = requestedTransfers[phoneNumber];
    require (destination != 0);

    // Number must currently be assigned to the sender of the transaction
    require (allPhoneNumbers[phoneNumber] == msg.sender);

    // Update the assignment for the number
    allPhoneNumbers [phoneNumber] = destination;
    // Delete the open transfer
    delete requestedTransfers[phoneNumber];
}
```

## Why exactly is it all secure?

So far so good. But what about the security of this approach? What would happen if one of the participants simply exchanged the smart contract on the blockchain node controlled by him or her and used a variant that was more advantageous to the participant? (Incidentally, depending on the blockchain, the attacker would have to dive relatively deep into the internals of the blockchain platform implementation - but it is still a valid attack vector, which we must at least consider on a theoretical level.)

In our example, an attacker could try to take control of all phone numbers by changing the test conditions in his or her version of the smart contract. But that would mean that the world state is changed only on the node of the attacker: The attacker is the only participant that has the impression of having control over all phone numbers. All other participants would continue to execute the correct version of the contract. If, due to the chosen consensus mechanism between the participants, the attacker is selected by the network itself to generate new blocks, they would immediately be rejected by the other participants. The attacker would lose the status of the block generator again because depending on the configuration of the blockchain, half or two-thirds of all participants must agree to the outcome of a transaction before it is considered correct across the network. Any attack attempt therefore hurts the attacker directly.

It is therefore possible to compare this attack with a situation in which a bank customer crosses out the balance on his or her printed bank statement and writes a higher total. From his or her viewpoint, there is much more money in the account than before. But as soon as this customer tries to access this money, he or she is very quickly brought back to reality: their own view is nothing but wishful thinking, which will be acknowledged by counterparts at best with a chuckle.

### Listing 5

```
// Contract object is generated on the basis of the ABI and an example
// destination address
let abi = JSON.parse(fs.readFileSync("demoContract.abi"));
let address = "0xFE363D7030Db8E93517bB6315165B68AC4387DE8";
let demoContract = new web3.eth.Contract(abi, address);

// In the background, the method call is translated into an Ethereum
// transaction, signed with a configured (and explicitly released in advance)
// default identity, and sent to the network.
let numberToTransfer = "491511234567";
demoContract.methods.requestTransfer(numberToTransfer)
    .send({/* Optionen */})
    .then(function (receipt) {
        // ... further processing
    });
```

## How is a smart contract called up?

Having seen how a smart contract is defined technically and knowing that it can trigger events that external (non-blockchain) systems can handle, one last piece of the puzzle is missing: How is the smart contract actually called up?

There are corresponding SDKs in JavaScript, .NET and/or Java for most blockchain platforms. These SDKs allow us to create, sign, and submit transactions to our own nodes or third-party nodes of the blockchain network.

For Ethereum derivatives, a smart contract is called as follows: First, the method name and method arguments are converted to a defined binary format. To do this, helper methods are provided by the SDK that can process the interface description (ABI - Application Binary Interface) of the smart contract and generate the corresponding binary data. Thereafter, a transaction is created using the address created in the instantiation of the smart contract described above as the destination address. This entire operation can either be performed explicitly by the client code in individual steps, or - as shown in Listing 5 - it can look like a normal function call through automatically generated methods. In the listing, we see that we are using the relevant part of the call of the *request transfer* contract method presented above, the *web3.js* [4] - the JavaScript SDK for Ethereum.

## Smart contracts and cryptocurrencies

We've seen that smart contracts are simply a way to run program code distributed across a network to achieve the data integrity guaranteed by each blockchain platform. But how does that fit with the hype in the IoT area (such as automatic cold chain monitoring with penalty payments mentioned in the introduction)? In such cases, smart contracts consistently take on management of the monetary units (cryptocurrencies). For example, a refrigerated container to be transported from Singapore to Amsterdam could be instantiated with a smart contract that implements the monitoring of the maximum allo-

### SESSION Implementing Ethereum Smart Contracts with the Proof of Authority Algorithm on Azure



#### PHILIPP PENDELIN

In this live coding session, Philipp is going to show how a private Ethereum blockchain based on the proof of authority algorithm can be set up and managed in the Microsoft Azure cloud. The aim is to get a feeling how smart contracts can be implemented and deployed and why the classical proof of work algorithm is playing just a secondary role in such scenarios.

wable temperature fluctuation range (guaranteed by the logistics company) and at the same time an automatic penalty payment system in case of deviations. The contract could be designed such that an IoT sensor reports the temperature to the smart contract hourly during the entire transport time and a deviation from the agreed temperature range immediately provides for another Blockchain transaction to transfer an agreement currency as a penalty payment. Blockchain networks which manage the automatic handling of decentralized electricity deliveries, also have similar combinations of IoT sensors with smart contracts in the background. (basically, the term „smart contract“ is based on such concrete preliminary agreements, which do not require any manual or even legal tracking and are implemented in a purely technical and automated manner.)

The really interesting and new thing about blockchains is that they allow us to create completely decentralized structures. Network participants no longer have to agree on a central, trustworthy site, but only on the content of the smart contracts. These then run in the blockchain network in a decentralized manner. The hope is that not only in the logistics sector, but in all market sectors, market participants will be able to

implement digitization processes without restricting their autonomy by having to choose a central processing body (possibly a pool of competitors from the same sector).

And personally, I think *that* is extremely exciting.



**Ingo Rammer** is one of the founders and managing directors of Thinktecture AG and has been assisting developers in the use of new technologies for nearly twenty years. His focus is the hype-free use of blockchain technologies in the B2B environment, especially in private networks based on Hyperledger Fabric or Parity.

---

## Links & literature

---

- [1] <https://www.hyperledger.org/projects/fabric>
- [2] <https://paritytech.io/ethereum/>
- [3] <https://remix.ethereum.org/>
- [4] <https://github.com/ethereum/web3.js/>

Interview with Ricardo J. Méndez, Technical Director  
at Samsung NEXT

# “Decentralization is about freedom, flexibility, and choice. Blockchain is just one more tool”

Decentralization is about freedom, flexibility, and choice and blockchain is just one more tool; one that can help in decentralized contexts, but a tool is always less important than the goal. We caught up with Ricardo J. Méndez, Technical Director at Samsung NEXT to discuss the importance of decentralization, his blockchain predictions for 2019 and the decentralized technologies that are gaining ground right now.

by Gabriela Motroc

---

**At last year’s Blockchain Technology Conference, one of the keynote speakers mentioned that decentralization is more important than blockchain. Do you agree with this statement?**

Ricardo J. Méndez: Without question. Blockchain is just one more tool, and one that can help in decentralized contexts, but a tool is always less important than the goal.

**What does decentralization mean to you?**

Ricardo J. Méndez: Decentralization is about freedom, flexibility, and choice. Technically, it’s about moving activities away from a central controller and

towards a network of participants that interact based on a shared protocol. In doing so, it shifts the balance of power back towards the user, giving them a choice between providers, and allowing a plethora of approaches and perspectives to emerge.

For example, E-mail is a decentralized system and we can all contact each other. We don’t all need to use the same e-mail provider, or even the same base software, as long as every server behaves in a standard way.

Imagine most people used “X-mail”, where you can only contact other friends who also use X-mail. Any such platform would have the power to decide what gets discussed and what you get to see. It would be able to monitor everything that happens, and do so blatantly, as users would be reluctant to leave because otherwise, they can’t keep in touch with their friends. And what

## *“Blockchains and distributed ledgers are just a tool and like any tool, they have trade-offs”*

would happen to those people that X-mail decides to exclude from the system or those who don't wish to be monitored?

On a decentralized system, the user has a choice.

### **What are your blockchain predictions for 2019?**

Ricardo J. Méndez: Heh, there's that line attributed to multiple people that “Making predictions is very hard, especially about the future”. It's hard to say this early in the year, but I would say that blockchain will become less and less of a magic word and that the ecosystem is likely to get “saner”.

The runaway cryptocurrency prices in the last couple of years brought the idea of blockchain into the mainstream. This also brought in a lot of “hustlers” – i.e. people more interested in cashing in quickly than solving a problem. It became a magic word for people to short-circuit common sense and raise massive amounts.

I'd predict that these speculators and opportunists are going start leaving for greener pastures, which will reduce the amount of noise in the ecosystem. The teams that remain will need a sane focus on building working software and actual use cases.

### **Is blockchain the land of milk and honey? Can it really offer an exponential benefit above other existing technologies?**

Ricardo J. Méndez: There ain't no such thing as a free lunch.

Blockchains and distributed ledgers are just a tool, and like any tool, they have trade-offs.

Relational databases offered amazing benefits compared to older systems, but you won't find anyone suggesting that you use PostgreSQL to store absolutely everything, including documents and vacation photos.

If the problem you are solving fits a distributed ledger's particular characteristics, then they will be a huge help, but anyone advising that you need to replace every system with a blockchain is trying to sell you a lot of consulting time.

### **INTERVIEWED IN THIS ARTICLE**



Ricardo J. Méndez is a software engineer, privacy advocate and speaker. He has worked all over the industry – from building systems for large financial institutions, starting a small game studio and running a number of startups. He is also the Technical Director at Samsung NEXT – Samsung's investment and software products arm, which invests in innovative technology startups.

### **What other decentralized technologies are gaining ground as we speak?**

Ricardo J. Méndez: There are three particularly interesting technologies that have been gaining a lot of traction:

Peer-to-peer web: best showcased with the Dat Project, HyperDrive and Beaker Browser, p2p web allows any machine to become a server or mirror a website. The Dat Protocol was originally created for sharing scientific data, but it has since been used for everything from censorship-resistant web hosting to peer-to-peer maps.

Federation via ActivityPub: Federation is not a new concept – e-mail is actually a federated system. ActivityPub is a protocol for federation which allows, effectively, a decentralized social network to emerge and flourish. This network can be heterogeneous, built out of multiple platforms, but anyone can share content that users on other systems can consume in a standard manner. Its best-known implementor is Mastodon (which you can think of as decentralized Twitter), but it's also supported by multiple other systems which can all communicate transparently through it.

Zero-knowledge proofs: ZKPs are a method to demonstrate that you know a value or hold some information, without having to reveal the information itself. They are more commonly associated with cryptocurrencies, as they came into public attention with Zcash, but have implications in decentralization beyond blockchains. A decentralized system is one in which there might be no trusted intermediaries, and ZKPs can allow you to still transact and provide positive identification while retaining your privacy.

### **Is decentralization a prerequisite for privacy?**

Ricardo J. Méndez: Absolutely. I think there can be no privacy when all the information is flowing through a centralized system. Even if most of it were encrypted, it'd be sensitive to metadata analysis by the central overseer. Just tracking who communicates with whom, when, and how often, reveals a lot.

It's just not a guarantee. You need only look at most cryptocurrencies, which are completely decentralized but also 100% public – their pseudonymity only needs to be broken once. Peer-to-peer approaches require you to broadcast your activity to peers, so they need an extra privacy layer (like a VPN or mix network).

This is why, as an industry, we need to get better at explaining to users the trade-offs of different approaches.

**Thank you!**

What are the possibilities?

# Blockchain and Web 3.0: The future of private data protection

What solutions become possible with the integration of blockchain technology into the Web 3.0 features? In this article, Premjith Purushotham shares his thoughts on what the combination of those technologies can offer to private data protection.

by [Premjith Purushotham](#)

---

The internet is today an indispensable resource for the whole of humanity. Even in the least developed locations of the world, some form of internet-based utility does exist, simplifying human lives there. The internet is driving many powerful technologies. And with power comes responsibility. Currently, we are sailing through the Web 2.0 tide in online technology development. Web 2.0 implies the present internet era of social media led connectivity and enhanced communications. The most significant question in the Web 2.0 era is the security of one's data.

We know that “time is money”, but in this era of internet “data is more money.” There is a particular reason why hackers are targeting, big companies like Apple, Amazon or Facebook. The idea is that these few

giant companies virtually hold the majority of data on the internet. Your search engines control and store the data you created through searches on its data centers. Similarly, social media companies like Facebook has its data centers located in remote locations which it maintains at a significant cost. The iCloud hacks are now so frequent that it no longer arouses the same curiosity as it used to.

Recently, Facebook was caught again at data collection using its free VPN service. The coming of new technologies like Big Data and AI has further made it clear that data is going to be the petrol for the future. The more data a company has, the more it can energize its strategies to expand the business and reach consumers. Hence, the new technologies have played us right into the hands of big companies. But as we say, “technology solves the problems technology creates.” The solution



## *A Web 3.0 will be a self-learning internet. It will be like a personal assistant, walking with you.*

comes from the possibility of integrating blockchain technology into the Web 3.0 features.

Blockchain is a reality. Its logic is simple and straightforward. A blockchain is just like a public ledger in which information is spread over different registers. It has three aspects: data, hash and cascading interlink. In a blockchain the first block that stores data is called a genesis block. It stores data and has a hash value assigned to it at the instance of data entry. It means that the hash generated is an encrypted form of the data entered and to change the data now a person will have a hash change as well, making it impossible to move without leaving a mark.

Now, another feature of blockchain is that it is scattered across millions of small servers. Unlike, a tech giant's large data center, the localization of data makes it difficult to get data in a lump sum for an attacker. Imagine storing a million dollars in small purses with 1 dollar bill in each wallet. It will drive the robber insane by the time he gets the whole. It is what blockchain does. Cryptocurrencies are already employing it.

Web 3.0 is the next big thing on the anvil. Experts are speculating that it is already happening. As far as one can gather from the concepts, a Web 3.0 will be a self-learning internet. It will be like a personal assistant, walking with you. For example, you want to know whether you can get to the office in time through today's traffic. You ask the browser "how's the traffic today?", the browser having a knowledge of the time of questioning and how often you have asked and based on your profile, will not only bring out the traf-

fic conditions and will give you the "best route to the office today."

The development of semantic technologies can further make Web3.0 intelligent. But to make it the instrument of data protection, it is essential to base this technology on the blockchain logic. The real blockchain logic will ensure that we carry our data as locally as possible. The peer to peer review system will keep the users alert of the data block building. However, some of the challenges in the integration of these technologies are:

- How to locate and build small servers?
- How the protocols can be redefined to enable the data access
- How to regulate malicious contents on terrorism, national security etc.
- How to share costs of building servers between individuals, government and private entities, without compromising data ownership?

While many of the challenges are at the political level, some of them have real technical constraints. For instance, how will two different blockchains interact with each other with localized data? Scalability and interoperability still need a technological alternative. In spite of the limitations blockchain and Web3.0 opens up the first real possibility of protecting data. The future looks prospective with strides in blockchain development.

### SHORT TALK **The Compliance Issues with GDPR and Blockchain Technology**



**SZABOLCS HARGITTAY**

The content of the presentation will cover the following topics:

- Introduction: GDPR and blockchain terminology
- What are the commonalities and conflicts?
- Potential Solutions: technical level
- What is the future? How can we regulate and harmonize the blockchain/GDPR world?

In my presentation I would like to show (i) the common features, (ii) the conflicts in GDPR and blockchain technology, (iii) the potential solutions and last but not least (iv) the regulatory framework.



**Premjith Purushotham** leads the Digital Marketing team at Aufait Technologies, a top-notch DMS software provider in India. He also heads the SEO team at Mindster, a frontier mobile app development company in India. With his 4 valuable years of experience in online marketing, he helps clients expand their online presence and mushroom novel business ideas.

Interview with Brian Behlendorf, Executive Director of Hyperledger

# “2019 will be a continued reiteration of how people are using blockchain”

In this interview, we speak with Brian Behlendorf about what the future holds for blockchain, how certifications are changing the enterprise and the goals of Hyperledger in 2019.

**Hyperledger really got its foot in the cloud door now that Fabric has the attention of all Cloud providers. What does this say about the maturity of Hyperledger and its commitment to the Cloud?**

**Brian Behlendorf:** The project is just barely three years old – it first launched in December 2015. When it was launched, the code was still very research and development oriented. It was still very prototypical, very early stage. There was a clear message that this code would take some time to settle. We thought, let’s try to deploy it for some things and see what happens, but it was tagged with a “not safe for production use” at the very beginning.

We were still figuring out a lot of questions like, “How do we grow this community? How large of a community

should we try to be? What standards do we want to follow? A lot of that congealed over the first year. Then, Fabric was our first production 1.0 release in the middle of 2017 – a year and a half in.

Now, most people don’t use a 1.0 product. You wouldn’t want to use Windows 1.0 or Linux 1.0. But for us, 1.0 says, “Here’s a big software that now developers are comfortable with people they’ve never met and they feel in control using this software in production. That doesn’t mean it’s done or perfect or bug-free, but it means there’s a level of confidence in that. There’s other vetting that we do. We have a third party security firm that comes in and scans the code for security vulnerabilities. We know you can’t catch them all, but we catch the low-hanging fruit.

*There's more room inside the Hyperledger greenhouse for many more projects. We want diversity at each level too. That's why we have competing frameworks. It creates a bit of rivalry, but also a lot of collaboration.*

With the 1.0 release, from that point forward, a lot of people started to use it and get it into production. Then you start to realize the next order of challenges, which is: “How do you make this easier to adopt or for somebody to turn on?” Especially since rarely in a blockchain application do you have only one company involved. Usually, you’ve got three companies, ten companies, a hundred companies. So, getting everybody to adopt a new technology is difficult. Ordinarily, it’s hard to get even one company to adopt one new technology. But to get an entire industry, or even a bootstrap number, you have to meet companies where they are. If it’s three banks starting a network, one might be very leading edge, but the other two might be more trailing edge.

And so, cloud adoption and cloud support for these technologies is incredibly important. It has to be easy for overworked, underpaid IT staff who don’t have the capacity to become experts in a new emerging technology to be able to stand up for their company’s interests. Many of these clouds including Amazon and Google, they’re hearing from their customers: “Do you support running Fabric?” Because they sell their access to virtual hardware essentially, customers could always run versions of Fabric on AWS. But now all those companies and all of the cloud providers now have branded, ported, first-class services around Fabric or Sawtooth. That’s a sign that our technologies are not just ready for early adopters, they’re ready for the broader market as well.

When somebody goes and builds a network for a banking sector, or a trading sector, or a supply chain, everybody can get on board, not just the early adopters.

**In your opening keynote at Hyperledger Global Forum, you mentioned the new library: Hyperledger Ursa. Will new libraries come in the future? Will they have their own group apart from tools?**

**Brian Behlendorf:** We will look at that over time. Our goal is not to have a thousand projects, our goal is to have a portfolio that’s well-curated and that our projects work well with each other. There are upper limits to how many people you can have in a community before it gets too unwieldy to manage. But I could see a few more wings of the greenhouse. Maybe a wing dedicated to libraries or templates. We are probably not going to be building any end-user apps. I don’t think we really want to get into being a home for applications that only barely touch blockchain as a way to store and retrieve data. I would love to see projects outside of Hyperledger.

What’s in store I think are libraries, templates, and frameworks that make it easier for repeating patterns.

For supply chain, traceability is a pattern that applies to lots of different sectors. Marketplaces and directories might be another pattern. Put all of these things together, and there’s more room inside the Hyperledger greenhouse for many more projects. We want diversity at each level too. That’s why we have competing frameworks. It creates a bit of rivalry, but also a lot of collaboration.

**You also mentioned certifications in your keynote. What are the benefits of passing the exam? What can these certifications do for Hyperledger?**

**Brian Behlendorf:** Certifications provide a professionalization of the space. We are helping people to understand that there is enough substance here to what we are building and enough that’s different. If you’re serious about building this into your core information system, you want to know that if you’re thinking of hiring someone and they claim to know how Hyperledger Fabric works that their claim is trustworthy. The training and test are intended to help convey that trust.

This is all about the mid-to-late adopters on board. The early adopters have their own process for vetting developers or are perhaps willing to pay for someone to receive training for a few months. Mid-to-late adopters need someone off the shelf who someone else has said that they are trustworthy and knowledgeable. In some cases, those late adopters will simply use cloud or cloud offerings and not have any staff. But there will be some that realize that they need a few staff with a direct connection who can audit and validate what’s going on.

My hope is that by being able to point to the thousands of developers who have that certification, it says that this is now a safer technology. Now there is a resiliency to the system. The certification for administrator of these technologies is what we are launching first. We will later look at certifications for other things, such as developer or for businesses offering blockchain as a service. How do you know that your instances or use of Fabric on one cloud provider is going to be the same as another, or that you can move it or add nodes for extra resiliency? All of that is important. The broader theme is professionalization.

We’re not doing this as a revenue generator or to add new members. This is in our mission. It’s what we are here to do. This is what The Linux Foundation does in other places. Here, the Cloud Native Computing Foundation (home of Kubernetes) does exactly the same thing. They have a Kubernetes certified administrator course and certification exam, as well as Kubernetes certified

*By being able to point to the thousands of developers who have that certification, it says that this is now a safer technology. Now there is a resiliency to the system.*

providers and solutions providers. This is something that modern IT enterprises expect from a mature technology base.

So, how do we help blockchain technology go from this early Cambrian explosion of a lot of different ideas into something that's a bit more predictable? I've been accused of making blockchain boring, and I'll take that!

**There is still a talent shortage, will the certifications help?**

**Brian Behlendorf:** It certainly will help. We need more of that, for sure. Here is where you will learn how to go to the community to get help. It will teach you how to support yourself. Certificates will also teach you if you find a bug and how to fix it, how to create a pull request. It is equally important to help the broader marketplace as well as specialists.

We want to show people that our technology is usable and that there are people out there who can help get it set up.

**Are you planning on reaching out to developers beyond the blockchain world? If so, how?**

**Brian Behlendorf:** I've been going to the Ethereum conferences each year without a marketing presence just to build personal connections and bring some of them closer. With other blockchain communities, I am letting them know how we work together. But I also spend a lot of time at other developer conferences, such as some of the Cloud and open source conferences by Linux. I give talks at those and help people who are all in very different places to see how our code works.

We want to do more outreach. Our marketing department wants to explain what this code is to other people. We do things such as have developer profiles on our website, as well as an active blog that we ask developers to write for. All of this hopes to encapsulate what is going on here into more digestible chunks that find their way well beyond the blockchain.

**There are still some concerns surrounding blockchain. Scalability, for example, is one of them. There are a lot of questions about blockchain that haven't been answered yet. What does it take for people to be confident in blockchain?**

**Brian Behlendorf:** Normalization. We need more people working with this code and more stories about how people use it. That's why it was great to have Aaron Symanski from Change Healthcare put a number down: 50 million transactions a day. That's a signal to everybody else that if they design it right and do the

right things they can build a system that can meet that number. Those numbers aren't everything. There are a lot of different questions to answer about what the scalability actually needs in a blockchain network. It's not just how many transactions you have, it's also your resilience to security issues and the number of nodes on the network. It's a little counter-intuitive, but the more nodes you have the slower the network gets.

There's something to the art and science of blockchain design and application design at that layer that is probably going to be as unique as database design emerged after the SQL programming language came out. Once people started to work with relational databases, it became clear that there was quite a lot of architectural thinking about the app you want to build and how you model it in the database. Early databases were slow and you had to think about every byte and how they align. You had to think about the physical format of the hard drive to get the performance you needed when CPUs were much slower. If you're a big database architect, you are constantly thinking about this. We are going to see a similar field of blockchain architecture. How do you squeeze the most performance while meeting all the business level objectives?

In the early days of the web, and even in the middle of the last decade, it was still the case that if you ran a website with a healthy amount of traffic you'd fall over. Sometimes you would go visit a site to see something really interesting, and it would often be down or not responding. It still happens here or there, but now we have things like content delivery networks and businesses to answer the question of, "What happens if I have too much of a good thing?" In the blockchain space, part of that architecture is how do we adapt what we're doing when suddenly there's ten times the amount of interest and traffic. I don't think there's ever a singular answer to the scalability problem.

How do we build skill sets that will have us answer the question: "How do I answer the next higher threshold of traffic?" It's a mix of technology and business to answer that question.

**Some people say that blockchain is the world's slowest database right now.**

**Brian Behlendorf:** Yes, and it's not a singular slow database. But this is where we really need to get crisp about public blockchains and permissions. If you're talking about Bitcoin or Ethereum, they can only do three or ten transactions a second. Even if they did three or ten million transactions a second, there would still be a really good reason not to run most of your transactions on the public ledger. Why make yourself subject to the

*Don't use a blockchain because it's a big data pool or because it's fast. Only use a blockchain if you have an issue of trust. Then it is worth paying the penalty.*

activity of people you have no relationship with? It is definitely true, but you can always use a centralized database to solve any problem. It will always be faster and cheaper to do it that way.

Don't use a blockchain because it's a big data pool or because it's fast. Only use a blockchain if you have an issue of trust. Then it is worth paying the penalty. There's always a good question of bringing down the cost of that penalty. The more we bring it down, the more places it'll be worth using. But there will always be a penalty over centralized and that's because there's value in decentralization.

#### **What are your predictions for 2019 and what's your New Year's resolution for Hyperledger?**

Brian Behlendorf: I think it will become clear to people that blockchains aren't just about cryptocurrencies and that you can solve many of these problems that people are hoping to solve. We've been consistent on the message of where and how to build blockchains. Our signal has been growing and growing. The hype cycle of cryptocurrencies is arguably on a bit of decline and people are asking harder questions. I don't think cryptocurrencies will ever go away, but I do think that this is the time that people see how they can use these technologies without having to worry about the market prices. 2019

will be a continued reiteration of how people are using blockchain and asking the harder questions about the value that they are getting from it.

Within Hyperledger, our resolution is about professionalization or normalization. It is a balancing act between enterprise software that doesn't change a lot that people can trust, that gets better and smarter but still has room for disruptive new thinking.

My only New Year's resolution is travel a lot less! I was on the road more than half of every day this year and circled a world a couple of times. I'd like to only circle it once or twice.

#### **Thank you!**



**Brian Behlendorf** is the Executive Director of Hyperledger Project. Behlendorf was a primary developer of the Apache Web server, the most popular web server software on the Internet, and a founding member of the Apache Software Foundation. He has also served on the board of the Mozilla Foundation since 2003 and the Electronic Frontier Foundation since 2013. He was the founding CTO of CollabNet and CTO of the World Economic Forum. Most recently, Behlendorf was a managing director at Mithril Capital Management LLC, a global technology investment firm. Follow him on Twitter @brianbehendorf.

Building the next blockchain revolution

# Why the development of next-generation blockchain platforms must be led by the community

Blockchain is a communal effort; why would building the next-gen blockchain platform be any different? In this article, Vladislav Dramaliev explains why community input is imperative for the future of blockchain and the key to mainstream adoption.

by Vladislav Dramaliev

---

“There is no power for change greater than a community discovering what it cares about.” — Margaret J. Wheatley

Every truly great movement begins when a group of people identify a problem, engage in dialogue and deliberation, and then act together to solve it. Ten years on from the launch of Bitcoin, it is by following this process that the most innovative blockchain projects will achieve mainstream adoption.

Third generation blockchain platforms aim to improve upon Bitcoin’s innovative technology in order to make it more commercially attractive. However, they are faced with numerous challenges. Instead of solving these challenges by relying on a single team, sacrificing decentralization, and creating a single point of failure, the “blockchain way” to address them is to present them to a global community of bright minds. This makes community-building in the public blockchain sphere an essential component of project development. A strong and motivated community will be an important vehicle for the transition to mainstream adoption.

## Adapt and achieve adoption

Community has always been extremely valuable in driving the success of blockchain projects, from contributing ideas and offering feedback, to hands-on development work and spreading of the project's message. Community has never been as important as it is right now, as everyone in the blockchain industry is finding unique ways to enable mainstream adoption of the technology.

To successfully achieve mass adoption, it is first necessary to adapt the technology to the needs and desires of those who would use it. This means making it scalable and user-friendly and then creating unique incentives for users to use it. These will be best achieved by engaging a diverse, global community of developers, UX/UI designers, entrepreneurs and users. They should be able to express their visions for the future of a public blockchain platform with tangible means – tokens, apps, and code. Social media and digital content are becoming less relevant as vehicles of expression in a world where, thanks to blockchain technology, “putting your money/code/app where your mouth is” will become the default approach to support an opinion.

Blockchain 3.0 is about adapting to achieve sustainable adoption. Adapting existing technology to fit the needs of a diverse community is an important first step.

### SHORT TALK **Blockchain-based Arbitration: an adequate Forum for the Resolution of international Commercial Disputes?**



**SARA HOURANI**

A smart contract is a software programme that is stored on the blockchain. Smart contracts can be used in supply chain management, trade finance and energy trade. In the context of these different transactions, it would be relevant to include an arbitration clause in the smart contract to prompt the parties to resolve their differences with regards to the performance of the contract. This type of dispute resolution has received widespread attention for the resolution of specific low-value claims.

In light of these technological innovations, this paper focuses on an examination of the extent to which the blockchain-based arbitration procedure guarantees the parties' right of access to justice in B2B disputes as an alternative to traditional face-to-face arbitration. The paper explores in the first instance the reasons behind the adoption of blockchain-based arbitration, and whether this ameliorates the parties' possibility of access to justice in contrast to traditional arbitration. The paper then analyses the extent that the parties' rights are protected by the current legal framework for providing adequate access to justice to resolve their disputes.

## Community correctors

Encouraging community-led development can be a difficult task, but is absolutely invaluable to highlight and correct issues that arise with the technology. There are a number of ways to encourage community members to get involved.

First and foremost, it is absolutely essential to make the technology accessible to anyone who is interested in improving it or building on it. Having an open-source codebase that is free to access but that no one can understand is not helpful at all. To address this, documentation, guides, and tutorials must be developed and remain updated as time progresses. These can also be prepared and updated collaboratively, involving experienced community members and the team that developed the code originally. Currently, at æternity, both the developer and æpp teams are working on the foundations of these resources. The idea is to make them almost completely community-driven.

Second, it is important to make the community interact with the code and apps as frequently as possible. This could be done through code or app-related challenges or online hackathons. Incentives could vary, but tokens are usually the most preferred option. Distributing tokens to community members that have created value in the form of improved efficiency, functionality, or usability is a great way to build a healthy community.

Third, hosting meetups and empowering ambassadors are simple and effective ways to spread the word about your blockchain project and get users interested in the philosophy and technology behind it. Presenting engaging opportunities to individuals to become part of the community by contributing ideas, code, or any other kind of useful effort can also be effective.

Finally, bounties have proven to be an important community-building tool in the digital world. From bug reporting/fixing to creative contests and app ideas, bounties usually provide token-based incentives to a diverse, global crowd to do what they love and become part of a community.

One of the fundamental ideas behind Bitcoin and public blockchain projects is to identify and put to the test new methods of generating consensus on a global scale. After all, central to the mission of the technology is the empowerment of micro and macro-value creators around the world. Especially the ones who are unable to realize their full potential due to limitations imposed by major concentrations of wealth and power existing in the world today. In order to manifest this change, thorough consideration must be given to those who will ultimately use, improve, and benefit from the technology.

With new powers come new responsibilities. Cryptocurrencies and tokens are confirming the validity of this statement. Existing governance models are losing their attractiveness to citizens who are fed up with waiting years to provide feedback (voting cycles), inefficiencies, corruption, and lack of accountability. By providing modern tools that enable users to participate in the

governance process directly and in real-time, one can ensure that project development will be led by a community in a transparent manner.

æternity's governance mechanism is implemented via delegated voting, weighted by the number of tokens each account holds. It is a form of direct, liquid democracy that will give the community a voice on any relevant topic, including platform development. The æternity community will be able to formally provide its opinion on any topic using the native AE token and an easy-to-use mobile application. This is the future of decentralized governance.

### Potential fulfilled

For blockchain to achieve its true potential, it needs to inspire value creation in any form imaginable – ideas, visual creativity, code, content, and actions. As we begin 2019 and reach a critical point for blockchain – widespread adoption – blockchain 3.0 needs to recognize that a core team can only take a project so far. What is needed is an active community with a stake in the project that is empowered by a system of communication, collaboration, and creativity. To achieve this, it is imperative that documentation, guides, and tutorials are develop-

ped, while various activities rewarding engagement and value creation are organized. User-friendly governance tools should be made available on mobile devices and complemented by a new generation of communication tools that are private, secure, and user-friendly.

In the ten years since blockchain's inception, we have seen several projects attempt to push their solution to the forefront and achieve mainstream adoption. Undeniably, the technology has achieved great success in this regard. But as the dawn of next generation blockchain is upon us, revolutionary ideas now need more than just conception, they need community leadership. Taking this approach will fulfill the goals that were laid out in the foundations of blockchain technology ten years ago and ensure that it reaches its full potential within the next ten. Or even sooner.



**Vladislav Dramaliev** is head of digital marketing with æternity. He is also Founder of BitHope.org, CryptoCrowd.org, and Bitcoin/Blockchain meetup in Sofia, Bulgaria. Co-Founder of the Bulgarian Bitcoin Association and the first website for bitcoin exchange in the country. Fascinated by technology, Bitcoin, blockchains and the future. Member of MENSA.



## Blockchain in 2019 and beyond

# Looking forward to a golden future for the blockchain industry

Blockchain's growth cannot be understated. But, are we currently in the golden age of blockchain, or is the best yet to come? Take a speculative look at what the golden future of the blockchain industry holds and what industries it will change.

by Dr. Demetrios Zamboglou

Since the turn of the century, an entirely new phenomenon has arrived to improve the lives of consumers and businesses alike. Still in its infancy, blockchain technology is developing at seemingly breakneck speeds to solve many of the as yet unresolved problems in modern business. As an early sign of what's to come – and also to wet the appetites of consumers globally – cryptocurrencies have emerged to deliver a kaleidoscope of benefits to users.

Faster payments, better accounting, secure storage, precise shipping – you name it and blockchain technology can deliver it.

The ongoing developments of the blockchain industry are delivering a variety of solutions, but still, many problems remain unsolved. One of the major challenges has been translating the blockchain prowess from the

highly technical into the understanding for the layman. In other words, how to make such poignant technology usable, functional and popular amongst hordes of people, rather than just for a few niche businesses.

From this perspective, cryptocurrencies, blockchains and distributed ledgers remain at the embryonic stage of development. However, 2019 promises to advance the gifts of this so-called new technological dawn into the stockings of the mainstream.

“The gap between what is relevant and what is people understand is so large, there is going to be a large lag time when new technologies come into the mainstream,” says Benjamin Cordes, from Macro Exchange.

“For Bitcoin this was about eight years, for Ethereum it was around four. The advance I am most excited about is improvements on the core economic protocols, namely the proof of work algorithm. And more generally for systems to become available which allow for a

decentral economy, democratising finance and access to markets,” says Mr Cordes.

### Blockchain to make its charge

Several blockchain projects stand ready to make a meaningful impact in 2019. With special regard for the mainstream consumer, one of the prime development focuses are cryptocurrencies and their future.

Some industry experts like Tim Draper argue (quite rightly) that for every growth step taken by the world’s leading spearhead cryptocurrency (Bitcoin) will likely mean reciprocal legislative measures are also undertaken to enable safe passage for its prior growth to continue.

“Identity and Know Your Customer (KYC) are certainly hot topics but focusing on the Blockchain as some sort of magic is misguided. It is like giving a 5-year-old a mobile phone without a network connection – after about a minute what was cool becomes useless, unless you add connectivity,” says Bradley Hall, Founder and Chairman of ICON Capital Reserve, a financial software company that’s currently advancing a rather unique crypto product that aims to secure investor interest in gold.

In terms of gold, a trusted store of value with a 6000 year legacy, I think Ray Dalio founder of Bridgewater Associates put it best when he said ‘Those who don’t allocate 10% of their portfolios to gold, don’t understand history, economics or probably both.’ It seems the future has arrived and it is getting a little more evenly distributed,” says Mr Hall.

The company’s flagship product is AUREALS – a fusion of Gold and the blockchain that insulates holders from currency, institutional and systemic risks.

“PC’s first created by IBM were ornaments until Microsoft offered them DOS and then changed the game entirely when they embedded Windows software on OEM devices. Sir Timothy John Berners-Lee created the ultimate geek club with the World Wide Web, but the internet was unleashed when Netscape re-imagined his browser and Google began to index the treasure trove of information,” says Mr Hall.

The same evolution is occurring within blockchain whereby a magnificent root invention is now being harnessed by various companies in a variety of ways, for the benefit of consumers.

### Not forgetting regulation

China is readying a draft of regulations concerning cryptocurrency and blockchain companies that are expected to come into effect in February this year.

Nick Szabo, one of the pioneers in blockchain, notes that more people will turn to cryptocurrencies. Thus, different approaches will occur given the decentralised nature of the entire concept, and also, to accommodate the variety of applications required by users.

For example, security tokens were offered through Security Token Offering, which is designed to protect

investor’s ownership rights. By taking possession of a particular token, the holder gets a certain amount of rights within the ecosystem and can help trade value within it. It also acts as a toll gateway in order to use certain functionalities of a particular system.

On a wider global scale, the first legitimate national cryptocurrency is set to be launched later this year, linked to a fiat currency from a G20 nation. Several industry experts concur that improving the image of cryptocurrencies will be one of the most challenging goals faced by blockchain and cryptocurrencies – possibly because of the inherent risk of loss when used as a speculative investment vehicle.

### Banking and cryptocurrencies, a match made in heaven?

According to Microsoft’s founder Bill Gates, “There will always be banking but not necessarily banks”, and it could well be the blockchain industry and cryptocurrencies that facilitate what may turn out to be Mr. Gates’ prophetic claim.

However, it’s not just the private sector that wants to get in on the action when it comes to blockchain and cryptocurrencies.

From an establishment perspective, it is expected that central banks might start to supplement their gold reserves with cryptocurrencies as a result of growing mistrust of foreign central banks and governments and the vulnerability of national gold reserves. Furthermore, it is expected that additional integration with different platforms will occur in terms of being able to pay for services. New types of cryptocurrency have the stage to make their ascent, like Stable Coins. Unlike Bitcoin, these coins are designed for price stability and to be largely insulated from often volatile market conditions.

According to Margot James, the British government is committing millions of pounds to fund blockchain projects in areas such as energy, voting and charity through Innovate UK and research councils. Solutions like Lightning Network, a Bitcoin protocol which makes small transactions possible using its native smart-contract scripting language, are also expected to continue their growth in 2019.

Silvio Schembri, Malta’s Junior Minister for Financial Services, declared that “2019 will see the materialisation of The Blockchain Island, firmly putting Malta at the epicenter of this industry”. One of the applications is “digital remote voting” that is believed to be widely adopted after the successful pilot program in West Virginia, USA.

Another variant is Hybrid – a blockchain that attempts to fuse the best parts of both private and public blockchain solutions, when, for instance, governments are not able to become entirely decentralised by using only public blockchains.

According to Mr Hall, “there is a bit of an arms race going on in financial services and in particular in payments where DLT based solutions like AUREALS are

disrupting incumbents, as the command and control hierarchies of the last 70 years begin to decay and implode under their own weight.”

## The private sector

Despite the rumblings coming from public institutions, governments and central banks, the private sector is where the major blockchain developments are set to find fertile ground in 2019.

“The next wave of innovation is being constructed with distributed ledger technology (DLT) but simple ledger entries that act like ‘claim checks with nothing to claim’ will revert to their intrinsic value of zero – as Voltaire noted all fiat currencies do and powerful replacements representing assets and cash-flows will take centre stage,” says Mr Hall.

In the private sector, an increasing number of companies are transforming their business models and operations to adapt blockchain into their existing business processes. One of the largest being deployed is by the world’s largest retailer, Walmart. The US company is currently developing and testing a blockchain system to streamline its sprawling global consumer market empire.

Smart contracts are also evolving. One example is ‘Ricardian contracts’, which unlike standard smart contracts, can be read as it uses human-readable text. Another system called Chainlink uses cryptography and a type of secure hardware called a ‘trusted enclave’ to securely feed data to smart contracts on the blockchain.

Other applications being developed are seeking to integrate blockchain with artificial intelligence (AI). According to Nick Dryden, 2019 is going to be a year which sees biometrics flourish, whereby personal data will further rely on such uniquely human traits like veins.

Further examples of blockchain technologies that are expected to be seen in 2019 include CyberLogitec’s soon-to-be-launched solution, dubbed ‘FREIGHT9’ and ‘OPUS9’ which promise to pave the way for a purely digital system for maritime businesses, eliminating the hassle of paper-based documentation for all shipping.

The Hedera Hashgraph Platform presented solutions using the virtual-voting consensus algorithm and asynchronous Byzantine fault tolerance (aBFT). The Energy Web Foundation is building a “blockchain of blockchains” that wants to let consumers sell the energy they generate at home in markets worldwide using the EWF open-source software application, thereby helping to create a confluence of digital links between existing energy trading platforms.

“The area we are working on at economic networks is algorithm trading, market making and open source finance. The current financial system is dominated by large institutions which are ineffective and very costly. The future of markets is a global one and driven by protocols. The Internet so far has been mostly a medium for communication, but we mean to repurpose it for use as a robust transaction network,” says Mr Cordes.

“If you currently book a flight online you will access two systems: the web for the website and a financial network in the background. The future will be a seamlessly integrated holistic network,” he adds.

According to the highly-respected publication MIT Technology Review, the year 2019 is when blockchain technology “finally becomes normal”. With so much happening at once in blockchain development all at once, in some respects, the gold rush is far from over but is only just beginning.

The result is that this coming year is set to deliver applicability in all sectors spanning the public and private divide – and possibly most satisfying for consumers – will work for the benefit of all market participants.

### SESSION Zero-knowledge Proofs: Concept and Principles



#### DR. STEPHAN VOLMER

One of the bigger trends in the blockchain universe is the need for privacy and confidentiality for transactions on public blockchains. That has given rise to the emergence to a technology known as zero-knowledge proofs (ZKP). A ZKP is a cryptographic method by which one party can prove to another party the possession of knowledge without the necessity of simply revealing it. That way a layer of confidentiality can be added to public blockchains by not disclosing the transaction history whilst providing full verifiability of their validity. ZKPs are one of the most powerful tools cryptographers have ever devised. However, they are difficult to understand. In this talk, we are going to

1. understand the essence of ZKPs by looking at intuitive examples without getting into the nitty-gritty maths
2. learn the underlying principles of ZKPs
3. become acquainted to the different variants of ZKPs
4. understand their privacy-preserving impact for the ecosystem of public blockchains



**Dr. Demetrios Zamboglou** is the Chief Operating Officer at ICON CAPITAL RESERVE SA. He is an award-winning Fintech Executive, Blockchain Expert, and ICO Advisor with a doctorate in Management Research. Zamboglou has held executive leadership roles at FXTM, zebraFX, and FOREX CLUB, specializing in issues including business development, strategy, risk management, market-making and compliance. He is a member of the following groups: Institute for Securities & Investment (CISI), the Institute of Directors (IoD), and the Behavioural Finance Working Group (QMUL). Zamboglou also serves on the Postgraduate and Research Committee at King’s College London.

## The future of manufacturing

# IoT and blockchain are ready to drive a manufacturing revolution

In this article, Ilya Pupko, chief architect at Jitterbit, explains how the convergence of IoT and blockchain can transform an entire industry. How can these two pieces of technology work together in manufacturing to help streamline workloads and provide a better view of assembly processes? The digital revolution is here.

by Ilya Pupko

The convergence of two technology trends is set to transform the manufacturing industry. The Internet of Things (IoT) is entering the mainstream, with sensors and related technology becoming affordable for almost anyone. At the same time, we are seeing blockchain technology mature to the point where it can be relied on for a variety of enterprise use cases. By combining these technologies, manufacturers have an opportunity to revolutionize the way products are assembled and distributed.

IoT gives a voice to products as they make their way through assembly and the supply chain. Internet connected-sensors turn a “dumb” component into a smart device that the manufacturer can monitor and interact with as it makes its way through assembly and the supply chain. Giving the device a voice enables manufacturers to streamline and automate a range of activities that would normally be handled through time-consuming, manual processes.

### The revolutionary potential of IoT data

First of all, digitally enabled devices can provide manufacturers and partners with a clear view of their supply and assembly processes in real time – which makes it possible to coordinate and plan with much greater precision. In addition, connected devices can automatically alert the manufacturer to potential defects or problems during the assembly process that would otherwise have only surfaced during inspection or a breakdown later in the process. What’s more, the manufacturers can now communicate back to the internet-connected products in order to automate certain activities that previously required direct human intervention.

While these potential benefits have been technologically available for some time already, the trend is only entering the mainstream right now because the costs have fallen to a level where it makes sense for most manufacturers to leverage IoT technology. When adding internet connectivity to a product meant spending two to five times more, manufacturers had to carefully consider whether the benefits would be worthwhile. But as

the price of IoT components has come down, the decision has become a “no-brainer” for most manufacturers. And as more manufacturers adopt IoT technology, they prove its value for a wider range of use cases that others can follow.

### The importance of a secure and trusted network

However, internet-connected sensors are only half the equation when it comes to deploying IoT technology. On the other side, manufacturers need a secure network that enables them to communicate with the devices while preventing interference from hackers or other malicious actors. This is where IoT and blockchain technology converge.

Security has become a major challenge for internet-connected devices because their very connectivity has opened them up to a wider range of attacks and manipulations. This is where blockchain’s distributed ledger technology provides a crucial foundation for IoT devices to share reliable data with a range of users. Because the data is tracked on a decentralized platform instead of a single database, it becomes much harder for any individual user to manipulate. And this level of reliability is built into blockchain technology, which allows manufacturers to avoid the expense of additional protection and oversight.

The decentralized aspect of blockchain technology also makes it an ideal foundation for sharing information between manufacturers, partners and even end customers. The fact that the ledger is distributed adds a layer of trust that direct reporting cannot match. Instead of being forced to trust reports from one supplier or factory, you now have dozens or hundreds of other parties telling you the data is reliable, which allows manufacturers to operate with a much greater degree of confi-

dence. This trust can also be shared with partners and customers to add a higher level of service.

### Integrating with the Internet of Things

With internet-connected devices and blockchain technology becoming widely accessible to manufacturers, there is still one more hurdle to overcome. Blockchain on its own is just the technological backbone: enterprises need a way to connect to that technology in order to interact and get value from it. This means integrating the blockchain technology with the applications and platforms enterprises already use – such as Salesforce or SAP – to get access to the data in a format that it can use. Building integrations with existing applications gives manufacturers, and their partners and customers, a simple and easy way to access IoT data from the blockchain without the hassle of adding another system interface (let alone building one).

Adding IoT technology, and using blockchain to share the data, will soon become as common for manufacturers as automated conveyor belts or basic process optimizations. The potential benefits are so broad, and the relative costs so low, that only a small portion of manufacturers would decide the technology is not worthwhile. In the not-so-distant-future, buying a product that isn’t digitally connected will become an increasingly rare occurrence.

#### WORKSHOP Developing æpps on æternity Blockchain using Sophia Language



##### MILEN RADKOV

This workshop is an intensive and fast-paced training course that will help learners quickly grasp the principles behind the æternity architecture as well as learn how to build basic smart contracts on the platform through Sophia. In the second part of this crash course, participants will also experience first-hand how to integrate æternity features into their contracts. The workshop covers the functional mark-up language (ML) Sophia, built-ins, functional programming, first-class objects, oracles and state channels among others. We also tackle real-life use cases to help participants gain an understanding of its potential implications in solving global problems – both for business and social impact.



**Ilya Pupko** is Jitterbit’s Chief Architect, responsible for outlining the company’s technical vision, leading Jitterbit’s software, systems and infrastructure innovation, and being an overall technical evangelist. He draws on almost two decades of extensive IT expertise, involving hands-on experience at major international corporations, including LexisNexis Risk Solutions and First Advantage. Ilya holds an MBA with concentration in entrepreneurship, marketing and global business, and a bachelor’s degree in computer science and engineering.

## Shifting paradigms

# Forget about the new internet! Blockchains are the latest ‘Linux alike’ revolution

In this article, Phil Zamani, Chairman and CEO of the AERGO Foundation, and co-CEO at Blocko Inc. explains the value of blockchain technology to the average consumer. It might not be the “new internet”, but it is still just as revolutionary. See how blockchain is paving the road.

by Phil Zamani

Forget everything you think you know about blockchain. You might have heard that this “disruptive” technology is comparable to “the internet revolution of 1996.” This definition is often utilized to explain blockchain to a wider audience how important the technology is, in simpler but more thrilling terms. But is that definition accurate? Until now, crypto experts haven’t been able to agree on what blockchain real value is to custo-

mers, other than its similarities to the internet revolution of the ‘90s. My approach is slightly different, but not less revolutionary: successful blockchain platforms are meant to follow a path similar to Linux, which paved the road to creating one of the world’s most successful open-source projects in history, Android.

But first, let’s explore how I became involved with the technology that is steadily transforming from a trend into a must-have for all startups, VCs, and enterprises alike.

## The source of inspiration: The Linux experience and how it paved the road to Android's creation

Even the most basic internet user has heard of Linux. Although most people have no idea what it does exactly, Linux is the most utilized open-source platform and it's the underlying technology that powers most of the world's IT systems.

For the last 20 years, I've helped customers adopt open-source platforms based on the Linux project. I became an expert at helping companies discover, experiment, and implement these open-source platforms like Linux in their business strategies. Often, these implementations were made at the expense of replacing Unix systems sold to them by bigger names (such as IBM, HP, and Oracle) with tailored Linux systems. In 2001, my team helped Nokia's advanced mobile phone division run a version of Linux on Nokia's new innovative mobile phone architecture. The core maintainer of the software refused. Years later, under the guidance of Google, a new project based on the same Linux Kernel emerged: Android, the world's most successful open-source consumer project in history.

All these years working in open-source taught me a few lessons on how a new technology expands in the market. Most importantly, it gave me the tools to recognize when it's time to embrace Linux-like technologies, and I believe blockchain is meant to be the new open-source technology spreading across enterprise-IT business environments.

## A paradigm shift: Why blockchain is the new Linux and not the new internet

This is not only great news for Android lovers and tech-savvy people. Apple fans will also benefit from utilizing IT systems that implement blockchain-based technology as a new open-source phenomenon. Regardless of the ICO fever we witnessed throughout 2017 and the millions of dollars blockchain-based projects secured in investments, the technology hasn't taken off and reached the widespread adoption that was expected. As we leave behind a promising 2018 and begin an uncertain 2019 for the crypto community, public blockchains still haven't driven any consumer use. But Linux did not become important because regular people used it, the mainstream audience felt more comfortable with more expensive and less powerful alternatives such as Microsoft and MacOS. Instead, Linux became the underlying infrastructure that powered Android and most IT systems.

So why do I expect blockchain to take the same direction that Linux did?

Because we are experiencing a phase of data overload: more customer data than ever before is in the hands of only a few digital players such as Google or Amazon. These few companies have exclusive access to most valuable and personal information, and they even dare to use it for their own marketing campaigns. Most impor-

tantly, even with some of the world's most advanced IT systems, data still gets compromised. Companies are starting to realize that open-source technology (like software did with Linux) will enable much needed trustless environments. Data will be put in the hands of all parties involved, and not only a few big corporations. Blockchain will allow businesses to share sensitive data like supply-chain registries, transaction books, and consumer data effectively and securely without involving a middleman. Distributed ledgers have the power to transform IT systems by providing them with tools to dictate how much they trust other parties.

It takes a deep understanding of open-source technology to properly implement its strengths into a business strategy, such as a product release, or incorporate it into a platform that requires a merger with another technological source. While it's true that a sound education can teach an engineer everything he needs to know about open-source, there is still no better way to understand the details behind the tech than to experience it first hand. My personal and professional trajectories have resulted in over 20 years of actionable open-source involvement. I have proudly reached a place where I was able to acknowledge that open-source became a limitation and it needed an extra boost, something to catapult it into the modern world, where it could change a variety of industries for the better. It's time to pivot, and where to? Blockchain.



**Phil Zamani** is the chairman of the AERGO Foundation and co-CEO at Blocko, Korea's largest blockchain infrastructure provider. He has over twenty years of experience in managing open-source businesses and has worked as senior VP at Deutsche Telekom's cloud business unit, VP of sales and business development at RedHat, and as Global Head of Big Data & cloud models at Banca Santander.

## Let's Blockchain with Hyperledger Fabric and Composer

# How much is the fish?

Few technologies have made such a strong impression in regards to their implementation as Blockchain. The image of blocks lined up as a chain is literally embossed in people's minds. Many conversations about Blockchain revolve around this image and the actual implementation. That's all nice and fun, but it also carries the risk that the key issue is passed over in the discussion: the application. Therefore, the goal of this article is to explore that issue above all. We need to clarify when and why a blockchain solution makes sense. Then, using Hyperledger Fabric and Hyperledger Composer, we will also implement a small use case directly.

by [Thorsten Deelmann](#) and [Jannik Hüls](#)

The world's largest container shipping company Maersk has launched the TradeLens [1] platform in cooperation with IBM. TradeLens now has more than ninety partners who really just want to do one thing: transfer goods digitally via TradeLens. So this can for example be a salmon caught in the northwestern Pacific, which finds its way to our German retailers through a number of stations. TradeLens is based on Hyperledger Fabric. But what is Hyperledger Fabric and why can such technology be interesting for trading platforms like TradeLens?

Hyperledger Fabric is a framework that was explicitly created for the development of business applications on distributed ledger technologies (DLT). Four points are essential here, which differentiate Hyperledger Fabric from public solutions such as Ethereum or Bitcoin: Anonymity participation in the network is not possible. Each participant has an identity for authentication and corresponding roles for access control. Confidential transactions can then be used to control who is allowed to see and execute them. In addition, no proof-of-work validation is necessary; it would not be practical for a

business application anyway, since the consensus mechanisms would take too much resources and time. In Hyperledger Fabric, validation and consensus building are depicted through the distributed validation of transactions, resulting in the fourth property: the ability to implement business logic.

## Hyperledger Fabric in the Hyperledger Universe

Hyperledger is a Blockchain open-source project which was launched in 2015 under the auspices of the Linux Foundation. The term Hyperledger does not refer to a specific technology here, but rather a project in which several teams develop DLT frameworks and tools. The goals are reusability, standardized interfaces and interoperability between the different projects. Listed below are the five currently active Hyperledger DLT frameworks and their main drivers:

- Hyperledger Sawtooth (Intel)
- Hyperledger Iroha (Suramitsu)
- Hyperledger Fabric (IBM, Digital Asset)
- Hyperledger Burrow (Monax)



- Hyperledger Indy (Sovrin)
- For tools, there are also five currently active projects with differing goals:
- Hyperledger Caliper (Blockchain benchmarks)
- Hyperledger Cello (Creation and management of Blockchain infrastructures)
- Hyperledger Composer (Development of business network applications based on the Composer high-level language)
- Hyperledger Explorer (Visualization of operational Blockchain systems)
- Hyperledger Quilt (interoperability between different Blockchains)

We will now use Hyperledger Fabric as a DLT framework and Hyperledger Composer as a tool.

### Asset Tracking - Same status for everyone

Going back to our trading platform example- we still need to answer the question of where the fish which we buy in the retail sector comes from. But also, which way did the fish take to get there? And how often, for example, did duties have to be paid or freight papers have to be exchanged? Digital information processing at the level of a single trader is usually quite good. However, each of these individual traders maintain well-kept information silos. Exchange of data or even a common database is usually dead loss in these cases. Yet that's exactly

#### Listing 1

```
asset Fish identified by id {
  o String id regex=/[0-9a-fA-F]{8}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{4}\-[0-9a-fA-F]{12}/
  o String type
  o Double weight
  o DateTime caught
  --> Retailer owner
}
```

#### Listing 2

```
abstract participant Retailer identified by id {
  o String id
  o String name
  o String country optional
}
```

#### Listing 3

```
participant ShippingCompany extends Retailer {
  o Ship[] ships
}
```

what global trade could benefit from. In a system where all participants process the same data, trading can be made more efficient, safer and, above all, more traceable. But who should have sovereign rights to the data? Everybody on equal terms would be best – and that's what distributed ledger technologies like Hyperledger Fabric offer.

The ledger is a data management system that keeps track of transactions and the resulting status of goods. In the Hyperledger context, these goods are called Assets. The transactions are carried out by Participants. So let's move on with our fish: the frozen salmon we have acquired is an asset. It is transferred from the catch all the way to our supermarket through many participants. Each participant transfers the asset, changing its status and owner in the process. Each fish has exactly one single status in a data management system. This information is traceable for all participants and above all, it is stored in the Ledger with no possibility of alteration.

### Hyperledger Composer - focus on the essentials

So how are assets and participants modeled and implemented in Hyperledger Fabric? Natively you can do that in Go, JavaScript or even Java. However, the initial hurdle is quite high and you end up with a large amount of boilerplate code. That's why Hyperledger Composer as a tool was launched. It allows you to develop Hyperledger Fabric applications, while hiding the complexity of the underlying infrastructure. Assets and participants are defined using the internal object-oriented Composer Modeling Language.

So what would our fish look like as an asset? To keep our example as simple as possible, let's assume that a fish has the properties of ID, species, weight, catch date and owner (Listing 1).

The code is self-explanatory and already illustrates one of strengths of Composer: simplicity. But definitely more complex domains and problems can be modeled. Concepts such as namespaces, imports, abstract types, enums, relationships, validations, optional fields and default values are available for this. In *Fish*, you see a regular expression that checks the ID for a UUID format. The owner - *Owner* - is represented by a relationship.

The participants are similarly defined using the keyword *Participant*. In our use case, all participants are any form of a trader. Therefore it makes sense to define an abstract *Retailer* (Listing 2).

Specific assets, such as a shipping company, the vessels of which initially catch the fish, can then be derived from this and extended (Listing 3).

Since a ship is neither a participant nor an asset and serves only as a structuring aid, it can be modeled as a so-called concept (Listing 4).

We now model a logistics company as yet another participant, which buys the caught fish from the shipping company and distributes it further (Listing 5).

## Distributed transactions for consensus

“I can also map assets and participants using my own database, why all the work?” is the question many will ask themselves now. We can certainly reproduce this in a centralized system. But we have to keep in mind what participants the system has: if a consortium of trading partners - as was done in TradeLens for example - decide to implement a trading platform, who should have sovereign rights over the data? With Hyperledger Fabric, each participant keeps the data - asset status and transaction logs - and everyone is equally involved in the system. This not only avoids making decisions about data sovereignty, it also eliminates the need to have to pay a central instance for the operation of a database.

Moreover, we need more than just pure data storage. Real added value is only achieved when we implement transactions. So back to our fish: along with the trade, payments must be made. Prices vary, for example depending on the quantity purchased, on the quality or even the time of day. These conditions become the logic of a transaction. Together, assets, participants and transactions form the Chain Code. Other DLT frameworks also refer to this as Smart Contracts. Transactions are modeled just like participants and assets (Listing 6).

The simple transaction shown in Listing 6 describes a change of ownership of a fish. There is an ID for the contract resulting from this transaction, a new owner, and of course the fish asset involved. We have not yet modeled a contract asset. For the implementation of the transaction, we assume that a *Contract* has been modeled, which has a price, an old owner and new owner, as well as the reference to the fish and of course an ID.

### Listing 4

```
concept Ship {
  o String name
  o Integer brt
}
```

### Listing 5

```
participant LogisticsCompany extends Retailer {
  o Truck[] trucks
}
```

### Listing 6

```
transaction TransferFish {
  o String contractId
  --> Retailer newOwner
  --> Fish fish
}
```

The function (Listing 7) is pure JavaScript with the support of Composer SDK. The comment to the function specifies that this is a *transaction* and in the transaction object we have *transfer Fish*. Each fish is sold at a special price. In the transaction, this calculation is outsourced to a dedicated specialist function which can be arbitrarily complex. For example, you could model a demand asset such that the transfer will only be accepted by the potential buyer at a certain price. In the presented transaction, there are only two prices, the standard price and one with a twenty-percent markup for very large fish. The contract asset is only there to record the terms and conditions of a conducted transaction. Finally, the receiver becomes the new owner of the fish and the corresponding asset will be updated.

How can other participants be sure then that buyers and sellers did not cheat? You just simply carry out the transaction as well and compare its result with the one received. In the above example, we would check if the correct price was saved and if the new owner is actually

### Listing 7

```
/**
 * @param {de.example.TransferFish} tx
 * @transaction
 */
async function transferFish(tx) {
  // Calculate price
  let price = calculatePriceFor(tx.fish)
  if (tx.fish.weight > 10000) {
    price *= 1.2
  }

  // Create a contract
  const contract = getFactory().newResource(assetNamespace, 'Contract',
    tx.contractId)
  contract.newOwner = getFactory().newRelationship(participantNamespa
    ce, 'LogisticsCompany', tx.newOwner.getIdentifer())
  contract.oldOwner = getFactory().newRelationship(participantNamespa
    ce, 'ShippingCompany', tx.fish.owner.getIdentifer())
  contract.fish = getFactory().newRelationship(assetNamespace, Fish,
    tx.fish.getIdentifer())
  contract.price = price

  // Persist contract
  const contractRegistry = await getAssetRegistry(`${assetNamespace}.
    Contract`)
  await contractRegistry.add(contract)

  // Change ownership of the fish
  const fishRegistry = await getAssetRegistry(`${assetNamespace}.Fish`)
  let fish = await fishRegistry.get(tx.fish.getIdentifer())
  fish.owner = getFactory().newRelationship(participantNamespace,
    'LogisticsCompany', tx.fish.owner.getIdentifer())
  await fishRegistry.update(tx.fish)
}
```

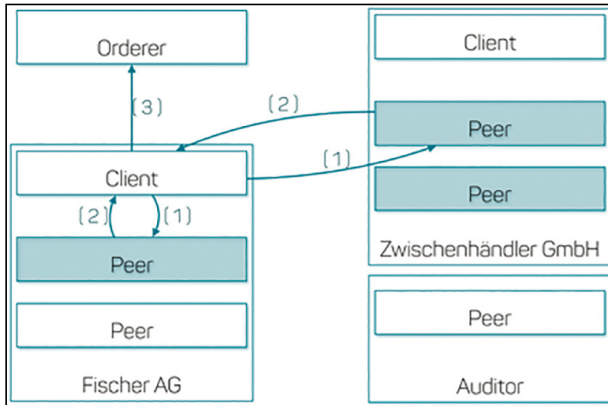


Fig. 1: Schematic presentation of a Hyperledger Fabric network

also the receiver of the transaction. This process is called endorsement. A transaction is not considered valid until a sufficient number of participants have “endorsed” it. The Endorsement Policy of the network defines how much exactly the sufficient number of endorsements is. But it should be bigger than one. This is the only way to ensure that at least one other participant has also checked the transaction.

A multi-level process will run through until a transaction can be considered as valid by all the participants. To make sure we can understand how this important step of the endorsement takes place, let’s look at an example:

Figure 1 shows a Hyperledger Fabric network with three participants: Fischer AG, Middle-Man GmbH and Auditor. Each of these participants operates nodes, which are divided into the three categories: Peer, Client and Orderer. A Client is the interface to the network and the starting point for new transactions. Peers store the asset state and the transaction history in their own ledger. In addition, peers may have the special Endorser role, meaning they are considered for necessary endorsements. The Orderer determines the sequence of valid transactions and distributes them to the Peers of the network. The Orderer is presented as a bit of an oddball here. In the current version of Hyperledger Fabric, there are two ways to implement an Orderer: Solo or Kafka. Even if Kafka is distributed, it is a centralized service used by our Hyperledger Fabric network. Consensus in the network is achieved through the complete regulation of the transactions by the Orderer. Further implementations of the Orderer concept are planned in later releases of Hyperledger Fabric, which should make it possible to distribute it to the infrastructure of the individual participants as well.

The initiation of a transaction is also illustrated in Figure 1. For the endorsement of a new transaction, a client sends a so-called Propose message (1) to the endorsement peers. That way the client asks the endorsers to simulate and validate the transaction. Nothing has been written to the ledger yet. If successful, the endorser will sign the response and send it back to the client. (2) The Endorsement Policy determines how many

endorsers must have at least simulated the transaction. If the client then receives a sufficient number of positive responses, the transaction is distributed through the orderer to the peers so that they can now update their ledger according to the transaction. The endorsement policy is therefore a very central element of the system. By having different peers simulate and validate the same transaction, trustworthiness is built - and that’s exactly what we want to achieve.

But how do we integrate ourselves into the existing world? The chain code given above is completely isolated, but sometimes it may be useful to use external services as well. So let’s assume that the fish should be traded at current market prices. That means that we have to source the current fish price from an external source.

In Listing 8, an external service is used to determine the current fish price. Now let’s recall what happens when a transaction is executed – here we will probably quickly realize that this can be a dangerous idea in the end: due to the required endorsement, the transaction will be validated by  $n$  peers. That also means that  $n$  peers will call up the external interface. Let’s suppose that the base price of the fish changes during the endorsement. So we can have a situation in which endorsers receive different results; therefore a consensus cannot be reached as to the result. So the transaction would not be validated in this case. We should only use external services if:

1. they do not change and use any states,
2. they are idempotent, meaning that they always generate the same outputs for the same inputs.

In general, the chain code has to be deterministic. For example, you should absolutely avoid having a UUID or timestamp generated and committed within a transaction. Since the results would be different for each endorsement, no consensus would be found. The transaction would therefore never be executed.

## Restrictions during execution

The question of who can sell the fish, or who can execute the translation, still remains open. So far, the *transfer fish* function does not include a permission check. This means that currently any participant - regardless of the actual owner - could transfer any fish asset. In general,

### Listing 8

```
async function transferFish(tx) {
  let price = await request.get('https://currentprice.com/fish')
  if (tx.fish.weight > 10000) {
    price *= 1.2
  }
  ...
}
```

we have two ways to implement permission checks. For one thing, we could check in the transaction whether the current owner of the fish asset also triggered the transaction.

Implementation in code (Listing 9) is perhaps the most obvious approach, but here the actual business logic is heavily mixed with the permissions, leading to unnecessary complexity. Composer therefore offers its own declarative Access Control Language (ACL) - the ability to implement access permissions detached from the actual business logic. The ACL distinguishes between access to the business network such as assets, participants and transactions as well as administrative access as a network admin. In both cases, the least privilege principle applies, which means that we must allow access explicitly. The permissions are defined in the *permissions.acl* file. If the file is not present, everything is allowed. We will create the entry shown in Listing 10 there.

With this entry, we make sure that the *UPDATE* operation on the asset *Fish* may be executed only by the owner of the asset through the *TransferFish* transaction. We check the condition for access here based on the participant who executed the transaction. Such a condition could for example also be a simple validation of transaction data. So we have the ability to centrally specify access permissions. However, it has been shown that this can quickly become very confusing and therefore the ACL is subject to constant refactoring and should be sufficiently tested.

### Added value through queries

In principle, we now have the most important Composer Tool ready. We have seen how assets, participants and

transactions are implemented and used. So we know how to describe a fish and its traders using Hyperledger Composer and send it on its digital way. In addition, by describing the endorsement, we have seen how Hyperledger Fabric brings trustworthiness to transactions for a consortium of traders. Another important point that we have not yet considered is reading of the data. Data storage in the ledger takes place in a CouchDB. This allows complex queries using an SQL-like syntax. For example, each trader could use such queries to find out which contracts the trader had completed to date. You can describe these queries using so-called Querys (Listing 11).

Queries have pure read access. Unlike for transactions, no endorsement is necessary and the queries can be made on the local ledger.

### Permissioned Ledger

Can anyone simply join in? No. Hyperledger Fabric implements a so-called Permissioned Blockchain. This means that all participants are known. Therefore, a consortium is a very central element in this context. Traders need to come together to form a consortium (like was done at TradeLens) and jointly implement the platform. From a technological viewpoint, this closed pool of candidates has the great advantage of being able to do without strict and complex consensus algorithms such as proof of work, letting you achieve significantly higher throughput.

But if we assume that we have a closed pool of known participants, there is still the issue of how each participant can be assigned a digital equivalent. The relation between physical and virtual participants is created using Hyperledger Composer through the generation of so-called Identities. This assignment results in the creation of a so-called card. This contains the connection information to the network as well as an X.509 certificate required for authentication. Let's take a quick look at one of the traders (Fischer AG) in the consortium and how this trader can now participate in the network:

The participant Fischer AG is provided with the *fischer\_ag.card* over a secure communication channel. If Fischer AG wants to transfer its caught fish, it will always use this card to make a transaction. It contains

#### Listing 9

```
async function transferFish(tx) {
  if (tx.fish.owner.getFullyQualifiedIdentifier() !== getCurrentParticipant().
  getFullyQualifiedIdentifier()) {
    throw new Error('Attempted to transfer fish that is not owned by the
    sender.')
  }
  ...
}
```

#### Listing 10

```
rule TransferFishOnlyThroughOwner {
  description: "Only the owner of a fish asset is allowed to transfer it."
  participant(t): "de.example.Retailer"
  operation: UPDATE
  resource(c): "de.example.Fish"
  transaction: "de.example.TransferFish"
  condition: (c.owner.getIdentifier() === t.getIdentifier())
  action: ALLOW
}
```

#### Listing 11

```
query ContractsForParticipant {
  description: "Returns all contracts for a participant"
  statement:
    SELECT de.example.Contract
    WHERE (
      (oldOwner.id == _$participantId) OR
      (newOwner.id == _$participantId))
}
```

the Private Key used to sign messages, as well as the Endorsement Peers and Orderers that can be addressed by the Fischer AG client. We note that proper storage of this card is particularly critical in terms of security. In addition, secure distribution is neither trivial nor particularly user-friendly. To improve usability, Hyperledger Composer provides its internal REST server - a middleware used to access the underlying Fabric network. The REST server makes it possible to store the cards of the individual participants in dedicated wallets and use standard protocols such as OAuth 2.0, SAML or OpenID Connect for authentication. This makes it very easy for Fischer AG applications to participate in the network; but we should remember that this actually contradicts the desired decentralized infrastructure. The

REST server has full power over the cards of all participants and is therefore a central trust center - a function which is actually not needed. As is so often the case, this is a trade-off between usability and security.

## Conclusion

We hope that this article has provided you with a comprehensive overview of the benefits and uses of Hyperledger Fabric. Experience has shown us that tools such as Hyperledger Composer make it easy to quickly implement a business application on Hyperledger Fabric. As a result, this very complex technology becomes suitable for everyday use in software development. The community is already very big, and you can look forward to the upcoming releases. In particular, the alternative implementation to the Orderer concept, which has been announced for the end of 2018, should be yet another milestone in the Hyperledger universe.

### SESSION Hyperledger Fabric 2.0



#### ARNAUD LE HORS

After more than three years of open source development, the Hyperledger Fabric framework is reaching its second major milestone with the release of version 2.0. Fabric version 1.0 represented a breakthrough in blockchain technology for the enterprise with several unique features, including:

1. a consensus mechanism providing more flexibility, better performance, and finality,
1. channels providing for greater privacy.

Version 2.0 brings a new set of functionalities for greater decentralization, including a new ordering option and a new lifecycle model to deploy smart contracts as well as a new way to handle digital assets called FabToken. This talk will give attendees an update on the latest developments of Hyperledger Fabric, explaining the new features and what it means for application developers.



**Jannik** has been part of the codecentric team in Münster since 2016. He mainly deals with the operation of IT systems. Cloud infrastructures and the integration of on-premise systems are on his agenda every day.



**Thorsten** has been with codecentric in Münster since 2015. For more than ten years, he has been involved in software development in distributed systems. He is particularly interested in the development and operation of containers as well as the brand new discipline of Chaos Engineering.

## Links & literature

[1] <https://www.tradelens.com/>

## Proof-of-Authority Chains with Parity and the Aura Consensus Engine

# Your Own Ethereum Consortium Chain

Bitcoin, Litecoin, Ether, ICO ... As the fog of buzzwords, hype and speculation slowly gives way to reality, the fledgling technology is being faced with new challenges. Uncontrolled public networks are not the right choice for every application, and in addition to environmental considerations, economic concerns regarding high energy consumption by miners also play a role when choosing a technology stack.

by [Christian Scharr](#)

---

If you mention blockchain, newcomers at first associate the term primarily with Bitcoin, or generally with cryptocurrencies. But that's only half the story, just like the belief in the one and only blockchain. Meanwhile, we now have nearly as many implementations and associated networks as there are text editors.

One of the most popular platforms besides Bitcoin and its derivatives is Ethereum. Designed by the Canadian-Russian software developer Vitalik Buterin and formally specified by the British developer Gavin Wood in the Ethereum Yellow Paper [1], Ethereum is known above all as a pioneer in the management and

execution of decentralized programs in the form of smart contracts. But Solidity [2], the contract-oriented, Turing-complete programming language for Smart Contracts, is by no means the only thing that distinguishes Ethereum from the rest. The transition from the energy-consuming proof-of-work (PoW) to the more efficient proof-of-stake (PoS) consensus algorithm already foreseen in the Yellow Paper began in late 2018 with Hard Fork on the new Constantinople version of the Metropolis milestone.

With such a change of the consensus engine planned from the beginning, it is therefore not surprising that most Ethereum clients have also been using alternative engines for quite some time. This includes Parity, the most widely used Ethereum client.

## Parity

This client, which was developed in Rust, focuses on performance and security and is also supplied as a Docker image, in addition to the binaries for Windows, Mac and Linux. You launch the modern and sleek web interface by default at <http://127.0.0.1:8180/>, giving you easy access to the included tools and external DApps. In addition, it supports not only the current PoW engine, but also a hybrid Casper FFG as well as a simple and very fast Proof-of-Authority (PoA) algorithm called Aura. The latter is a nearly perfect solution especially for consortium chains (see Table 1).

### Listing 1: Basic structure „chain.json“

```
{
  "name": "CHAIN_NAME",
  "engine": {
    "ENGINE_NAME": {
      "params": {
        ENGINE_PARAMETER
      }
    }
  },
  "genesis": {
    "seal": {
      ENGINE_SPEZIFISCHES_GENESIS_SIEGEL
    },
    "difficulty": "0x20000",
    "gasLimit": "0x2fefd8"
  },
  "params": {
    "networkID": "0x656e7477696366b6c6572",
    "maximumExtraDataSize": "0x2",
    "minGasLimit": "0x1388"
  },
  "accounts": {
    GENESIS_ACCOUNTS
  }
}
```

Parity can be configured in two ways: Using CLI options or through a *.toml* configuration file. If parity is known in the system path, calling up *parity --help* will provide an overview of the available options. Further information can also be found in the official Parity Wiki in the CONFIGURING PARITY ETHEREUM [4] menu item.

### Listing 2: System-Contracts

```
"accounts": {
  "0x0000000000000000000000000000000000000000000000000000000000000001": {
    "balance": "1",
    "builtin": {
      "name": "ecrecover",
      "pricing": {"linear": {"base": 3000, "word": 0}}
    }
  },
  "0x0000000000000000000000000000000000000000000000000000000000000002": {
    "balance": "1",
    "builtin": {
      "name": "sha256",
      "pricing": {"linear": {"base": 60, "word": 12}}
    }
  },
  "0x0000000000000000000000000000000000000000000000000000000000000003": {
    "balance": "1",
    "builtin": {
      "name": "ripemd160",
      "pricing": {"linear": {"base": 600, "word": 120}}
    }
  },
  "0x0000000000000000000000000000000000000000000000000000000000000004": {
    "balance": "1",
    "builtin": {
      "name": "identity",
      "pricing": {"linear": {"base": 15, "word": 3}}
    }
  }
}
```




	Proof-of-Work	Proof-of-Stake	Proof-of-Authority
Security	 New blocks are secured by applying very costly mathematical puzzles (mining).	 The blockmaker vouches for the correctness of the new block with his or her own stake.	 Only special accounts, the Validators, are allowed to create new blocks.
Incentive	The first miner to solve the math problem gets a reward.	The stakeholder may claim the transaction costs.	The validator may claim the transaction costs.
Conclusion	The required power can usually only be provided by a few large mining networks. Low throughput Power-hungry	A very fast and inexpensive alternative, but it also has its problems. "Nothing-at-stake" (for details see [3])	This variant is well suited for use in trusting partner networks. Trusting partners as a prerequisite

Table 1: PoW, PoS and PoA in comparison





own practical experience, I can report that this procedure can spare a lot of stress when setting up a new chain. The smart contract responsible for data management and administration has only one limitation: It has to implement the official *ValidatorSet* interface (Listing 5).

An example of a current publicly running validator contract can be found on the Ethereum-Kovan network [6]. Among the vast amounts of existing implementations, in addition to voting/reporting systems used to select (deselect) validators, there are those that limit access to *owner* or individual *administrator* accounts.

If the implementation is selected or self-written, the contract must be compiled and pasted into the Genesis block as a bytecode. This is done through an entry in the *accounts* range of the chain specification in the form of a line of code in the following format:

```
"0x...": {"balance": "1", "constructor": "0x..."}
```

Once complete, the chain specification can be passed to the Parity client with the startup parameter.

```
parity --chain path/to/chain/spec.json
```

Alternatively, the path can also be entered in the aforementioned *config.toml* to not have to manually specify it each time:

### Listing 5: “ValidatorSet”

```
contract ValidatorSet {
    /// Issue this log event to signal a desired change in validator set.
    /// This will not lead to a change in active validator set until
    /// finalizeChange is called.
    ///
    /// Only the last log event of any block can take effect.
    /// If a signal is issued while another is being finalized it may never
    /// take effect.
    ///
    /// _parent_hash here should be the parent block hash, or the
    /// signal will not be recognized.
    event InitiateChange(bytes32 indexed _parent_hash, address[] _new_
set);

    /// Get current validator set (last enacted or initial if no changes ever
made)
    function getValidators() constant returns (address[] _validators);

    /// Called when an initiated change reaches finality and is activated.
    /// Only valid when msg.sender == SUPER_USER (EIP96, 2**160 - 2)
    ///
    /// Also called when the contract is first enabled for consensus. In this
case,
    /// the “change” finalized is the activation of the initial set.
    function finalizeChange();
}
```

```
[parity]
chain = "path/to/chain/spec.json"
```

If everything went well, you are now the proud owner of your own Ethereum Blockchain network based on PoA via Parity Aura Engine. Adding new nodes and validators is relatively easy afterwards. All you need is exchange the *chain.json* with your new partner to connect to your network. After successful synchronization, if necessary, the new node can be added as a validator by calling the appropriate method on the Validator Contract.

## Conclusion

The open-source Ethereum platform, in conjunction with the Parity client, is a powerful tool for building your own blockchain networks. Whether for private use, small or medium-sized companies, you can implement your own Blockchain solutions with moderate effort. The trick here is to find trustworthy and willing partners for a common set-up, rather than technical implementation. Only in a multi-participant community can this technology, which is based on trust-building through cryptographic mechanisms, realize its full potential.



**Christian Scharr** is an agile full-stack developer at CSS Insurance in Lucerne, Switzerland. During his business information science studies, he developed a passion for breathing life into abstract things like blockchain with clever solutions.

## Links & literature

- [1] <https://github.com/ethereum/yellowpaper/>
- [2] <https://github.com/ethereum/solidity/>
- [3] <https://medium.com/@jonchoi/ethereum-casper-101-7a851a4f1eb0>
- [4] <https://wiki.parity.io/Configuring-Parity-Ethereum>
- [5] <https://wiki.parity.io/Chain-specification>
- [6] <https://github.com/parity-contracts/kovan-validator-set/>

## Solving the oracle issue

# Reliable blockchain oracle for financial data

Blockchain-based applications have been the talk of the town for several years. It is always very easy to talk about the „decentralized revolution“, and Management is sure to have a number of great ideas on how you can improve your own product with a decentralized app (DApp). The fact that tangible technical issues can appear along the way is not necessarily obvious at first glance. This article therefore introduces the reader to the „oracle issue“, which is the seemingly simple process of making entries in blockchain applications.

by Samuel Brack

Basically, a blockchain is a tool for creating a global consensus about an (orderly) sequence of transactions. Common to all blockchains is that this consensus is created by as many participants as possible in a peer-to-peer network and can be independently verified by all participants. To this end, there are Miners which receive the transactions and combine them into blocks in competition with each other. In doing so, they also solve a cryptographic task, the difficulty of which lies in the fact that to create a block on average a certain time set by the network needs to pass - for example, with Ethereum this is about 15 seconds. These blocks then form a chain (the blockchain) the cryptographic relationship of which can only be falsified with a great deal of computational effort. In general, this is an audit-compliant transaction list that cannot be changed by normal attackers.

Smart contracts are an extension of this basic idea. Used extensively for the first time in the Ethereum system are small programs - so to speak - in each transaction. They are executed in parallel by all participating miners, while the result ends up in the blockchain forever. Such contracts with special functions can also access and even manage the monetary value of the transaction. So you could also refer to Smart Contracts as a

type of programmable money. It is already conceivable that many areas of the financial world can be supplemented by smart contracts or their implementation can be further automated.

One challenge is to feed these smart contracts with data from the real world. Let's imagine a scenario: The creator of the contract wants to assign the current exchange rate of the euro to the dollar to a variable. In a normal program on his PC, he would call up the API of the European Central Bank to get that value online. In the case of a smart contract, however, he does not know which machine will ultimately successfully add this transaction to the blockchain (there are thousands of miners, all of which compete with each other to be the first to solve the cryptographic puzzle and integrate the transaction into the blockchain), nor does he know what view of the world this particular computer will have. Who can guarantee that this computer is not behind the „Great Firewall of China“ and indeed may have access to the Ethereum network, but access to the ECB website is blocked?

For these reasons, there is no intention in Ethereum to get data through the external interfaces of the executing computers. All input data must always reach the miner directly via a transaction in order to be included in a calculation. This interface between real world data and a smart contract is called an oracle.

Such an oracle usually consists of a function in a larger smart contract, which is called up by a transaction and can write to a variable in the internal state of the contract. If the variable needs to be updated, the information provider generates an Ethereum transaction with a call-up to this function with the desired value as a parameter. This transaction is sent to the network and ends up with all the miners; the miners then incorporate them into the global history and execute them. This means that in the Ethereum State Machine, a function in the Smart Contract is called which in turn sets the variable to be manipulated to the new value.

After this oracle transaction has finally landed in a block, the next read access to the variable will show the new value. All transactions (and function calls) in Ethereum happen atomically. Despite the extremely distributed calculation of smart contracts, there is no invalid state when reading a variable which only contains half of the update.

### Problem in the financial sector

Due to the coupling of monetary transactions and Turing-complete programs, smart contracts are predestined for financial and investment banking applications. Any data needed from other systems, web interfaces and user inputs must be packed into transactions and transferred over to the executing smart contract. It's not cheap though: The Ethereum Network generates transaction costs at the level of the sender of a transaction, depending on the network utilization. A kind of auction procedure is used here. Each transaction consumes a certain amount of „gas“ depending on the compiled machine code of the transaction.

The gas should cover the costs of the miner, who eventually has to execute the code. A loop needs more gas than a simple instruction. For this gas (which can

be deterministically calculated from the source code), the sender of a transaction then sets a price per gas unit, while the miners then search for the most profitable transactions for the next block. Thus a sender of a transaction can fairly accurately determine how quickly the transaction will end up in a block: If you set a low price, the transaction will be included in a later block later when there are fewer transactions on the network. There is no guarantee that a transaction will not starve because it was waiting forever for inclusion in a block.

These costs can range from a few cents up to a few euros per transaction. So at the end of 2017, there was a popular game called Cryptokitties as a smart contract. Just by the popularity of this one game, transaction costs increased enormously in a short period of time.

This cost argument leads to the idea of saving costs through shared data. So not everyone has to package the data of an API as an oracle independently just because the exchange rate between the euro and the US dollar is needed, but rather a shared oracle provides this exchange rate regularly and anyone interested can then programmatically access it. The sample source code of a very simple oracle for the exchange rate between the euro and the US dollar can be found in Listing 1. The

### Listing 1: Solidity source code for an EUR-USD oracle

```
contract EURUSDOracle {
    uint256 public exchangeValue;
    uint256 public lastUpdateTimestamp;
    address owner;

    event oracleUpdated(
        uint256 exchangeValue,
        uint256 timestamp
    );

    constructor() public {
        lastUpdateTimestamp = 0;
        owner = msg.sender;
    }

    function updateOracle(uint256 newExchangeValue, uint256
newTimestamp) public {
        require(msg.sender == owner);
        exchangeValue = newExchangeValue;
        lastUpdateTimestamp = newTimestamp;
        emit oracleUpdated(newExchangeValue, newTimestamp);
    }

    function getExchangeValue() public view returns (uint256, uint256) {
        return (exchangeValue, lastUpdateTimestamp);
    }
}
```

### SHORT TALK Blockchain-backed Analytics: Fraud Prevention in Data-driven Projects



#### MARKUS HERRMANN

Blockchain-backed analytics (BBA) is a scientific concept to transparently document the lineage and linkage of the three major components of a data-driven project: data, model and result. BBA enables stakeholders of data-driven projects to track and trace data, models and modeling results without the need of trust validation of escrow systems or any other third party. This talk covers the theoretical concept of BBA, showcases a BBA prototype. Participants will be provided with re-usable code and learn how to design blockchain-backed analytics solutions, e.g. for situations in which partners share and distribute data and AI models in an untrusted setting.

oracle is based on real software that is used productively by BlockState. The programming language for smart contracts is Solidity. Basically, the oracle stores three variables: the actual value that should be saved (the exchange rate), the Unix timestamp of the last update, and the address of the entity allowed to update that oracle. The example is deliberately kept very simple - in general, these contracts are of course much more complex, like traditional object-oriented programs.

In addition to the three variables, there is a so-called event. It can be called up from a smart contract function and then generates an event that can be monitored by a piece of front-end code. Common front-ends are JavaScript applications that provide Ethereum integration. This event can then be used to initiate further actions outside of the smart contract.

The main purpose of the designer is to define the owner, which will be the only one allowed to make updates later on. The usual code used to change ownership has been omitted for reasons of clarity.

Finally, the last two functions are the actual oracle functions: The owner uses the first one to update the oracle with fresh data. The function call-up is made via a normal Ethereum transaction, which includes the Application Binary Interface (ABI) of the smart contract and the parameters of the function. A miner can do this in the Ethereum Virtual Machine and write the result back to the blockchain. Afterwards, the smart contract stores the new value in the variables.

The last function can be used by anyone and does not require any gas as it only processes the last value of the oracle. This allows a node that reads the current state of the blockchain to automatically read this value locally and also use it in its own smart contract.

Due to its simplicity, this template is easy to adapt and can be flexibly used for all kinds of applications. The option of integrating it into other smart contracts makes it an interesting solution to use as a kind of building block in a larger context.

## Reliability of the input data

As already indicated, the update of values in an oracle usually has to come from a trustworthy instance. This is where the community approach comes in - here a representative is appointed to submit the data. Another benefit of these community-shared oracles is the reliability of data. In the case of the exchange rate between the euro and the dollar, there may be little room for interpretation, but what about the price of Bitcoin, for example? This is determined differently in different stock markets worldwide. It is customary then to calculate an average from these prices. A single smart contract user may not be willing or able to provide a reliable collection service for all the data needed first of all. Existing aggregator services in this sector stand out with their inconsistencies and lack of documentation. Such a jointly operated oracle therefore needs reliable, well-defined and documented data sources first of all.

If all this is available, the oracle can periodically retrieve this data, pack it in a transaction and send it. This also automatically generates a history that invariably writes the price into the blockchain and makes it immediately transparent for viewers later on, whether or not the oracle has provided correct data in the past. Building on such a trusted oracle, financial applications which depend on correct data can then be implemented in smart contracts. BlockState develops such applications. The underlying community project that serves the oracles with reliable data as outlined above is called DIA and is a non-commercial entity.

## Solution

As already indicated, the best solution for the oracle issue from the perspective of the author is an infrastructure operated jointly by the community, which on the one hand aggregates the raw data, pre-processes it and puts it in the oracle, and on the other hand, provides control mechanisms for the community.

Economic incentive systems are particularly effective in cryptocurrencies because of the native currency nature of the system, and so it makes sense to ensure data quality through such an incentive system. The DIA project has implemented such an incentive system. Open source developers can enter source code for data acquisition and processing there. With the token present in the system (a type of currency that is meaningful only in certain smart contracts), it is possible to determine an operation that bets on the acceptance or rejection of a specific piece of new source code for a particular documented task. After a few days, the smart contract automatically decides whether or not to accept the new code based on the bets. Also, a dispute system for errors or corrections discovered later on is provided. Financial applications based on this, such as those by BlockState, have then a solid foundation to process reliable and up-to-date data, for example, in another smart contract or in an application that is no longer on the blockchain itself.

Together with open-source source code, this results in a very transparent system in which it can be verified at any time (even for data that is already several years old) where the data comes from, how it is processed, what level of confidence supports this data and how the data is used. In the face of many unclear financial transactions in traditional investment banking, this should also be a welcome development overall.



**Samuel Brack** is a cybersecurity expert who researches data protection, digital currencies and distributed systems. As CTOP of the Goodcoin project, he has developed a completely anonymous bonus point system. Samuel is a published author in a number of specialist publications on IT, security and cryptography. He holds a Master of Science in Computer Science from the Humboldt University in Berlin and is an active member of the Berlin Hackers Community and the IEEE Computer Society in Los Alamos, California. He is co-founder of BlockState, which develops financial applications on the Ethereum Blockchain.

Corda, the open source enterprise blockchain

# Where the hell are the blocks?

Corda was developed by the R3 consortium [1] in collaboration with more than 200 technology and industry partners. According to the manufacturer, Corda is an open-source blockchain designed specifically for enterprise use. Unlike other blockchain solutions, information is only shared between the parties that actually need it. This information is referred to as „Shared Facts“ in Corda. This article shows how to create, distribute and historicize such shared facts.

by Christian Koller

Corda is a flexible framework that makes it possible to depict contracts electronically. Supporting the developer in the best possible way is a key aspect. Another important feature is privacy. Only a specific group of participants can validate transactions, therefore making Corda a Permissioned Blockchain. The core of Corda was written in the Kotlin programming language. Technical solutions can thus be created in a JVM-compatible technology. For the purposes of this article, I used Java.

## Parties

At least two parties are needed to create shared facts. Parties in Corda can be identified by their official name, meaning that the public key of each party is tied to a legal entity. Corda adheres to the X.509 standard [2]. Corda also assumes that you can basically trust your contracting partners. In the case of any disputes, however, decisions must be made by a court. A party is often referred to as a node in Corda.

## Shared facts

Would you desire a situation in which contracts of a company could be viewed by anyone in a B2B environment? No, definitely not. But this is exactly the case with a network such as Bitcoin. The rules are visible, and eve-

ry transaction can be followed by all participants in the network. Transactions in Bitcoin are not encrypted and can be viewed using the Blockchain Explorer [3], for example. In Corda, shared facts are not distributed all over the world, but only in a predefined circle of participants. This is accomplished by using private transactions. This can be best illustrated with a Venn diagram (Fig. 1).

The small white circles represent shared facts. Only Roger and Jürg know about Fact A, and only Jürg and André know about Fact B, while only André and Roger know about Fact C. Roger, Jürg and André know about Fact D. So the facts are shared only between those people who have an actual interest in it. In the terminology of Corda, a fact implements the `ContractState` interface because it reflects a particular state of a contract.

## Evolution of a fact

A shared fact may become invalid over time. For example, if Roger borrowed 100 euros from Jürg, that may be a valid shared Fact A today. For example, Fact A could represent the state of a promissory note (Fig. 1). Tomorrow, however, Roger could already pay half of the debt and thus only be 50 euros in debt. What happens to the existing shared Fact A? Since Corda is a blockchain according to R3 and unchangeability is an essential feature of a blockchain, this already shared fact may not be changed. Instead, a new fact A' is generated in a new

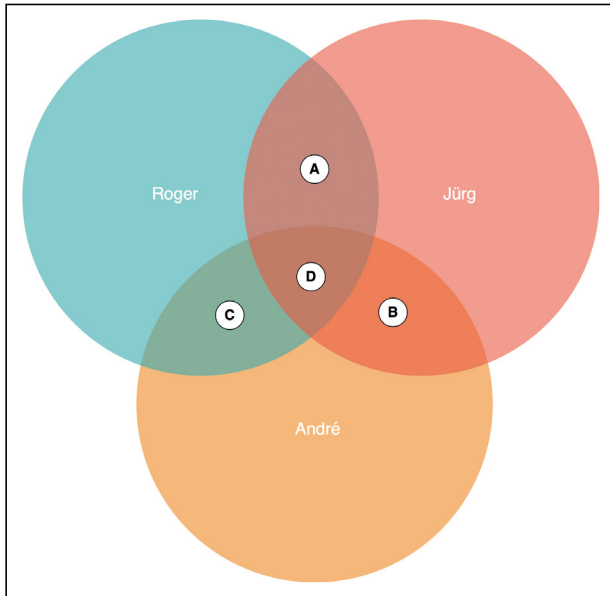


Fig. 1: Shared facts

valid transaction, with the state of the promissory note showing only an amount of 50 euros. At the same time, the old earlier valid fact is historicized (Fig. 2). Thus, over time, a chain of shared facts is created.

### Signer

In order for a new contract to emerge, the contracting parties must agree to the new state - just as they would do in the analog world. If someone wants to change a contract, all parties must sign again. A digital signature therefore means that all agree that a new fact may arise. A signer denotes any party that has a right to a say here.

### Transaction

A transaction changes the state of particular circumstances. Roger is now only 50 euros in debt and no longer 100 euros. For such a transaction to be valid, it must

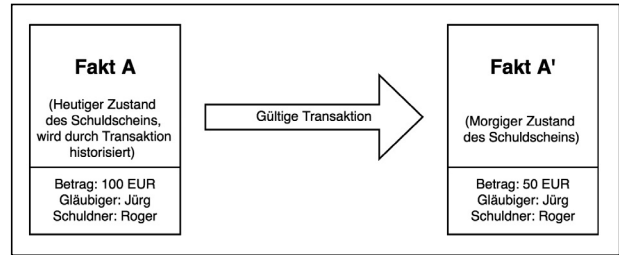


Fig. 2: Evolution of a shared fact

follow a specific contract and comply with all the rules established in the contract. A transaction in Corda can have inputs and outputs. The input is a valid and not yet historicized shared fact. If all parties defined as signers agree to a transaction, the result is a new, valid fact, provided that the Notary Service does not have a veto right. Further information on the Notary Service can be found in the „Double Spending Problem” box. The new fact is immediately valid, unlike Bitcoin, where the transactions are collected in blocks. In Bitcoin, a transaction is valid only if the block is in the longest chain in which the transaction resides.

As described above, a transaction input that references an earlier transaction output may not yet be historicized. Each input may only be used once. As an example, a virtual coin from the same owner may only be issued once. Otherwise, one euro could suddenly become two euros, which is referred to in the blockchain environment as the „Double Spending Problem“.

### Contracts and rules

For a transaction to be valid, it must adhere to a predefined contract. For example, this contract defines how many inputs and outputs a transaction has, what type it can be, or who need to sign the transaction. This is referred to as a “Contract” in Corda and must be defined in advance between the contracting parties. This contract is stateless and only describes the rules of how

### Double Spending Problem

A transaction can generate new outputs called „Unspent Transaction Outputs” (UTXO). These outputs can be reused as input in a new transaction. However, a given UTXO may only ever be used as input in one single transaction, otherwise we have a double-spending problem [4]. So the problem is how to prevent this. There are two approaches to this: centralized and decentralized.

The decentralized approach was chosen for Bitcoin because in principle you do not trust a central office in this network and you do not want to have a „single point of failure”. To achieve this, the proof-of-work consensus algorithm is used. Such a system is mistakenly called the Trustless Transactional System [5], which of course is not entirely true because we trust the miners who have an incentive not to cheat the system.

Also in the case of Corda, you must make sure that a used UTXO is not used as input in several transactions. This happens in Corda with a Notary Service, which checks in an existing transaction whether the input has not already been used in an earlier transaction and has therefore already been historicized. It isn't until the Notary Service has signed an existing transaction that it is considered completed, and the new shared fact is saved with the participants. In most cases, the Notary Service is not just a single node, but a whole pool of Notary Nodes is created, which are each provided by different participants. Nevertheless, Corda has a centralized approach and you have to be able to trust the notary pool.

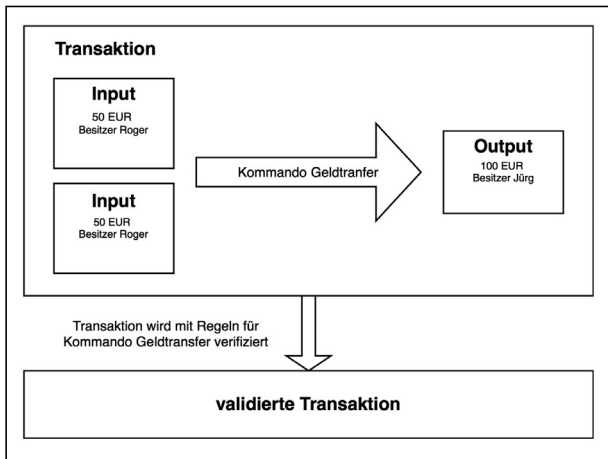


Fig. 3: Command of a transaction

facts can generate new facts. In the example presented above (100 euros debt) Jürg would have to agree that after the transaction, Roger is only 50 euros in debt. The creditor is therefore defined in the contract as a required signer. If a transaction fulfills all the rules in the contract and all signers of the transaction agree, a new shared fact, also known as an output, is created. This output is now stored in a database for all contracting parties. Corda also refers to this location to store current and historicized facts as a Vault. In a later transaction, this output could again serve as input.

### Command of a transaction

A particular state (shared fact) of a contract can usually be changed by various influences. Imagine a digital coin. This coin must be generated by someone and is assigned an owner. Later, you want to transfer this coin to someone and receive a physical product in return. From this, you can already derive two transaction types. The purpose of the first transaction type would be money creation, and of the second - money transfer. In Corda, this purpose is referred to as *Command Data*, and we derive our own commands from that. Thus, a transaction has one or more commands, and the rules to be verified in the contract may differ depending on the command. It would be difficult to determine what a transaction is doing only by looking at the inputs and outputs of the transaction. Thanks to the command though, the intention is clear.

Figure 3 shows a transaction with the money transfer command. Depending on the command, the rules are now checked. For example, such a transaction must have at least one input. Here in the example we have two, which is also possible, because 50 euros times two also gives us 100 euros again. When validating the transaction, you need to make sure that Roger is the owner of the inputs and that the transaction has never been used. The output must have a different owner than the inputs. The contract should additionally define that the amount of the output is the same as the sum of the amounts of the inputs.

### Flows

Sitting at the heart of Corda are the flows, which summarize all the necessary steps until a new shared fact can be written into the vaults of the network nodes. A transaction must be created, validated and signed by a contract partner. Thereafter, it is necessary to make this transaction available to the other contract partners via the network, so that it can be validated and signed there as well. Once all have agreed to the transaction, the transaction is also sent to the Notary Service, which verifies that there is no Double Spending. If everything went right, the new shared fact will be stored with all contract partners. Flows can be freely defined depending on the application, so all necessary business relationships can be mapped in Corda. When writing a flow, the developer does not have to deal with things like concurrency. A flow is written sequentially and does not have any callback functions or the like. To achieve this, Corda uses the Quasar Library [6], which provides lightweight threads (so-called Fibers [7]). All network challenges are abstracted for the Corda developer, and so the developer can focus on the business logic. To write a flow, the *flow Logic* class is extended.

### Listing 1: Implementation of the „CouponState” fact

```
public class CouponState implements ContractState {

    private CouponSubject couponSubject;
    private final Party issuer;
    private final Party owner;
    private final int value;

    public CouponState(CouponSubject couponSubject, Party issuer, Party
owner, int value) {
        this.couponSubject = couponSubject;
        this.issuer = issuer;
        this.owner = owner;
        this.value = value;
    }

    @NotNull
    @Override
    public List<AbstractParty> getParticipants() {
        return ImmutableList.of(issuer, owner);
    }

    public enum CouponSubject {
        REAL_KITTY,
        CRYPTO_KITTY;
    }
}
```

## Corda ContractState, Contract, CommandData and FlowLogic in action

On a small example, we will now show how all these parts merge into a whole. Here, we are creating coupons and giving them away. But the coupons do not necessarily have to be accepted by the recipient. Maybe the recipient has no interest whatsoever in getting a coupon for a Crypto Kitty [8]. On the other hand, the recipient would gladly accept a coupon for a real cat.

First, a *coupon State* will be created that implements a *Contract State* (Listing 1). As a reminder: A *Contract State* is a fact that can be shared and therefore become a shared fact.

*Coupon State* has four fields. The field *couponSubject* describes what this coupon can be redeemed for, and *couponSubject* can accept the values *REAL\_KITTY* or *CRYPTO\_KITTY*. The field *issuer* refers to the party that had created the coupon. The field *owner* references the

owner of the coupon, while *value* depicts the value of the voucher. The *getParticipants* method is prescribed by the interface and must return all parties which have an interest in this *CouponState*. These parties ultimately store the *coupon State*, Now the contract *CouponContract*, referred to in Corda as *Contract*, is created (Listing 2).

*coupon Contract* must implement the *verify* command. If something is not compliant, an exception will be thrown. This means that the transaction is invalid and does not abide by the contract that was shared with all parties at the beginning. Each party validates an incoming transaction against its own copy of *CouponContracts*. If no exception is thrown, the transaction is valid as long as it is also accepted by the Notary Service as valid. As described above, separate rules may apply depending on the command, and the transaction must be signed by other parties.

Last but not least, the business logic must be implemented. Questions pop up, like: Who creates a transaction? Who does it need to be sent to for signing? Here in

### Listing 2: Implementation of the „CouponContract” Contract

```
public class CouponContract implements Contract {
    public static String ID = "example.CouponContract";

    @Override
    public void verify(@NotNull LedgerTransaction tx) throws
    IllegalArgumentException {
        List<CommandWithParties<CommandData>> commands =
        tx.getCommands();

        if(commands.size() != 1) {
            throw new IllegalArgumentException("must have one command");
        }

        if(commands.get(0).getValue() instanceof Commands.Issue) {
            validateRulesForIssueCommand(tx);
        } else {
            throw new IllegalArgumentException("only issue command is
            supported");
        }
    }

    private static void validateRulesForIssueCommand(LedgerTransaction tx) {
        List<ContractState> inputs = tx.getInputStates();
        List<TransactionState<ContractState>> outputs = tx.getOutputs();
        List<CommandWithParties<CommandData>> commands =
        tx.getCommands();

        if(inputs.size() != 0) {
            throw new IllegalArgumentException("must not have input states, the
            coupon will be issued here");
        }

        if(outputs.size() != 1) {
            throw new IllegalArgumentException("must have one output state");
        }

        if(tx.outputsOfType(CouponState.class).size() != 1) {
            throw new IllegalArgumentException("transaction output must be a
            coupon state");
        }

        CouponState couponState = (CouponState) tx.getOutput(0);

        if(couponState.getValue() < 1) {
            throw new IllegalArgumentException("coupon state value must have a
            positive value");
        }

        Party issuer = couponState.getIssuer();
        Party owner = couponState.getOwner();
        List<PublicKey> signers = commands.get(0).getSigners();

        if(!signers.contains(issuer.getOwningKey())) {
            throw new IllegalArgumentException("issuer must be a required
            signer");
        }

        if(!signers.contains(owner.getOwningKey())) {
            throw new IllegalArgumentException("owner must be required signer");
        }
    }
}

public interface Commands extends CommandData {
    class Issue implements Commands { }
}
```



**Listing 3: Implementation of the „CouponIssueFlow” flow**

```

@InitiatingFlow
@StartableByRPC
public class CouponIssueFlow extends FlowLogic<SignedTransaction> {

    private CouponState.CouponSubject subject;
    private final Party owner;
    private final int value;

    public CouponIssueFlow(CouponState.CouponSubject subject, Party owner,
int value) {
        this.subject = subject;
        this.owner = owner;
        this.value = value;
    }

    @Suspendable
    @Override
    public SignedTransaction call() throws FlowException {
        Party notary = getServiceHub().getNetworkMapCache().
getNotaryIdentities().get(0);
        Party issuer = getOurIdentity();

        // create a new transaction output, an input is not necessary when
generating the coupon
        CouponState couponState = new CouponState(
            this.subject,
            issuer,
            this.owner,
            this.value
        );

        // create a new transaction
        TransactionBuilder transactionBuilder = new TransactionBuilder(notary);
        transactionBuilder.addOutputState(couponState, CouponContract.ID);
        transactionBuilder.addCommand(
            new CouponContract.Commands.Issue(),
            issuer.getOwningKey(),
            owner.getOwningKey()
        );

        // check if your created transaction fulfills the rules of the contract
        transactionBuilder.verify(getServiceHub());

        // sign the transaction using the private key of the issuer, the
transaction becomes unchangeable
        SignedTransaction partlySignedTx = getServiceHub()
            .signInitialTransaction(transactionBuilder);

        // Have the transaction signed by the owner
        FlowSession ownerSession = initiateFlow(owner);
        SignedTransaction fullySignedTx = subFlow(
            new CollectSignaturesFlow(partlySignedTx, ImmutableSet.
of(ownerSession))
        );

        // have the transaction authenticated by the Notary Service and store
the new shared fact with all participants
        return subFlow(new FinalityFlow(fullySignedTx));
    }
}

```

**Listing 4: Implementation of the „CouponOwnerFlow” flow**

```

@InitiatedBy(CouponIssueFlow.class)
public class CouponOwnerFlow extends FlowLogic<Void> {

    private final FlowSession counterpartySession;

    public CouponOwnerFlow(FlowSession counterpartySession) {
        this.counterpartySession = counterpartySession;
    }

    @Suspendable
    @Override
    public Void call() throws FlowException {

        // If the initiator requests the signature of the owner for the
transaction, the request must be answered with a SignTransactionFlow.
        class SignTxFlow extends SignTransactionFlow {
            private SignTxFlow(FlowSession otherSession, ProgressTracker
progressTracker) {
                super(otherSession, progressTracker);
            }

            // SignTransactionFlow automatically checks the rules of the contract,
additional things can be checked within checkTransaction
            @Override
            protected void checkTransaction(SignedTransaction stx) throws
FlowException {
                Party identity = getOurIdentity();
                CouponState couponState = (CouponState) stx.getTx().getOutput(0);
                Party outputOwner = couponState.getOwner();
                if (!identity.equals(outputOwner)) {
                    throw new IllegalArgumentException("we must be owner of the
coupon");
                }

                if (couponState.getCouponSubject().equals(
                    CouponState.CouponSubject.CRYPTO_KITTY)) {
                    throw new IllegalArgumentException("we don't like crypto kitty
coupons");
                }
            }
        }

        subFlow(new SignTxFlow(this.counterpartySession, SignTransactionFlow.
tracker()));
        return null;
    }
}

```

our coupon example, it's quite easy. You only need two parties to create the coupon: the creator and the first owner of the coupon. Listing 3 shows how the coupon is initiated with a transaction by the creator.

The `@InitiatingFlow` annotation states that the `Coupon Issue Flows` class can launch communication with a business partner. `@StartableByRPC` means that you can start this flow via RPC. Entering `@Suspendable` with the `call` method makes it possible to put the execution into hibernation, for example if the business partner cannot answer right now. A new `CouponState` is generated within the method which is passed to a new transaction as output. This output can only be generated if the created transaction adheres to the rules of the contract, which is submitted via `CouponContract.ID`. No class as such is submitted in this case, just an ID. Every business partner should check this with his or her own copy of the class. After creating a transaction, it is validated by the initiator and then signed. Then the transaction must be submitted to the business partner - in this case the owner of the voucher - for signing. The owner could also not agree with the transaction. If the owner does agree, the transaction is

authenticated by the Notary Service and the new shared fact is stored with all business partners.

After the initiator has been implemented, the flow of the business partner, or in this case the new owner of the coupon, must be implemented (Listing 4).

The `call` method of the `CouponOwnerFlow` class is called when `CouponIssueFlows` requires signatures. Please note the `@InitiatedBy (CouponIssueFlow.class)` annotation in the process. The response to that is the `SignTxFlow` class which is derived from `SignTransactionFlow`. The overwritten `check transaction` method is used to examine the transaction more closely and reject it if something does not fit the expectations. In the example above, only those coupons should be accepted that are actually intended for this contracting party. The other transactions are not signed. Also, no coupons are accepted for the purchase of a Crypto Kitty. Once we have signed, the flow continues from the initiator.

## Conclusion

You can rightly wonder: Where the hell are the blocks? The fact is: Corda does not collect transactions in blocks and does not use a proof-of-work consensus algorithm either. Since there are no blocks, there are also no references to an earlier block. Therefore, Corda is not really a proper blockchain in the true sense. The shared facts are chained, though. The evolution, as it flows from one fact to the next, is quite clear and understandable here.

Corda has great potential because it can easily map contracts that used to be paper-based and had to be laboriously signed by each of the contracting parties by hand. We are spared of all the sending back and forth of paper. As a result, the process should accelerate drastically. Corda provides the developer with all the features needed to do this easily while simplifying network communication.

After a successful transaction, all the required parties have agreed to a fact. This state of the contract is stored in the vault of each respective party.

## SESSION Cash on Ledger – issuing Cash on R3 Corda for (M2M)- Payments in the Industry



THOMAS KRECH UND  
VITALIJ REICHERDT



This presentation will provide a look into the architecture of the R3 Corda framework, a ledger solution especially for the financial world, by giving a concrete example use case from the financial world: cash on ledger. The

first part of the session will provide an overview of the philosophy behind Corda, its components, process flows and more. In the second part we show how we connect traditional bank payment systems with the distributed ledger world and the business rationale behind it. The following questions will be answered: How can Euro securely be issued on ledger, and used for payments? What is the rationale behind it? Which technical and regulatory requirements and problems have been solved? Specifically, we will show:

- who is R3
- what is Corda and why is Corda different
- what are the main components
- contract, states and flows
- transactions and consensus
- network and architecture
- what is cash on ledger and why we need it
- which problems (technical and regulatory) are to be solved
- what is our solution
- Commerzbank API



**Christian Koller** is Senior Consultant at INNOQ Schweiz GmbH. He is primarily active in the backend area and does programming using Java or Scala. His focus is on web applications, microservices and, more recently, blockchain technologies.

## Links & literature

- [1] <https://www.r3.com>
- [2] <https://tools.ietf.org/html/rfc5280>
- [3] <https://www.blockchain.com/explorer>
- [4] <https://en.wikipedia.org/wiki/Double-spending>
- [5] <https://medium.com/@preethikasireddy/eli5-what-do-we-mean-by-blockchains-are-trustless-aa420635d5f6>
- [6] <http://docs.paralleluniverse.co/quasar>
- [7] <https://zeroturnaround.com/rebellabs/what-are-fibers-and-why-you-should-care>
- [8] <https://www.cryptokitties.co>