Ecosystem for COllaborative Manufacturing PrOceSses – Intra- and Interfactory Integration and AutomaTION
(Grant Agreement No 723145)

D4.3 The COMPOSITION Blockchain

Date: 2019-02-28

Version 1.0

Published by the COMPOSITION Consortium

Dissemination Level: Public

Co-funded by the European Union's Horizon 2020 Framework Programme for Research and Innovation under Grant Agreement No 723145

Document control page

Document file: D4.3 The COMPOSITION Blockchain.docx
Document version: 1.0
Document owner: CNET

Work package: WP4 – Secure Data Management and Exchange in Manufacturing
Task: T4.2 – Log-oriented architecture design and Implementation of a secure and trusted blockchain
Deliverable type: OTHER

Document status: ☒ Approved by the document owner for internal review
☒ Approved for submission to the EC

Document history:

Version	Author(s)	Date	Summary of changes made
0.1	Mathias Axling (CNET)	2016-12-23	Initial TOC
0.2	Mathias Axling (CNET)	2019-01-09	Updated TOC
0.3	Mathias Axling, Fabian Johannsen (CNET)	2019-01-15	Bitcoin, Blockchain background
0.4	Mathias Axling (CNET)	2019-02-11	Additional content
0.5	Rodrigo Díaz, Nacho González, David Rojo, Mario Faiella, Javier Romero (ATOS)	2019-02-19	ATOS contributions
0.6	Mathias Axling (CNET)	2019-02-25	Edits
0.9	Mathias Axling (CNET)	2019-02-26	Ready for peer review
1.0	Mathias Axling (CNET)	2019-02-28	Final version submitted to the European Commission

Internal review history:

Reviewed by	Date	Summary of comments
Thomas Kreuzer (FIT-WI)	2019-02-27	Approved with minor comments.
Alexandros Nizamis (CERTH)	2019-02-27	The document is good and address completely the subject of the corresponding Task. Some formatting issues have to be fixed, an abbreviation table is missing and the Conclusions can be extended.

Index:

1 Executive Summary 4
2 Abbreviations and Acronyms 5
3 Introduction 6
3.1 Purpose, context and scope of this deliverable 6
3.2 Content and structure of this deliverable 6
4 Background 7
4.1 Blockchains Overview 7
4.2 Applications 8
4.3 Block generation 8
4.4 Permissioned and Permissionless Blockchains 9
4.5 Distributed Consensus 10
4.5.1 Public Blockchain 11
4.5.2 Private Blockchain 12
4.5.3 Consortium Blockchain 13
4.6 Smart Contracts 13
5 The COMPOSITION Blockchain 14
5.1 Concerns 14
5.1.1 Quality attributes 14
5.2 Design decisions 14
5.2.1 Blockchain implementation - Multichain 14
5.3 Blockchain API 16
5.3.1 Functional View 16
5.3.2 Deployment View 17
5.3.3 Information View 18
6 Applications in COMPOSITION 20
6.1 Public Key Infrastructure (PKI) 20
6.4 Location tracking 23
6.5 IPR protection 23
6.6 Supply chain logistics demo 25
6.7 Confidential streams 25
7 Conclusions 27
8 Annex 1: Background - Bitcoin 28
9 List of Figures 30
9.1 Figures 30
10 References 31

1 Executive Summary

This deliverable presents the results of task 4.2 “Log-oriented architecture design and Implementation of a secure and trusted blockchain”. It describes the design and applications of the COMPOSITION blockchain.

The main concerns for the blockchain is to provide an audit trail for manufacturing and supply chain data, enabling both product data traceability and secure access for stakeholders. The implementation mechanism chosen for the blockchain was Multichain[1], an open-source blockchain implementing the Bitcoin API[2]. It provides configurable permissions for assets and consensus, high transaction speeds and several useful abstractions for dealing with general time-stamped data without the need to explicitly use cryptocash or other assets.

Several applications of blockchains have been tested in the COMPOSITION project, mainly with a supply-chain focus.

The COMPOSITION blockchain directly corresponds to COMPOSITION Technical Objective 2.1: “Design and implement a Log Oriented Architecture”.

[1] https://www.multichain.com/
[2] https://bitcoin.org/en/developer-reference

2 Abbreviations and Acronyms

Acronym	Meaning
API	Application Programming Interface
BTC	Bitcoins
CXL	COMPOSITION eXchange Language
IPR	Intellectual Property Rights
JSON	JavaScript Object Notation
JWE	JSON Web Encryption
JWK	JSON Web Key
JWS	JSON Web Signature
PKI	Public Key Infrastructure
RAM	Random Access Memory
REST	Representational State Transfer
RPC	Remote Procedure Call
WP	Work Package

3 Introduction

3.1 Purpose, context and scope of this deliverable

This deliverable describes the design and implementation of the COMPOSITION blockchain component, performed in task 4.3 of WP 4. It provides an overview of the context and purpose of the component in the COMPOSITION system, the requirements and quality attributes that has guided the design, the rationale for the design decisions taken and the applications in the COMPOSITION system. A brief background and overview on blockchain technology is included. For details on digital signatures, hashing and related standards, we refer to (COMPOSITION D4.2).

3.2 Content and structure of this deliverable

Diagrams and some content may be repeated from other WP4 deliverables. The remainder of the document is structured as follows:

Section 4 - Background: provides an overview of blockchain concepts and possible applications.

Section 5 - The COMPOSITION Blockchain: describes the design of the composition blockchain and the blockchain API.

Section 6 - describes suggested, designed and implemented applications of blockchains in COMPOSITION.

Section 7 - Conclusions: presents a summary of the deliverable, conclusions from the development of the component and how future work will proceed.

4 Background

4.1 Blockchains Overview

Blockchains was first described for use in the cryptocurrency Bitcoin[3] (Nakamoto, 2008), but has since found many other applications. A blockchain consists of a growing list of linked elements. Each element, called a block, contains transaction data, the hash of the previous element and a timestamp. This makes it easy to verify the integrity of the entire chain. The chain is typically distributed to and replicated by all parties in a peer-to-peer network. The consistency of the replicas is upheld by a consensus algorithm. Blockchains are increasingly used as a distributed shared transaction ledger, without the need for a trusted third party to manage access. It is a database that can be shared for writing across boundaries of trust.

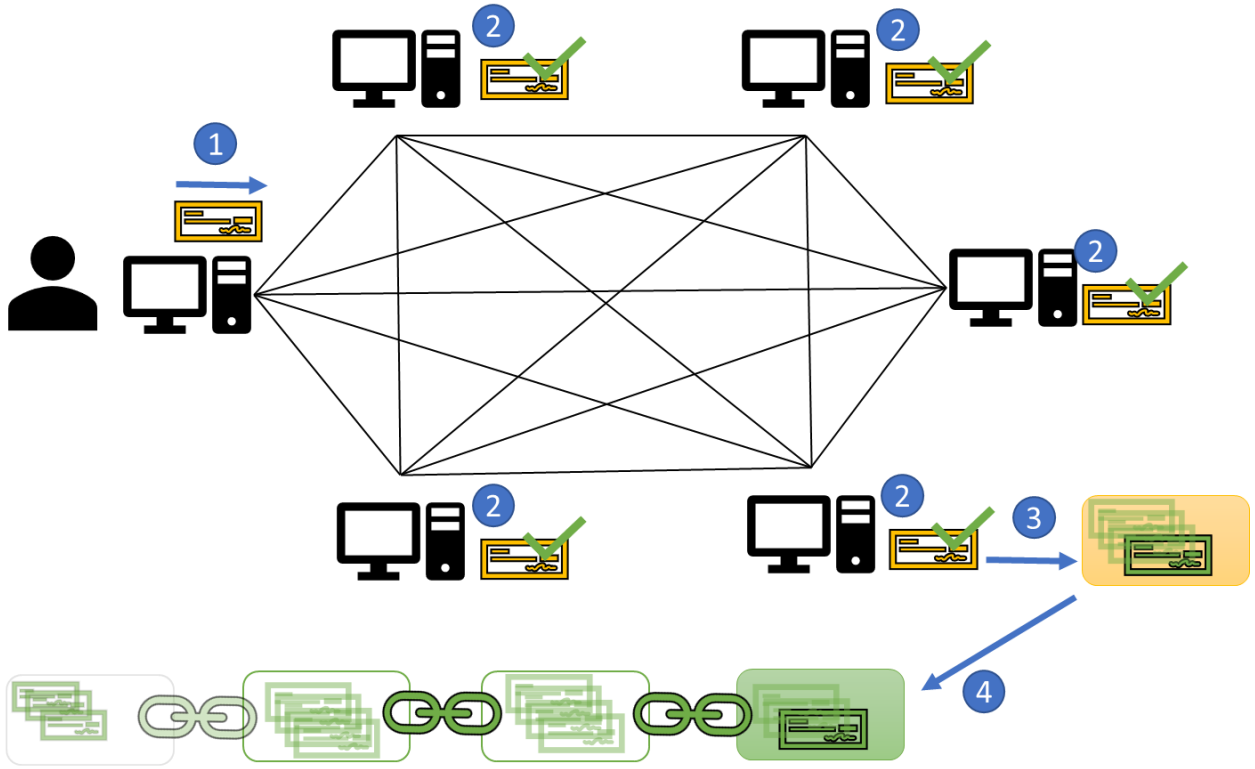

Figure 1: Simplified blockchain

In Figure 1, a participant in the network sends a transaction to the blockchain network (2). Any participant in the network can independently verify that a transaction to be included in a block is valid by examining the contents of the transaction according to the rules of the protocol[4] (2). The transactions are signed by the issuer and so provide proof of authorization. A transaction that is not valid will be rejected by all participants independently, no central authority is needed. Sets of valid transactions are collected into candidate blocks by different participants (3). The consensus algorithm is used to make one candidate block the next block in the chain, linked to the preceding block by containing a hash of that block. The new block is propagated to all nodes in the network. The following sections will describe block generation and consensus in more detail.

Today blockchains is being applied in many domains (Boucher, 2017), including supply chain management[5], electronic health records[6], land registry[7], product tracking and traceability[8] and of course different types of financial services[9].

[3] Bitcoin is described in more detail in Annex 1.
[4] Which are sometimes configurable, see smart contracts and smart filters.
[5] https://www.ibm.com/Watson/Supply_Chain
[6] https://www2.deloitte.com/content/dam/Deloitte/us/Documents/public-sector/us-blockchain-opportunities-for-health-care.pdf
[7] https://www.gov.uk/government/news/hm-land-registry-to-explore-the-benefits-of-blockchain
[8] https://www2.deloitte.com/content/dam/Deloitte/lu/Documents/technology/lu-blockchain-internet-things-supply-chain-traceability.pdf
[9] https://advisory.kpmg.us/content/dam/advisory/en/pdfs/blockchain-future-finance.pdf

The main benefits of blockchains are: 1) they provide a database designed to be secure, 2) the robustness from being decentralized and replicated - transaction validation and consistency in the blockchain is not dependent on any specific node – and 3) that it can be applied in an environment of limited or distributed trust, without the need for trusted third parties.

However, the blockchain design also entails limitations compared to other databases, mainly, performance and scalability. The consensus algorithm will require more time to make the database consistent than a centralized algorithm (and in the case of Bitcoin proof-of-work, a lot of processing power).

Data confidentiality can also be a problem. Most blockchains are designed for transactions to be readable and verifiable by any party. Even blockchains where the access to the chain is restricted do not have fine-grained access control or encrypted transactions.

If data confidentiality is a vital requirement and the need for distributed consensus (trust) is low, e.g. in the presence of a trusted third party or a central administrator of a private blockchain (as in the intra-factory value chain), a regular database may be a better choice. If the intra-factory value chain is connected to supply chains, however, there may be benefits from using a common infrastructure.

Storing larger data, such as images and scanned documents, is also a task that blockchains are not designed for. One solution is to store a hash of the data together with a link to another storage media, e.g. the web, a peer-to-peer file network or a document database. This allows us to verify that the data is correct while keeping the on-chain data a manageable size. However, ensuring that the linked document is still there, reachable and keeping the off-chin data in sync is a problem that must be managed.

When using blockchains as an immutable distributed store for information generated outside the blockchain, there is also always the problem of verifying that the information put in the blockchain is correct to begin with.

4.2 Applications

Some main groups of commercial applications[10] that we have identified are:

- Financial systems (transfer of rights and values)
    - Cryptocurrencies, e.g. Bitcoin, Litecoin[11], Ripple[12], Dash[13], Ethereum[14]
    - Financial services
        - Trading Platforms: exchanges using blockchains
        - Smart contracts for bonds, allowing instant settlement.
        - Peer-to-peer lending
- Provenance tracking (certifying a fact)
    - Health data and identity management
    - Secure digital product memories
    - Land registry
- Inter-company audit trails (tracing the origin and validating every step in a chain of actions)
    - Supply chain traceability
    - Fair trade certification
    - Secure medical supply chain process
    - Food traceability
    - Cold chains in food and medical logistics

4.3 Block generation

Transactions are broadcasted to all participants in the network. Any participant, from here referred to as “node”, participating in the consensus process in the network can take a set of unconfirmed transactions (TXn) and put them in a block. This block will later on be put on the chain of blocks containing confirmed transactions and then broadcasted to everyone in the network. Common for all blockchains is that the blocks need to contain a reference value so that the next block in the chain can be linked to its predecessor. This is done by calculating a conjoint hash value with all the metadata in the block as input to the hash function. Hashing all transactions creates a Merkle root (Merkle, 1980) which is visualised in Figure 2.

[10] Not including CryptoKitties (https://www.cryptokitties.co/)
[11] https://litecoin.org/
[12] https://ripple.com/
[13] https://www.dash.org/
[14] https://www.ethereum.org/

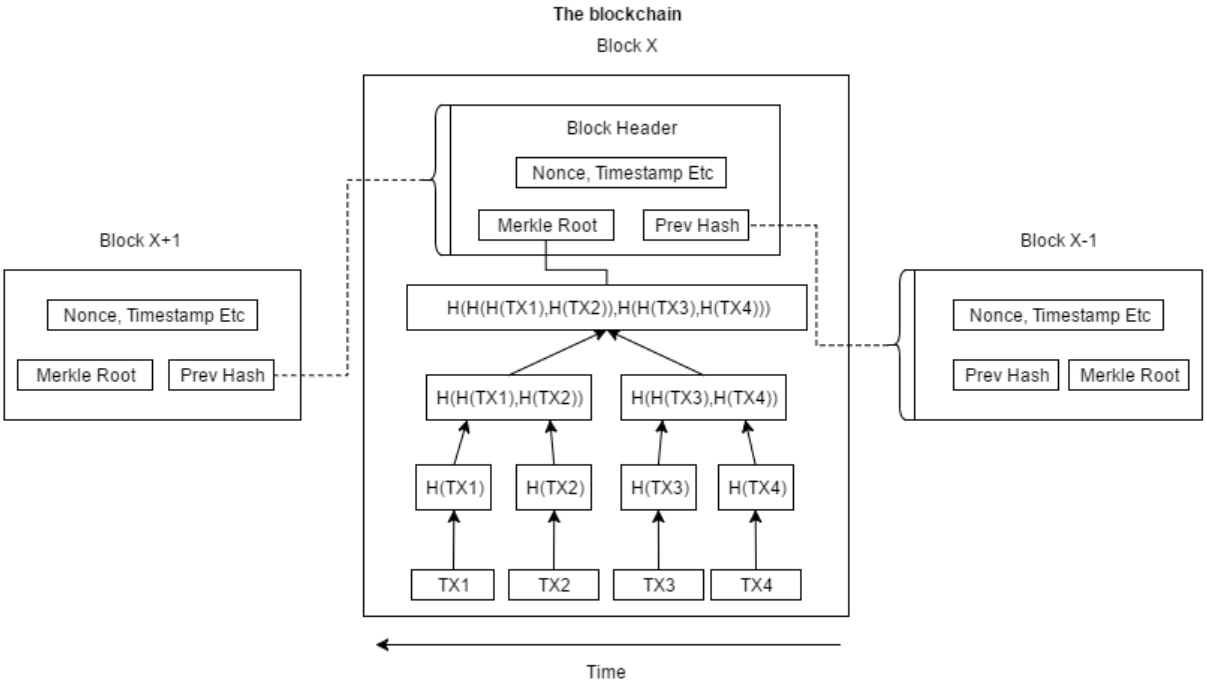

Figure 2: Block design

In Figure 2 the block number *X* has registered four transactions (TX1,2,3,4). These transactions are hashed, H, to get a transaction hash value. These values are concatenated, and the concatenated values are hashed. This continuous when you only have one hash value - the Merkle Root. The Merkle Root, the previous hash (the hash for block X-1) and some other values make up as the previous hash value for X+1, which is the next block in the blockchain. This is what links the blocks to each other making it a chain of blocks.

What the block header contains and exactly how the Merkle tree is used differs from what blockchain you use, but the figure above represents the main idea of the usage of the Merkle Tree algorithm.

There is a valid reason why you hash the transactions in a tree-like structure, and not just bundles all transaction to get a hash value out of that. The answer is Merkle proof, which is an efficient way of authenticating a hash value that could be extended to also authenticate a large data set of hash values. Let's say that person A wants to prove to person B that H(TX1) is in the Merkle tree in the above figure. The only value available for person B is the Merkle Root. To prove H(TX1) existence in the tree, person A only needs to provide the siblings from H(TX1) to the root node. In this case person A only needs to provide the value for H(TX2), the right child of the root node and the root node itself to prove H(TX1) existence. This is done without the need for all the transactions, which would have been necessary if all transactions were bundled and then hashed.

The Merkle proof is useful because one can use a “thin client” or a “lightweight client” which only need to check the block headers for verifying transactions, instead of downloading every single block in the chain. For these clients the growth of the chain is linear because they are not dependant on how many transactions there is in a block. This also makes the chain relatively scalable.

4.4 Permissioned and Permissionless Blockchains

A blockchain may be permissioned with regard to which parties are allowed to join the blockchain and issue transactions. In Bitcoin, anyone may create a wallet and start trading bitcoins with other wallets. On a permissioned chain, there is centralized control of who is allowed to join the chain.

Bitcoin is what is called *permissionless* – it allows any node to join the blockchain, issue transactions and view all other transactions. The identities of the participants are pseudonymous or anonymous. This is not desirable in all applications, so a growing number of blockchain implementations allow for *permissioned* blockchains, where participants are known by the blockchain validator and explicitly allowed the rights to see or issue transactions.

4.5 Distributed Consensus

As for any distributed database, there are difficulties to achieve consensus between nodes in the network. In the blockchain technology to achieve consensus is the process of deciding the next block in the chain. Consensus algorithms are mechanisms that are used to achieve agreement on a single data value, and thus obtain reliability in a network that can involve unreliable participants. In particular, the network needs to reach consensus on a single state while dealing with participants that do not only fail to send values but send unreliable and conflicting information[15] to different parts of the system. This problem was initially presented as the "Byzantine Generals Problem" in (Lamport, 1982), with conditions for reaching consensus given the relative number of unreliable participants, the availability of signed or unsigned messages and connectivity of the network. A system that solves this problem exhibits Byzantine Fault Tolerance; the consensus algorithms for blockchains need to do this due to the high economic incentive for corrupting the system. This is a large field, and this section will only provide an overview on approaches to distributed consensus relevant to the choice of implementation mechanism for the COMPOSITION Blockchain.

In bitcoin (Nakamoto 2008), the proof-of-work mechanism is used to avoid double-spending of bitcoins; issuing multiple (internally valid) transactions using the same funds. Any participant that has the opportunity to decide on the next block on the chain could issue the same payment to two other participants and just include the transaction they want. As described in Annex 1: Background - Bitcoin, validating the next block involves finding a number that combined with the block produces a hash with a number of leading zeroes. This problem is kept hard enough (by adjusting the number of required leading zeroes) so that any party wanting to double-spend bitcoins would have to control a majority of the network's computing resources.

However, there is also an incentive for participants to connect their computer to the network and spend resources on solving mathematical puzzles. In Bitcoin's proof-of-work consensus algorithm, the incentive for someone to maintain the security of the blockchain is that they get paid a certain amount – successful mining generates bitcoins. This also makes it more profitable for any partner with sufficient computing resources to maintain the integrity of the chain and generate valid blocks than to double-spend resources.

Proof-of-work consensus is time- and resource-consuming. A widely used alternative is proof-of-stake. In this scheme, the chance of being picked by the network to create the next block is proportional to some kind of stake in the system, e.g. number of coins owned and sometimes also to the time that the participant has held the coins. In some variants, the assets serve as collateral for the block generation process, with a cost associated to generating blocks with fraudulent transactions. Combined with at transaction fee for the validation process it creates the incentive not to compromise the blockchain.

Delegated proof-of-stake involves selecting block validators in a decentralized voting process where votes are distributed according to proof-of-stake. A reputation system helps the system participants choose delegates. The delegated block validators may also have to put assets as collateral which is forfeited if malicious behaviour is detected. Delegated validators will collect the rewards for block generation. The incentive for being a delegated block validator will create competition to be chosen and promote good behaviour.

There are several other algorithms that have been proposed. However, there are applications where one or several trusted parties that have set up the blockchain network and can be trusted to validate new blocks. (One might call this implicit proof-of-stake.) In this case, a round-robin or randomized selection process will be used without any algorithm for distributed trust or any underlying asset being exchanged. This is the case for many of the applications considered for the COMPOSITION blockchain.

We divide blockchain applications into three categories depending on how which participants are allowed to validate (a.k.a. "mine") blocks.

[15] Thus behaving in a byzantine manner: "of, relating to, or characterized by a devious and usually surreptitious manner of operation". https://www.merriam-webster.com/dictionary/Byzantine

4.5.1 Public Blockchain

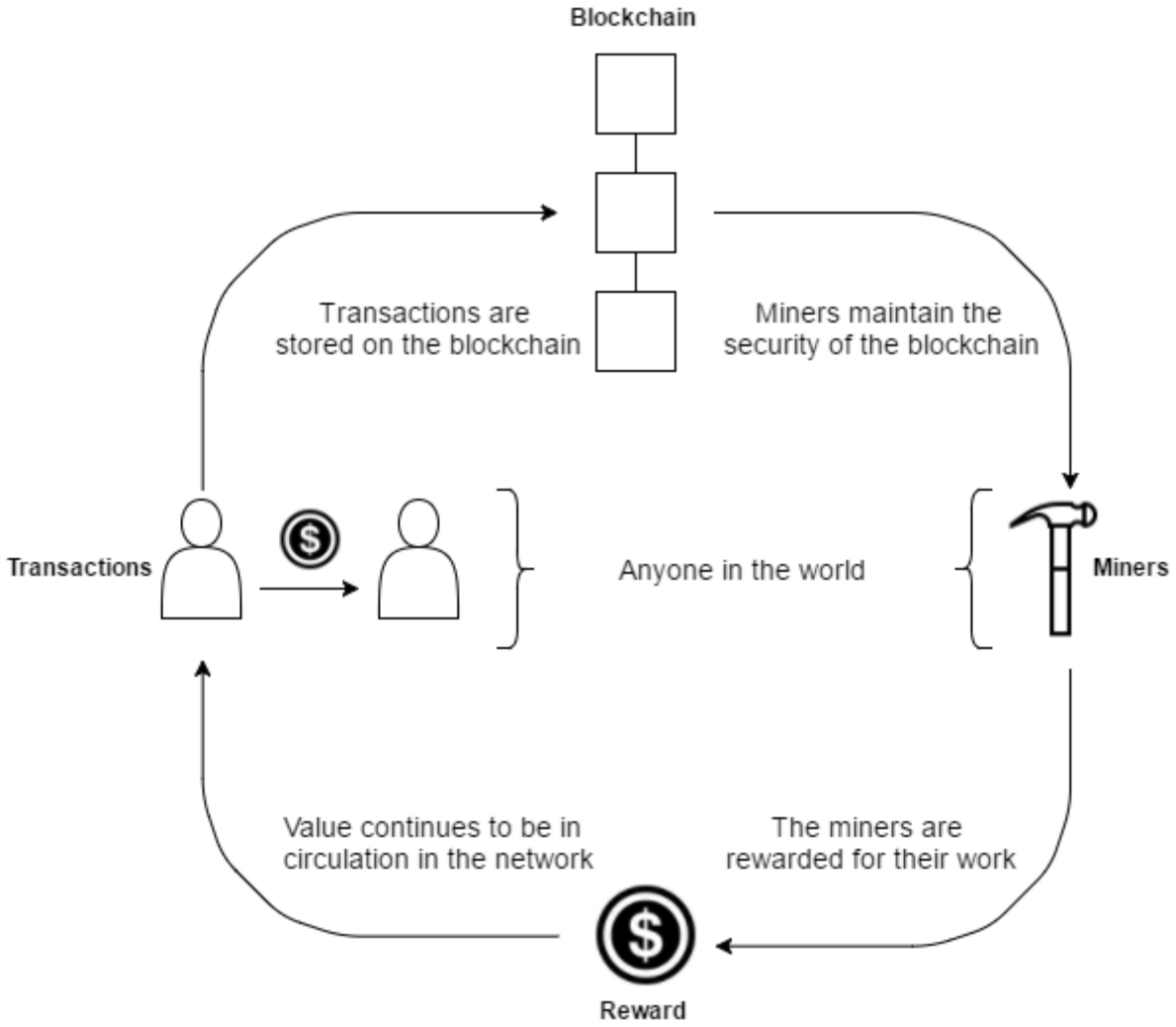

Figure 3: Public blockchain

On a Public blockchain anyone can join as a validator (a.k.a. "miner") of transactions and contribute to the consensus on the correct chain at any time. The protocol itself assures that consensus is reached on valid data only.

Bitcoin is an example of a fully public blockchain. It is a blockchain which anyone in the world can read and interact with. If a transaction is valid, you can expect it to be registered on the blockchain no matter who is the creator, or validator, of that transaction. One of the most important parts of a public blockchain is that anyone could participate in the consensus process, what is called “mining” in bitcoin. Anyone with a computer can be a part of deciding the next ledger state, i.e. the next block on the chain. In public blockchains, an algorithm for distributed consensus such as proof-of-work or proof-of-stake must be used.

4.5.2 Private Blockchain

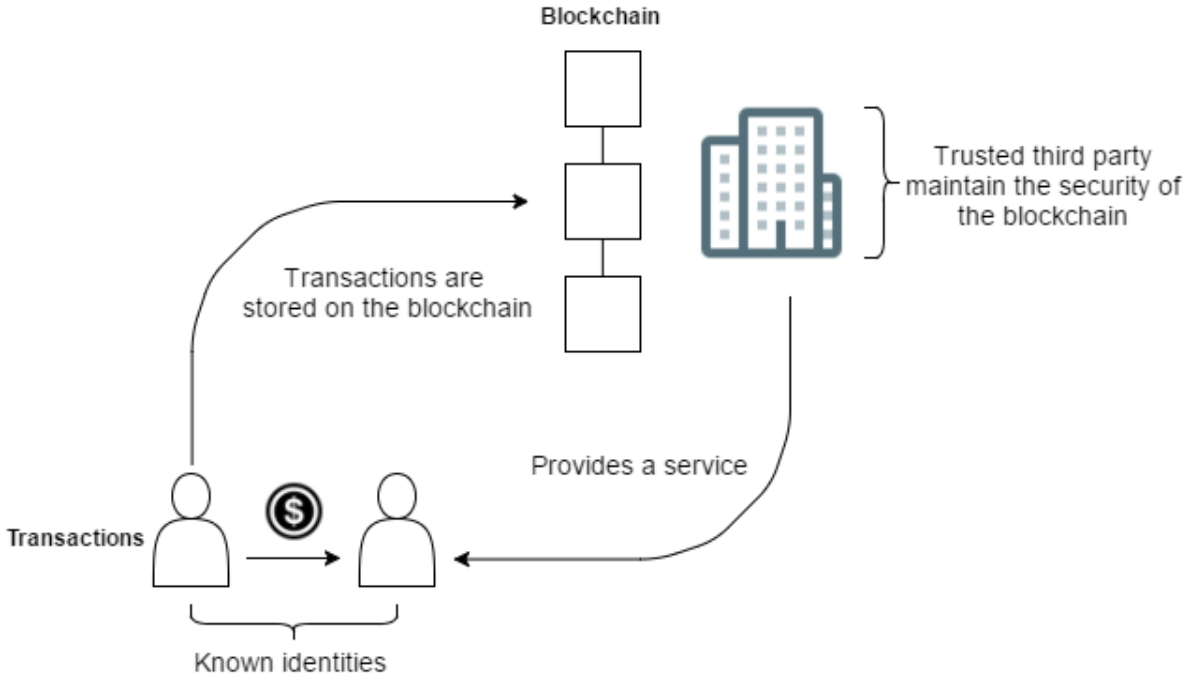

Figure 4: Private blockchain

The opposite of public blockchains are private blockchains. Instead of everyone being able to read and interact with the blockchain, we now have one participant which performs the block-validating operation. A private chain is run and administrated by a single entity, acting as a trusted third party in the chain. Consensus algorithms like proof-of-work are not necessary in this case. Depicted above is a "*permissioned"* private blockchain - participators are permissioned to interact with the blockchain and are therefore known by the blockchain validator.

4.5.3 Consortium Blockchain

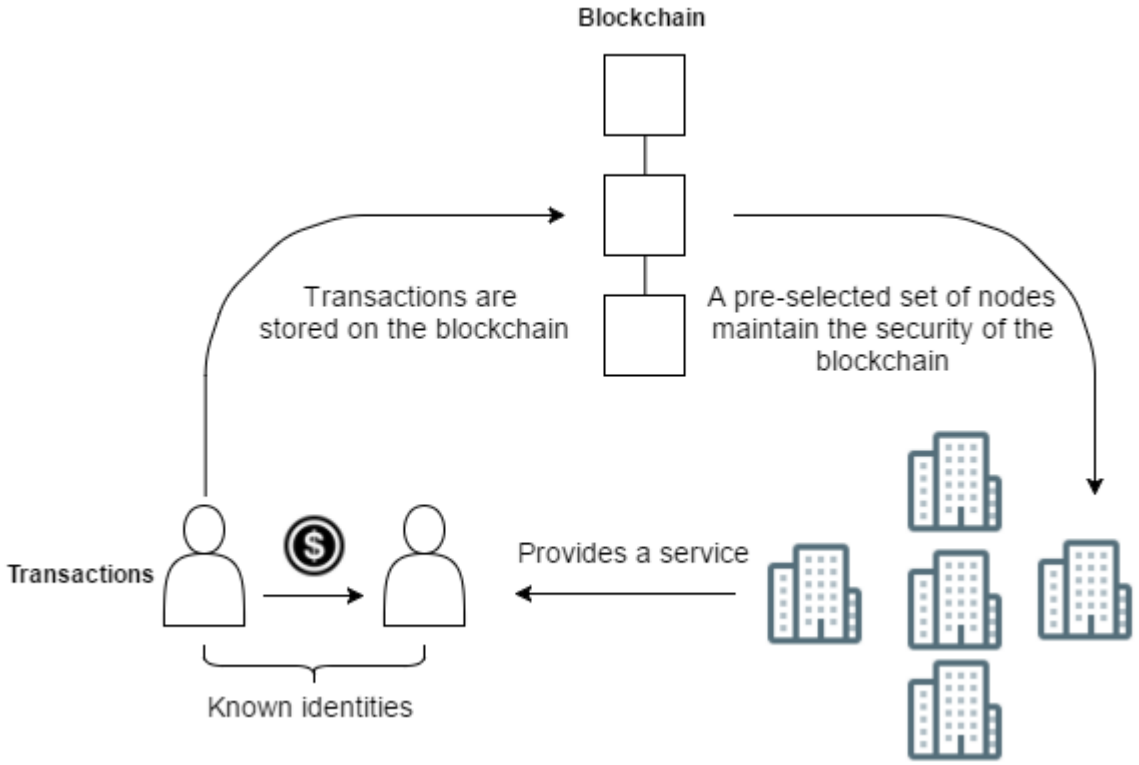

Figure 5: Consortium blockchain

A consortium blockchain has a closed, predefined set of validators that reach consensus on the correct chain. The consensus may be reached using less resource consuming algorithms. This speeds up the process of validating transactions and lowers processing power requirements. The COMPOSITION Marketplaces are a use case for consortium blockchains.

4.6 Smart Contracts

A smart contract is a general-purpose code or script which is stored and executed on the blockchain, the term being made popular by the Ethereum consortium (Buterin 2014). It is an agreement specified in code between an arbitrary set of parties where everything is settled and computed on the blockchain - the code of the contract will be executed when somebody makes a transaction with the smart contract. In Ethereum, a contract is run when a message containing enough fees (ether) is sent to the contract address. The contract will encapsulate the business logic, not only the transaction rules for the digital asset. An example of a smart contract could be a simple if/then statement describing the sale of digital goods: if someone sends a specified amount of assets to a certain address on the chain, then this address must issue a product end user license to the sender of the money.

Smart contracts could be pretty much anything and can reflect any kind of business logic. With smart contracts it is possible build more applications than just transactional ones, and they will still benefit from all the characteristics a blockchain brings. This means that an agreement between parties will be recorded on the blockchain, it will be tamper- and censor-resistant, it can control blockchain assets and it will only be executed by the blockchain, so no one can interfere with the agreement.

Smart contracts may rely on information outside the chain for contract conditions, like the outcome of a football game, the weather conditions, or the current stock price. In such cases, there is no direct way for the smart contract to validate the conditions. For these purposes, trusted off-chain components providing this information in a secure way has been proposed, called “oracles” or “cryptlets” (Gray, 2017).

5 The COMPOSITION Blockchain

The COMPOSITION Blockchain was designed to achieve COMPOSITION Technical Objective 2.1:

> *Technical Objective 2.1: Design and implement a Log Oriented Architecture: based on blockchain technology, ensuring the trusted, secure and automated exchange of supply chain data among all authorized stakeholders, to connect factories and support interoperability and product traceability along the supply chain.*

This section will describe the concerns and design decisions for the Composition Blockchain.

5.1 Concerns

COMPOSITION has three key stakeholder groups: developers, acquirers, and users. The design concerns of these stakeholders are expressed in different form and in different artefacts: the project specification (COMPOSITION, 2016), innovation and exploitation documents, use cases and the requirements (COMPOSITION D2.2).

The main concern for all stakeholders is the management of supply chain data, where multiple stakeholders may be involved over boundaries of trust. Specifically, product traceability along the supply chain and factory interoperability – e.g. the agent communication specifying offers and agreements. User requirements indicate the agent requests and final agreements should be stored, but not all negotiation steps.

Generic functionality for logging any message with the blockchain and intra-factory applications is also mentioned in user requirements. Data to be stored may be quite large for a blockchain, up to half a megabyte.

Logging sensor data or other high frequency, high volume data streams in the platform is not a concern for the blockchain.

Data exchange should be both trusted and secure (confidential).

The supply chain data should be exchanged among all *authorized* stakeholders. COMPOSITION defines multiple marketplaces, which may be open to all participants or closed, by invitation only (COMPOSITION D2.4). This indicates that a permissioned blockchain is needed.

Developer stakeholders need a transparent, extendable, open-source-licensed blockchain platform, that can be further developed and built upon by any partner.

5.1.1 Quality attributes

The quality attributes that were specifically considered were performance and scalability. The blockchain implementation transaction processing speeds must be able to match the data generated by the system, primarily the agent marketplace. Bitcoin confirmation of a block takes on average 10 minutes, which is not sufficient.

5.2 Design decisions

5.2.1 Blockchain implementation - Multichain

After an initial survey and trials of open-source blockchain solutions available at the time[16], the choice was made to build the COMPOSITION blockchain on Multichain.

The choice of blockchain platform for the first feasibility prototype was Ethereum. Contributing to this was the wide acceptance of the Ethereum platform, the programmability (smart contracts in the Solidity language) and the possibility to build consortium solutions. Also considered was taking advantage of the network effects of using the public Ethereum chain and the ether currency. However, the dependency on ether for all transactions and the fact that there was no COMPOSITION use case for smart contracts made us decide against it. Multichain was found to be easier to install and configure, as well as being a better match for the system concerns and COMPOSITION use cases.

[16] Primarily Openchain, Hyperledger, Multichain and Ethereum.

Multichain[17] provides several useful abstractions and methods on top of the Bitcoin API (Greenspan, 2015). As the name suggests, multiple blockchains can be configured and run concurrently on the same node. Blockchains can be permissioned or permission-less and the choice of public, consortium or private mining is configurable. Permissions can be used to control who can connect to the blockchain, issue transactions, create assets and streams (sets of time-tamped data items, see 5.2.1.2), validate blocks ("mine") and administrate the system.

Multichain can be deployed on the following platforms:

- Linux: 64-bit, supports Ubuntu 12.04+, CentOS 6.2+, Debian 7+, Fedora 15+, RHEL 6.2+.
- Windows: 64-bit, supports Windows 7, 8, 10, Server 2008 or later.

Recommended requirements for a node are 512 MB of RAM and 1 GB of disk space. The implementation currently uses Multichain 2.0, which supports 500-2,000 transactions per second.

5.2.1.1 Assets

Multichain can be used to issue multiple custom assets on a blockchain, with the same built-in verification of valid transactions as for the native currency. (The native currency defined by the protocol is usually not used in multichain applications, but it could be.) The input and output quantities of defined assets are encoded within each transaction and the balances are verified by every node. Assets services include issuance, reissuance, transfer of assets, support for atomic exchanges, escrow and destruction of assets.

5.2.1.2 Streams

Some applications may require the robustness and immutability of a blockchain, but the schema is more suitable for with time series data, a NoSQL key-value or document database. For use cases that are oriented towards distributed databases for time-stamped data archiving and retrieval, Multichain has introduced an abstraction to deal with time-stamped data without the need to directly involve an asset.

A stream is a set of indexed and time-stamped data items, signed by the publisher. The payload may be text, JSON (JavaScript Object Notation) or binary documents up to several megabytes. Indexing is by keywords, publisher, and timestamp. Multichain provides extensions to the protocol for creating streams, writing to streams, subscribing to streams as well as indexing and retrieving. Retrieval supports merging of consecutive JSON documents to provide the latest state of every attribute.

5.2.1.3 Off-chain storage

The problem of storing large data items on the blockchain mentioned in section 4.1 has been addressed by Multichain 2.0. By declaring stream data as off-chain, the developer can let Multichain handle hashing, data verification, discovery, decentralized storage and delivery and delayed availability.

On-chain and off-chain items can be used within the same stream, and the various stream querying and summarization functions relate to both types of data identically. The information design of streams and client applications will have to take into account that not all data will be available immediately and that data should be grouped and retrieved in such a way as to avoid replication on all nodes.

Currently the storage is file based, pluggable backed databases are in development.

5.2.1.4 Smart Filters

In a general sense, Multichain takes the approach in which data is embedded immutably in a blockchain, but the code for interpreting that data is in the node or application layer. This is deliberately different from the "smart contracts" paradigm, as exemplified by Ethereum, in which code is embedded in the blockchain and runs in a virtual machine.

However, Multichain 2.0 provides smart filters, which allow custom rules for the validity of transactions or streams to be defined. This provides great flexibility for streams and custom assets. Smart filters are pieces of code, written in JavaScript[18], that are embedded in the blockchain. When determining transactions validity, filters have access to information about the assets and streams on the chain, data about the preceding blocks

[17] https://www.multichain.com/
[18] Running in a deterministic version of Google V8 JavaScript engine.

and the permissions of nodes. Like the regular built-in transaction validation, smart filters are run independently by all nodes for all transactions.

Smart filters are for transaction validation, not general-purpose programs acting on and keeping their own internal state as Ethereum smart contracts. Nor do the smart filters call other smart filters or external data sources.

Two types of Smart Filters are supported, transaction filters and stream filters. The former inspects the input, output and metadata of a transaction and determine if the transaction is valid. If a transaction is invalidated by the filter, it will be rejected by all nodes, as will any block containing an invalid transaction. The latter inspects the on- and off-chain data of a stream item, the publisher data and keys and determine if the stream item is valid. Invalid stream items cannot be published or retrieved.

5.3 Blockchain API

Rather than integrating all composition components that needed access to blockchain functionality with Multichain, it was decided that an adapter should be provided. This would provide the functionality needed and integrate with the identities provided by the Security Framework to provide access to the streams and assets in the blockchain. Each stakeholder in a marketplace will use one multichain node and wallet accessed by one instance of the Blockchain API. This will isolate COMPOSITION components from changes between Multichain versions and provide a way to plug in other blockchain implementations.

5.3.1 Functional View

The Multichain runtime publishes two interfaces: the peer-to-peer communication port for synchronizing the blockchain and a JSON-RPC API for issuing commands. The JSON-RPC API implements the functions of the Bitcoin protocol and the Multichain extensions.

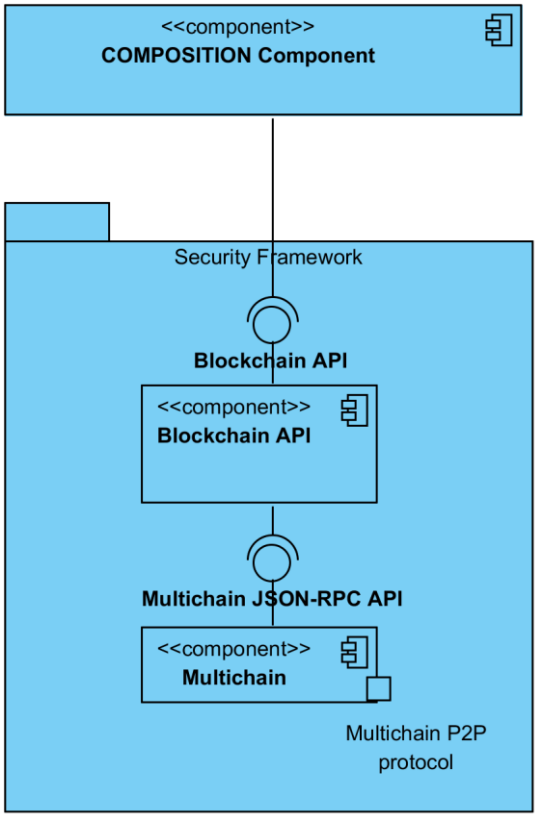

Figure 6: Blockchain API

To provide a chain-agnostic and easy-to use interface to the blockchain, a RESTful Blockchain API proxies the JSON-RPC-API. This RESTful interface is used by the stakeholder agents in the inter-factory deployment

of that stakeholder. An OpenAPI Specification (OAS)[19] of the interface provides defines a standard, programming language-agnostic interface description that can be used for client code generation.

5.3.2 Deployment View

A Docker[20] container hosts the RESTful Web Service in Python 3 that implements the Blockchain API. Only the Blockchain API has access to the Multichain JSON-RPC port of the multichain node. The multichain parameters are set by environment variables for the COMPOSITION Docker images for the blockchain nodes. Although multichain is peer-to-peer, two types of Docker images have been configured: one for the main node that will define the blockchain properties and mining rules, set up by the marketplace administrator. The other is aimed at the other marketplace stakeholders that will connect to and interact with the main marketplace chain. The Docker images are uploaded to the COMPOSITION Docker organization.

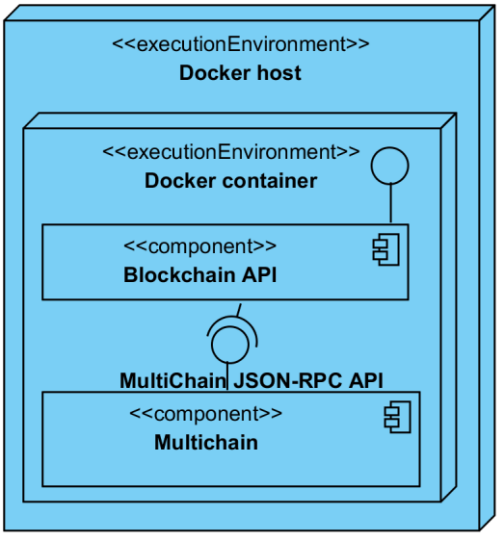

Figure 7: Blockchain API Deployment

The container requires at least 512 MB memory and sufficient storage space for the stream data, recommended by Multichain documentation is 1 GB.

A blockchain node may be deployed by any marketplace stakeholder, with the Blockchain API accessible only to the stakeholder's agents. The multichain nodes synchronize the state of the blockchain in a peer-to-peer network, but only the Blockchain API has access to the Multichain JSON-RPC API.

[19] https://www.openapis.org/
[20] https://www.docker.com/

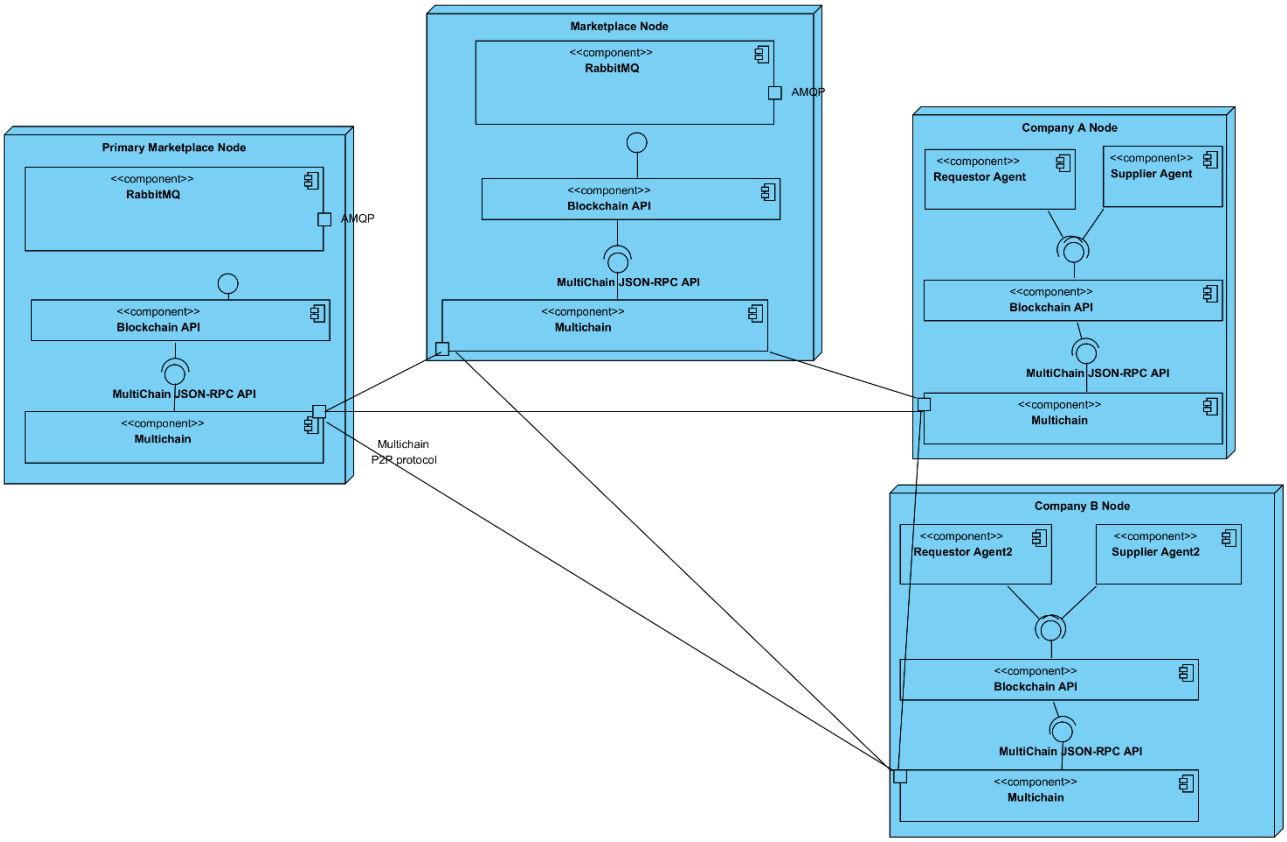

Figure 8: Blockchain Marketplace deployment

5.3.3 Information View

The Multichain marketplace blockchain is the store of agent public keys, agent CXL (COMPOSITION eXchange Language) messages and reputation data. In a COMPOSITION Marketplace, a set of autonomous agents representing stakeholders interact using a common vocabulary through the same shared Broker. The marketplace is open to stakeholders with valid COMPOSITION credentials (any such stakeholder or a closed group depending on the type of marketplace). The marketplace may be operated by one stakeholder or a group of them. The choice of using public, private or consortium consensus depends on the administrator(s) of the marketplace, who will configure the blockchain.

Each marketplace will use one blockchain. The blockchain is used to store the agent public keys and log the transactions and reputation data of agents. Permissions are set for the stakeholders in the blockchain, who may host their own blockchain nodes. There will be one stream in the blockchain for publishing public keys. Each agent will have one stream that provides an immutable ledger of CXL (the protocol that agents use to communicate offers and agreements) messages. This will be writable by the agent stakeholder only but readable by all other stakeholders. Agents will also use another stream to store reputation data.

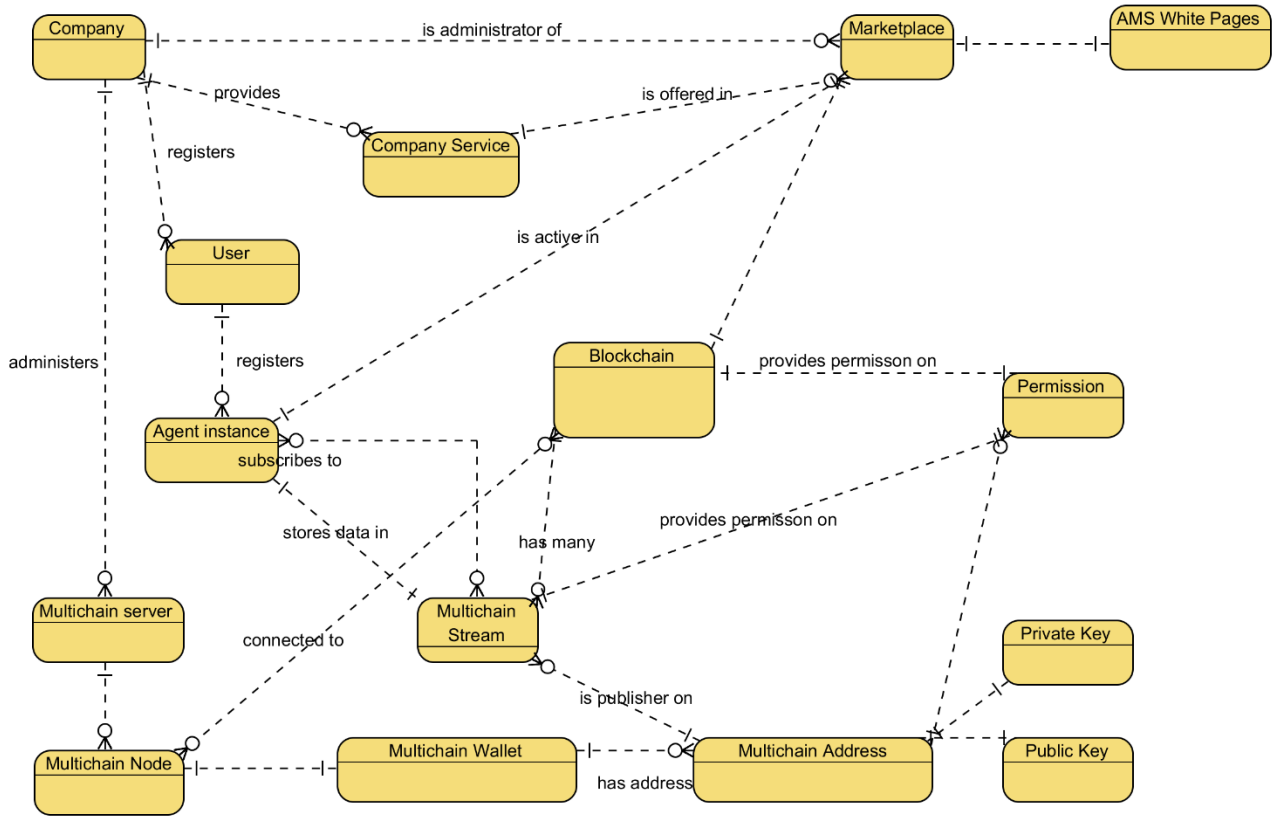

Figure 9: Blockchain in the COMPOSITION Marketplace

The streams can be queried by a set of arbitrary index keys, set by the clients. One keyword is set by the API to create a unique id for each message. The messages are stored in JSON format.


```
{
    "id": "a7fe45ee33+13b4e7f823",
   "indexkeys": [
    "sender:agent_KLEEMANN",
     "receiver:agent_ELDIA",
     "conversation-id:a7fe45ee33",
     "message-id:13b4e7f823"
  ],
  "payloadType": "JWT",
  "payloadJSON": {},
   "payloadJWT":
   "eyJhbGciOiJSUzUxMiIsInR5cCI6IkpXVCJ9.eyJvbnRvbG9neSI6WyJGSVBBLXByb3B
   vc2FsLXByb3RvY29sIl0sInByb3RvY29sIjoiRklQS1Db250cmFjdCIsZW5k
   ZXIiOnsiYWRkcmVzc2VzIjpbeyJleGNoYW5nZSI6ImFnZW50X0tMRUVNQU5OIn0seyJxd
   WV1ZSI6ImFnZW50X0tMRUVNQU5OIn1dLCJuYW1lIjoiYWdlbnRfS0xFRU1BTk4iLCJ1c2
   VyLWRlZmluZWQiOnsiYWdlbnQtb3duZXIiOiJLTEVFTUFOTiIsImFnZW50LWlkIjoiYWd
   lbnRfS0xFRU1BTk4ifX0sImxhbmd1YWdlIjoiWC1DT01QT1NJVElPTi1GSVBBLUpTT04i
   LCJyZXBseSlieSI6IjIwMTktMDEtMTRUMTY6NDA6MjErMDA6MDAiLCJjb250ZW50Ijp7I
   m9mZmVyLWRldGFpbHMiOnsiZ29vZCI6InNjcmFwLWlldGFsIiwicHJpY2UiOjQlLCJjdX
   JyZW5jeSI6IkVVUiIsInF1YW50aXR5LXVvbSI6ImtnIiwicXVhbnRpdHkiOjEyMCwicGl
   ja3VwLWRldGFpbHMiOnsiZW5kLWRhdGUiOiIyMDE5LTAxLTE5VDAwOjQwOjEwKzAxOjAw
   Iiwic3RhcnQtZGF0ZSI6IjIwMTktMDEtMTRUMjA6NDA6MTArMDE6MDAifX19LCJhY3QiO
   iJjZnAiLCJtZXNzYWdlLWlkIjoiZDdiZGNmMzktOGRjZC00NGE2LWI4NzUtNjllYWJjNj
   NiYjY2IiwiY29udmVyc2F0aW9uLWlkIjoiMDU4YTdhMzgtNDlmOC00YTQ5LWFhMWMtMjN
   iNjljNzkyOGZiIn0.TwDjG1rYR7yExRfThclOdK719Mj973VTLs92rqWS0BU-CS8Mqyna
   vgK_TimsRlkCyWvhspO6ZK8oNsaUg7YO4jUbWhWCtLPhtklSGxPBiF-XJTo3oAZGFfmuf
   nxkxIRM5clxHJTezGbri8Kshr9-Po58shJXPhZBA-CniTvJ-Ir0DbANT8Mgbh2qRzbadA
   cdi7aoWDFyO94wKtt4c_CHSEgSBgPSNeExqnawLjoHKgpnK-CCEGONcuxm3IYBOudWRjv
   IJu_go1NqxUgVRDlNUfdWFIAxgge8RLECU5eji6_l_2UwfkwVHFGvCmZZ5GMXV239Jae9
   3USIMWzoyxZ3Ow"
}
```


Figure 10: Example POST to Blockchain API

6 Applications in COMPOSITION

A number of applications of blockchain have been proposed and discussed in COMPOSITION. Due to limited development time, applications were prioritized by how well they fulfilled Technical Objective 2.1 (COMPOSITION) and pilot user concerns. The following section provides an overview of implemented applications, those that have been designed but not implemented and use-cases still in development.

6.1 Public Key Infrastructure (PKI)

Since it is proposed that all messages flowing in the COMPOSITION platform through the COMPOSITION Message Broker (RabbitMQ[21]) must be signed using JWS[22] (JSON Web Signature) standard proposed by IETF[23], there is the need to make available to the subscribers of messages the public keys so it is possible for them to verify the digital signature. Instead of using the common approach of publishing the public keys through a web site or a web service, in COMPOSITION we plan to use blockchain technology to make these public keys available.

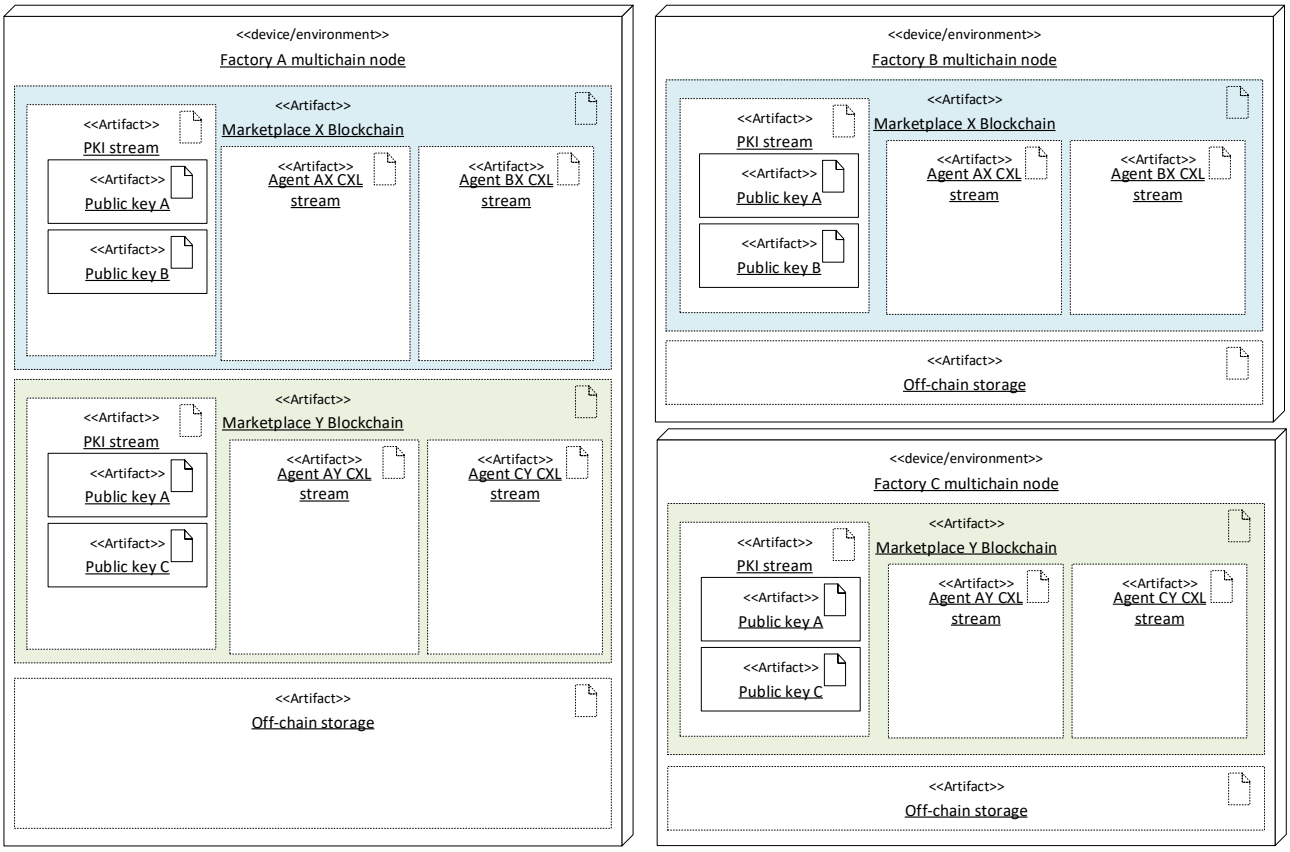

Figure 11: Multichain stream PKI (factory A is participating in multiple marketplaces)

The idea in the beginning is simple; the message publishers put in their blockchain node their public key while maintaining the private key locally and secret. The public keys published will be replicated on all blockchain nodes connected and keeping a copy of them making it accessible to all subscribers that have the rights to read them.

Each marketplace uses a separate blockchain. The marketplace blockchain has a multichain stream dedicated to public keys, readable by all marketplace stakeholders. The blockchain API provides convenient methods for publishing and retrieving public keys in a JSON structure (using JWK (JSON Web Key[24]) format) to and from the stream.

[21] https://www.rabbitmq.com/
[22] https://tools.ietf.org/html/rfc7515
[23] https://www.ietf.org/
[24] https://tools.ietf.org/html/rfc7517

6.2 Message Logging

The blockchain API provides methods for general logging of messages. The idea is that the publisher of a message should calculate the hash of the message using a hash cryptographic function (to be decided) and store the result hash value in the blockchain along with some metadata. Upon receiving a message, a subscriber can calculate the hash of the received data and can look for it in the blockchain, ensuring this way the integrity of the data received. This together with the digital signature of the message is going to give the subscriber security to trust on the message received.

The following diagram (Figure 12) gives a high-level overview on the signing and logging procedures and data flow between the components involved.

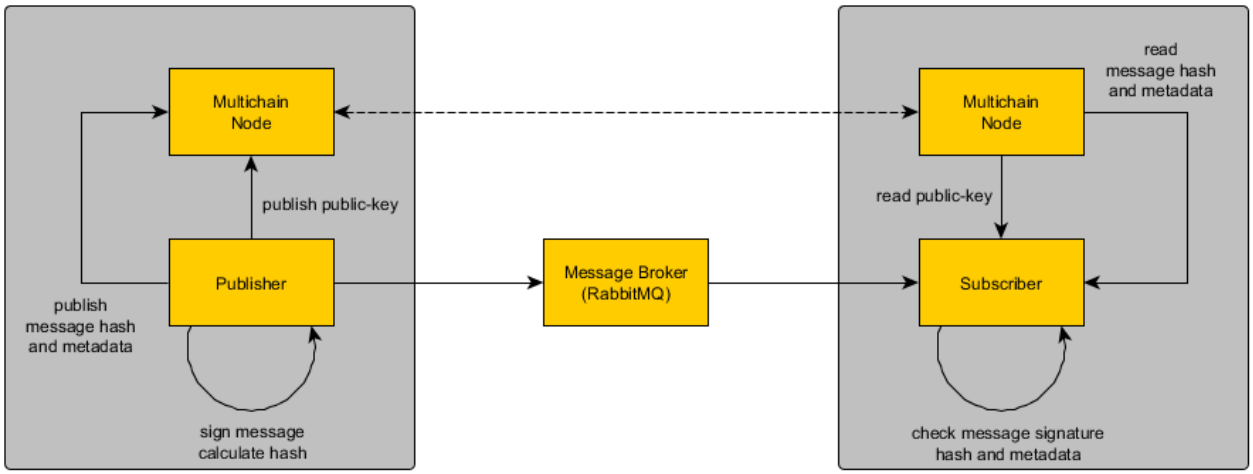

Figure 12: Overview of signing and logging of messages

A more detailed view on the whole process of publishing and subscribing in COMPOSITION taking into account the use of the methods of signing messages and keep log of them, proposed in COMPOSITION Security Framework, as well as the steps to validate the signature and the content of the message can be seen in the flowchart diagram below (Figure 13).

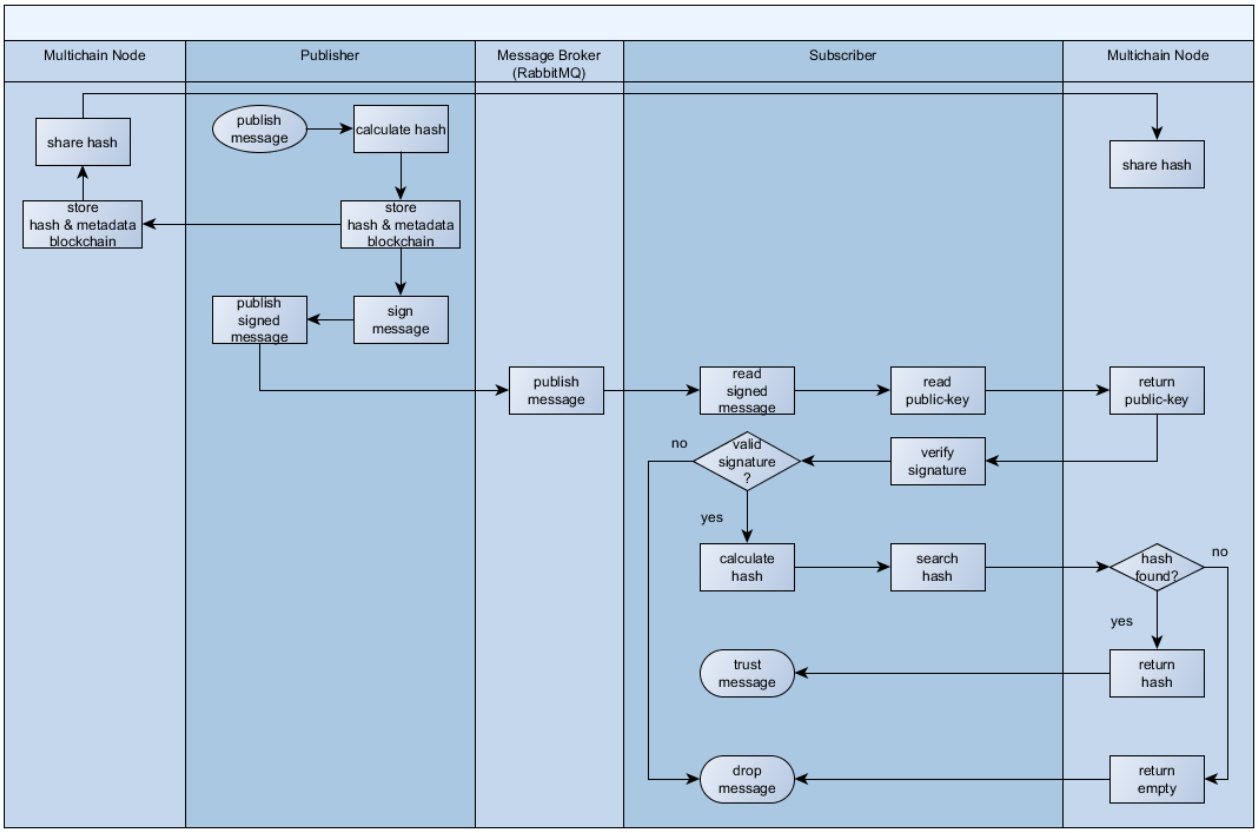

Figure 13: Flowchart diagram publish-subscribe procedure in COMPOSITION

The Blockchain API provides convenient methods to store and retrieve messages.

6.3 Blockchain, Trust and Reputation

COMPOSITION is going to define a reputation model adding another level on trust. Blockchain is also the technology to be used to store the reputation of the stakeholders and the way to share it with other stakeholders. It will also keep track of the reputation over time due to the immutability nature of the blockchain technology.

The basic concepts and models of the COMPOSITION reputation model are detailed in D4.4 Prototype of the Security Framework I whilst the technical deployment and implementation is detailed in D4.5 Prototype of the Security Framework II.

COMPOSITION is relying on blockchain technologies as the central component of its log-oriented architecture. This technology will be used for implementing a secure, trusted and automated information exchange related to supply chain data. Considering the distributed nature of blockchain (Nakamoto, 2008) and, more in general, of the COMPOSITION infrastructure, it makes sense to rely on a distributed Reputation Model: each agent will compute his own reputation values, and will be in charge to provide these values to the other entities.

In the literature, there are some academic papers related to the usage of blockchain in trust management and authentication (Alexopoulos et al, 2017), (Moinet et al, 2017). However, considering the dynamic and distributed nature of blockchain, some interesting scenarios could be explored.

In (Hoffman et al, 2009), the three fundamental dimension of a generic Reputation Model have been identified: formulation, calculation and dissemination. So far, the last dimension, which includes also how reputation values are stored, has not been taken into account.

Blockchain could be really helpful in this case: the idea is to exploit this technology for storing local reputation values (R2, R3): in this way each agent would be aware when a reputation value given by a specific agent A to another agent B has been updated (R5). Then, he can choose if consider or not this new value, basing on his local reputation value related to agent A, when he should interact with agent B. The usage of blockchain will also help recognizing possible cheating behaviours, for instance a “bad” agent who tries to submit

misleading reputation values on behalf of other agents or re-join the marketplace for resetting his low reputation (R9). With the adoption of the blockchain, all the agents will have a global view of every interaction related to each agent of the marketplace.

6.4 Location tracking

Another application that has been considered is material and asset tracking inside the factory. A Realtime Location Tracking System (see UC-BSL-3, (COMPOSITION D2.1)) is complemented with a private blockchain were all material and component movements/actions are logged to create an immutable log of positions for valuable materials. This blockchain use case is under design but currently not in implementation.

While private, the blockchain could still employ several nodes for consensus, all trusted and owned by the company but located in different IT environments and administrative domains. Using a blockchain would also allow the location tracking to tie in to logistics (see section 6.6) and supply chain audit trails. Location data would have to be signed by a trusted component, or the sensors could possibly be configured to send raw signed transactions directly to the multichain node.

6.5 IPR protection

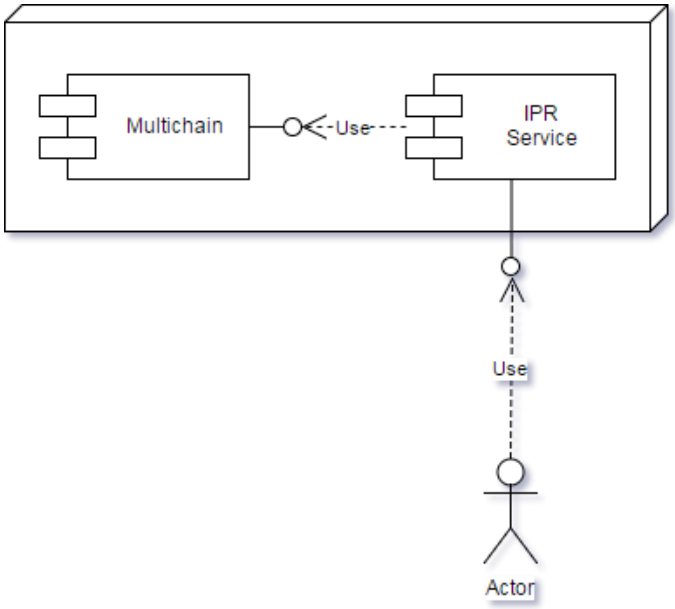

Figure 14: IPR Service

One blockchain use-case that has not been developed due to low priority by end-users, is the protection of IPR. An actor may use an IPR service to upload any kind of digital document and get a digital certificate of authentication for the document. A hash is stored in the block chain that will verify that the actor in question provided this document at the specified time. The document does not need to be stored in the blockchain, but the Multichain off-chain storage may be used to also store the document.

The method to obtain a certificate for a document is simple:

1. Upload document
2. The IPR service calculates the hash
3. The IPR service checks if the hash has been stored in the blockchain
4. The hash is not found
5. The IPR service stores the hash in the blockchain
6. Return hash

Figure 15 depicts the process:

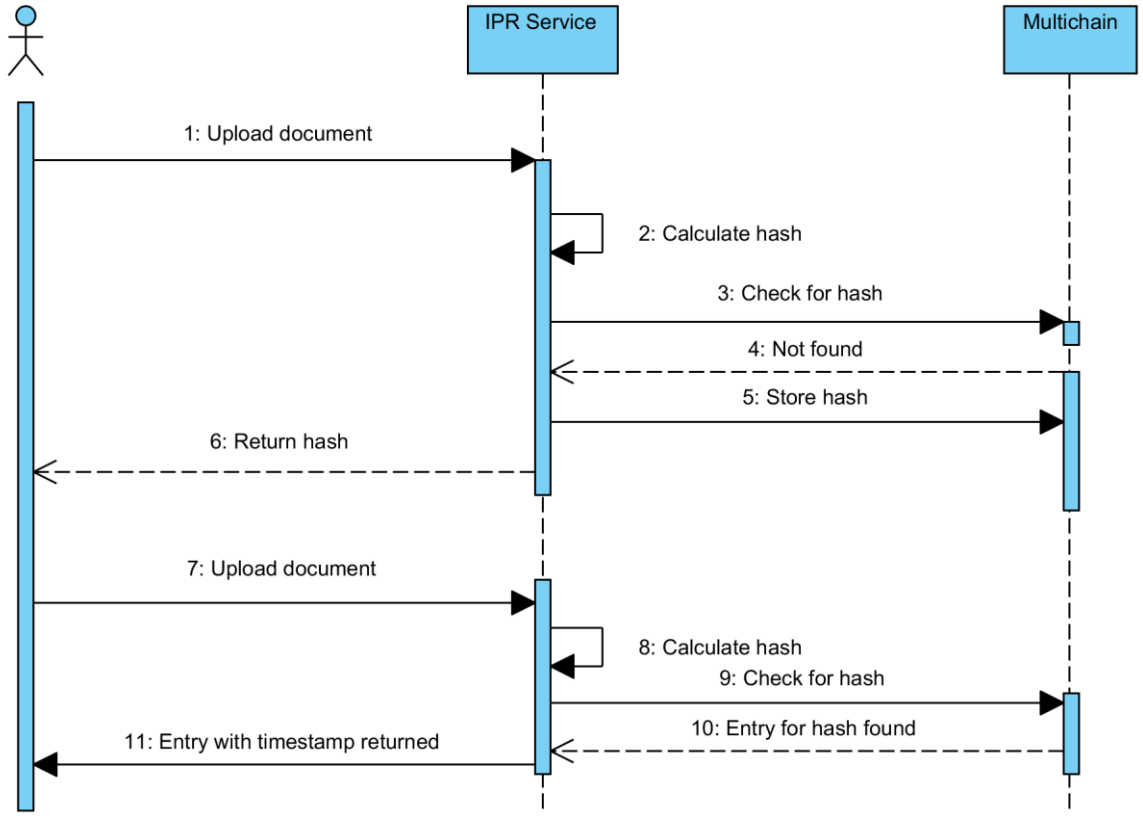

Figure 15: IPR Service sequence diagram

The method to check if a document existed at any given time is simple also. The steps are the following:

7. Upload document
8. The IPR service calculate the hash
9. The IPR service checks if the hash has been stored in the blockchain
10. The hash is forum in the blockchain
11. Return the blockchain entry (with metadata like timestamp)

6.6 Supply chain logistics demo

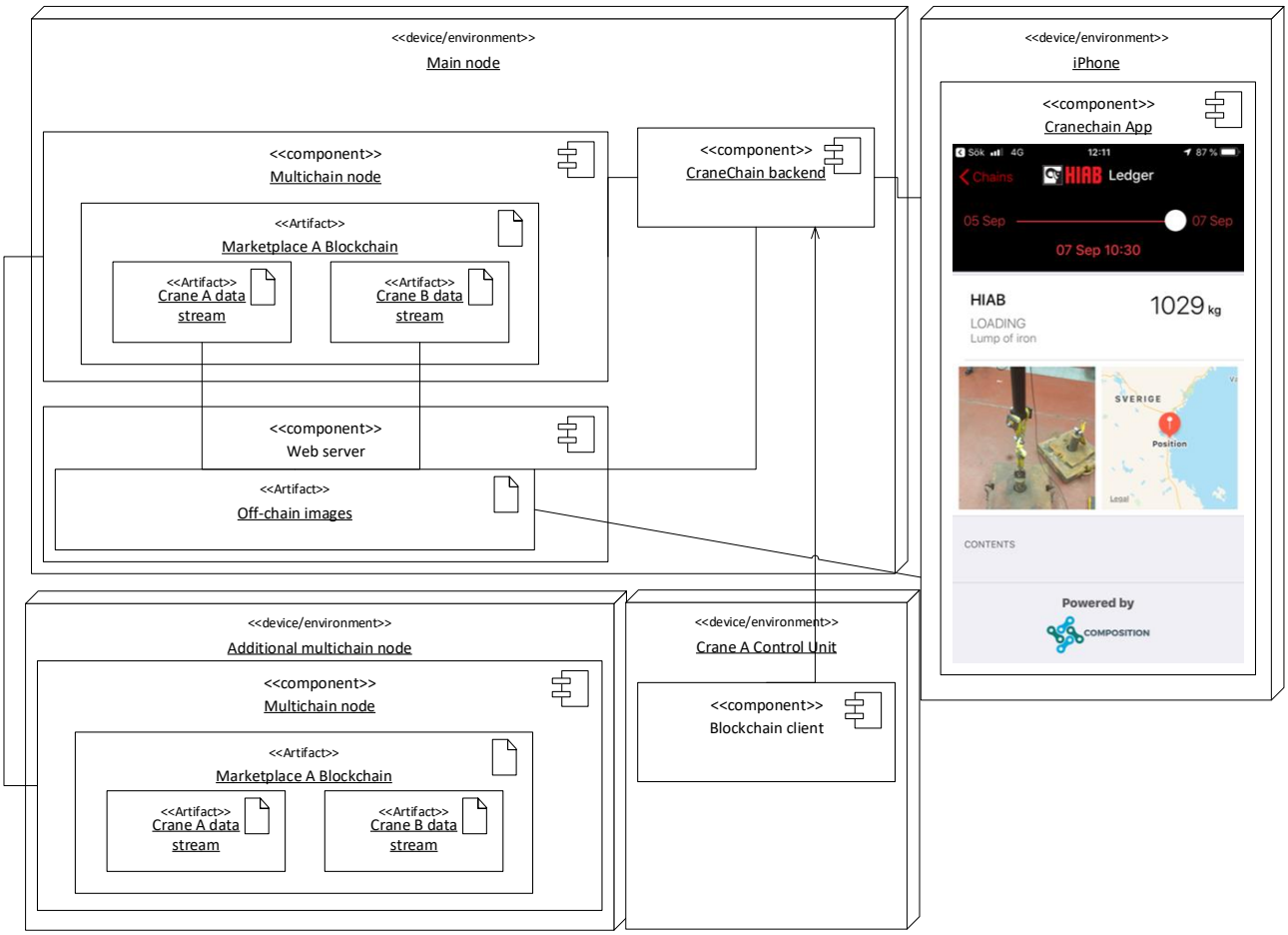

Figure 16: Blockchain supply chain logistics demo

During the COMPOSITION project, the preliminary results in task 4.3 were applied and disseminated in a development challenge initiated by the on-road load handling equipment manufacturer HIAB[25]. The case put forward by COMPOSITION was to track shipments on a blockchain by using a multichain stream to log container weight and other parameters measured by the crane, positioning data, and images from the crane camera. This will provide an audit trail of everything that had happened with the shipment. Images were stored off-chain on a web server. This was implemented using Multichain 1.0, today the built-in off-chain storage would have been used for the images.

6.7 Confidential streams

User requirements indicate that some agent interchanges need to be confidential, e.g. it is not always desirable to let all marketplace participants know the agreed price of a contract or the capabilities of a supplier. This has not yet been implemented COMPOSITION at the time of writing, but Multichain guidelines provide a generic solution to the problem of keeping data confidential on the blockchain[26], by using a combination of streams.

[25] https://www.hiab.com/en-US/HIAB/highlights/cargohack/
[26] https://www.multichain.com/blog/2016/09/introducing-multichain-streams/

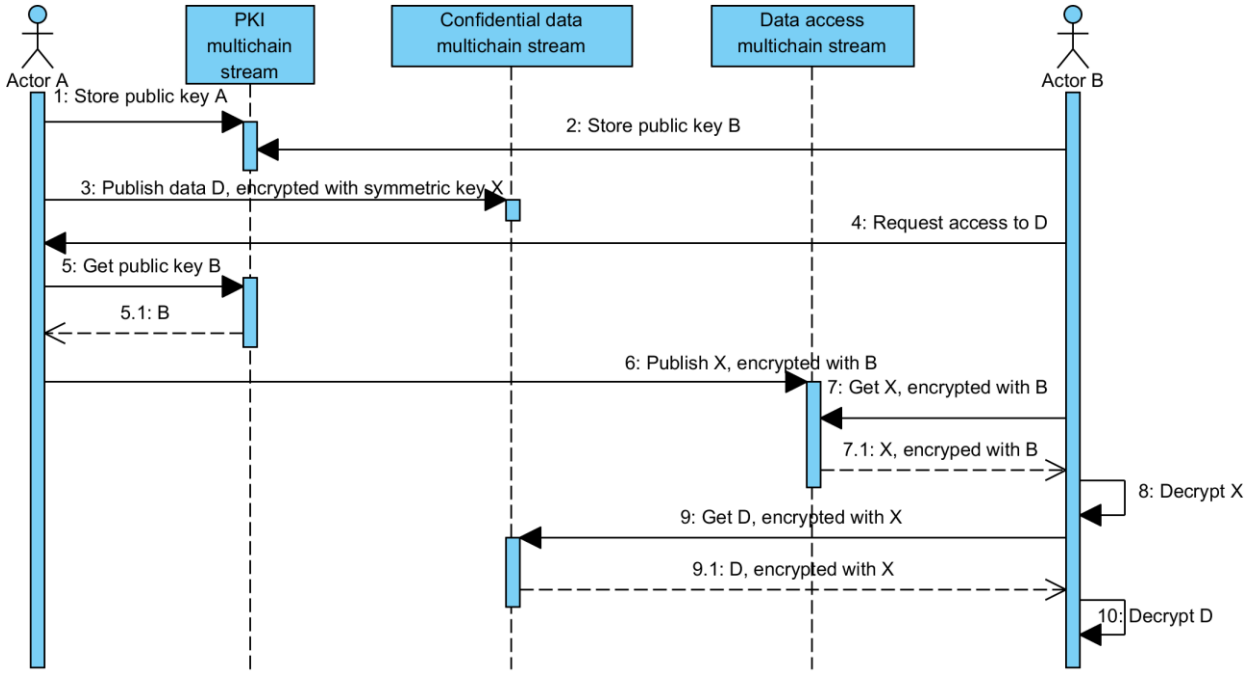

Figure 17: Confidential data in multichain streams

One stream is used by stakeholders to distribute their public keys (as described in section 6.1).

A second stream is used to publish data. Each data item (stream entry) that should be confidential is encrypted using symmetric cryptography with a unique key created for that item.

A third stream provides the data access. To let a stakeholder read a data item, the unique secret key for that item is encrypted using the public key of that stakeholder. This is then published on the stream. The stakeholder may then retrieve the symmetric key and the data and decrypt the message.

7 Conclusions

The work in task 4.3, realizing technical objective 2.1, has been largely focused on finding and applying a suitable blockchain implementation for supply- and value chain applications. It contributes to the core set of security mechanisms that makes up the COMPOSITION Security Framework of WP 4, Secure Data Management and Exchange in Manufacturing. The use cases for blockchains in industry 4.0 are different from the original cryptocurrency applications of blockchains; more oriented towards audit trails and secure exchange of data. The blockchain implementation selected, Multichain, has proven to be relatively easy install and configure. It has also provided a number of useful abstractions and features in subsequent versions, e.g. indexable data streams, off-chain storage and programmable transaction rules (smart filters). Many of these solve problems that would otherwise require an ad-hoc solution.

One feature that Multichain does not provide is general purpose smart contracts. However, the Blockchain API, the proxy used by COMPOSITION components to access the blockchain(s), could use another implementation that provides this functionality if needed. COMPOSITION Components access the blockchain via a REST API without dependencies on the specifics of Multichain.

Several use cases for blockchains has been proposed and evaluated during the project, both intra- and inter-factory applications. Some of these have been implemented in the pilot deployments and also validated in other relevant environments, e.g. the HIAB crane logistics challenge. Overall, we found that the blockchain provided solutions to many of the concerns expressed in the project specification and the user requirements. The Blockchain API and the specific implementations and configurations of Multichain used in COMPOSITION applications provide a supply-chain specific layer on top of blockchains and useful body of knowledge on how to apply blockchains to industrial use cases. We will continue realizing the blockchain use cases and developing the Blockchain API during the remainder of the project.

The COMPOSITION Blockchain implementation and deployment will be evaluated by ATOS from a security perspective.

8 Annex 1: Background - Bitcoin

The first application ever to use the blockchain technology was the electronic cash system Bitcoin (Nakamoto, 2008). Bitcoin is a fully decentralised application where you are able to allow online transactions from one party to another without the use of trusted third parties (such as financial institutions, banks or accountants). This is done in a peer-to-peer fashion where every node connected to the Bitcoin network holds the same version of a distributed ledger. All transactions ever made on the network is kept in this ledger, which is public and for everyone to inspect, making the system completely transparent. This ledger is immutable so once a transaction is registered no one can either change it or delete it, and it is nearly impossible to hack. With these characteristics of the ledger one can find out how much funds are linked to one user of the network at any point in history, and at the same time be sure that this information is correct.

The Bitcoin ledger is essentially the blockchain and it is one of the core features of Bitcoin. To understand what the blockchain is, and what it can be used for, it makes sense to first go through the basics of Bitcoin. This is because Bitcoin was the first to adopt the technology and it is easier to get a grasp of it when considering the blockchain as solely a financial solution.

The Bitcoin network is a world of transactions. Alice sends money, in this case Bitcoins or BTC, to Bob. Both Alice and Bob are clients in the Bitcoin network. The transaction is then put in a pool with other unconfirmed transactions all made at the same time. (To make sure Alice has indeed enough BTC to send to Bob and that she is the rightful owner of the funds is resolved with a technique called “digital signing” and “reversing a transaction chain”, both of which we will not get into to more details about).

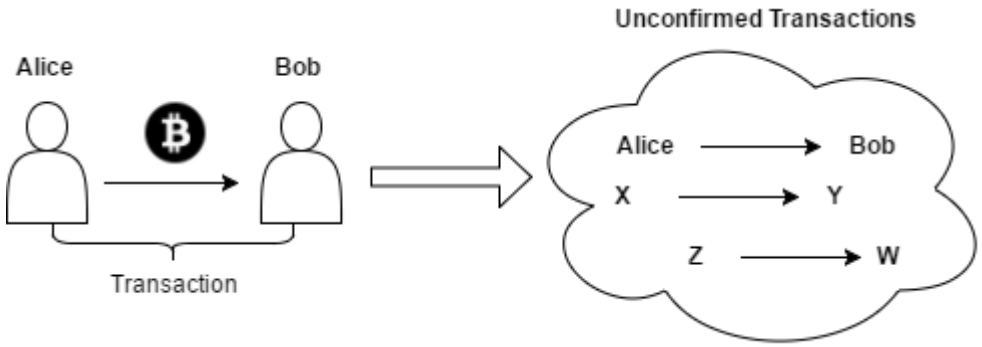

Figure 18: Bitcoin transaction

Any node in the network can take a set of unconfirmed transaction and put them in a block. This block will later on be put on the chain of blocks containing confirmed transactions and then broadcasted to everyone in the network. The ledger which all user have is updated with the latest confirmed transactions including the transaction of BTC from Alice to Bob. Every block in the chain has a reference to the previous block in the chain, making it chronological.

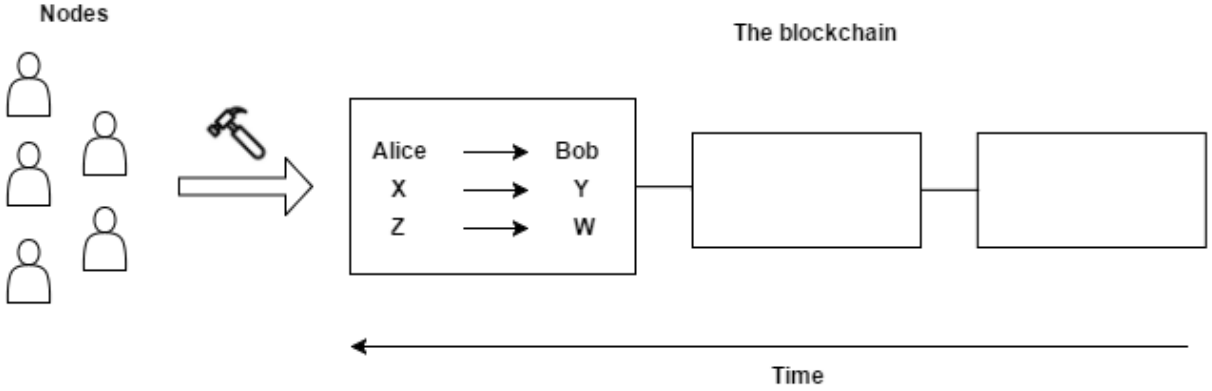

Figure 19: Collecting transactions

But there is a problem if anyone, without any effort, could confirm transactions and then add them in a block and put that on the blockchain. For example, let's say Alice sends an arbitrary amount of BTC to Bob and in exchange Bob ships a product of any kind to Alice. At the same time Alice sends the same BTC to her second account and confirms that transaction before the transaction to Bob. Now no one in the network will confirm the transaction to Bob because the BTC which was meant to him has already been spent. No BTC will be sent to Bob, yet Bobs product has already been shipped to Alice. This is called the double spending problem and it introduces the approach to achieve consensus between nodes in the network.

In the financial world today, we trust a bank to keep track of our belongings. A bank keeps a centralised ledger with all their customers account balances and updates them accordingly. In bitcoin everything is decentralised and everybody in the network has the same version of the ledger. So who's ledger of all the nodes in the network are we supposed to trust?

To achieve consensus in Bitcoin a Proof-of-work system is implemented. As Satoshi Nakamoto, the creator of Bitcoin, explained it in the Bitcoin whitepaper "The proof-of-work involves scanning for a value that when hashed, such as with the algorSHA-256, the hash begins with a number of zero bits" (Nakamoto, 2008). One can think of system with a number of people doing coin flips. The first person of the group who manage to flip a large number of heads in a row wins. The node in the Bitcoin network who manage to find the value which hash begins with the correct number of zeros is the one who can create the new block in the blockchain. The reward for this achievement is a number of BTC and the process is popularly called mining. This is also the main incentive for people to maintain the system in a decentralised fashion.

To find the specific value which hash begins with a number of zeros requires computing power. Theoretically Alice could do the double spending attack against Bob as explained above with the only catch that she would need 51% of the computer power in relationship to the whole Bitcoin network. This would make Alice statistically more likely to create the new block and make the payment to Bob invalid, but to have that much computer power is considered impossible. This is why the Bitcoin blockchain is immutable. To alter or delete any transaction in the blockchain you'll have to recalculate every block, that is find the hash value which begins with a given number of zeros, from the block where the transaction you want to alter or delete is located to the current block in the blockchain.

To summarize, Bitcoin is secure peer-to-peer digital currency which uses the blockchain technology to order transactions and making double spending attacks almost impossible to execute. Every transaction ever made is recorded on the blockchain and every block is linked to the previous one making it a chain from the very first created block, the genesis block. Nodes in the network are rewarded by their work of maintaining the blockchain which makes the system fully decentralised and without the need for a trusted third party.

The monetary aspect of a blockchain is just one use-case of the technology. At its core, money exists to facilitate trade of any kind. Money as we know it is just a token which we can trade for other things. The Bitcoin is the same thing, but the ledger which keeps the Bitcoin alive does not care what the Bitcoin, the token, represents. It can represent any given amount of fiat currency, a share in a company or anything else which is considered to have value. The blockchain technology is therefore useful for many applications where you want to eliminate trusted third parties and/or to facilitate trade of any kind.

Bitcoin is not the only decentralised application running on top of a blockchain technology. Throughout the last years the many applications of the blockchain has been put forward and many companies and organisations has been introducing their approaches to the technology. Some of which has the decentralisation as main focus and a cryptocurrency as base, when others solely aim to provide a simplified way for business to trade value in a secure manner.

9 List of Figures

9.1 Figures

Figure 1: Simplified blockchain 7
Figure 2: Block design 9
Figure 3: Public blockchain 11
Figure 4: Private blockchain 12
Figure 5: Consortium blockchain 13
Figure 6: Blockchain API 16
Figure 7: Blockchain API Deployment 17
Figure 8: Blockchain Marketplace deployment 18
Figure 9: Blockchain in the COMPOSITION Marketplace 19
Figure 10: Example POST to Blockchain API 19
Figure 11: Multichain stream PKI (factory A is participating in multiple marketplaces) 20
Figure 12: Overview of signing and logging of messages 21
Figure 13: Flowchart diagram publish-subscribe procedure in COMPOSITION 22
Figure 14: IPR Service 23
Figure 15: IPR Service sequence diagram 24
Figure 16: Blockchain supply chain logistics demo 25
Figure 17: Confidential data in multichain streams 26
Figure 18: Bitcoin transaction 28
Figure 19: Collecting transactions 28

10 References

(COMPOSITION)	The COMPOSITION Description of Action
(COMPOSITION D2.1)	D2.1 Industrial Use Cases for an Integrated Information Management System
(COMPOSITION D2.2)	D2.2 Initial requirements specification
(COMPOSITION D2.4)	D2.4 The COMPOSITION Architecture Specification II (COMPOSITION D4.2)
(COMPOSITION D4.2)	D4.2 Design of Security Framework II
(Boucher, 2017)	Boucher, P., "How blockchain technology could change our lives – In-depth analysis", European Parliamentary Research Service, ISBN 978-92-846-0549-1, 2017.
(Lamport, 1982)	Lamport, L., R. Shostak, and M. Pease, “The Byzantine Generals Problem”, ACM TOPLAS (Transactions on programming languages and systems), 1982.
(Merkle, 1980)	R.C. Merkle, "Protocols for public key cryptosystems", In Proc. 1980 Symposium on Security andPrivacy, IEEE Computer Society, pages 122-133, April 1980.
(Greenspan, 2015)	G. Greenspan, “MultiChain Private Blockchain — White Paper”, 2015.
(Buterin, 2014)	V.Buterin, “Ethereum White Paper - A Next-Generation Smart Contract and Decentralized Application Platform”, 2014.
(Gray, 2017)	M. Gray, “Introducing project Bletchley - Microsoft's Blockchain Architecture Overview”, 2017.
(Nakamoto, 2008)	S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system,” 2008.
(Alexopoulos et al, 2017)	N. Alexopoulos, J. Daubert, M. Mühlhäuser and S. M. Habib, “Beyond the Hype: On Using Blockchains in Trust Management for Authentication,” in Trustcom/BigDataSE/ICESS, 2017.
(Moinet et al, 2017)	A. Moinet, B. Darties and J.-L. Baril, “Moinet, Axel, Benoît Darties, and Jean-Luc Baril,” in arXiv preprint arXiv:1706.01730, 2017.
(Hoffman et al, 2009)	K. Hoffman, D. Zage and C. Nita-Rotaru, “A survey of attack and defense techniques for reputation systems,” ACM Computing Surveys (CSUR), vol. 42, no. 1, 2009.