Semantic Smart Contracts

And Their Integration With Semantic Data Licesing

Zahra Jafari

University of Oxford

A thesis submitted for the degree of

Master of Informatic

2019

This thesis is dedicated to
someone
for some special reason

Acknowledgements

plenty of waffle, plenty of waffle, plenty of waffle, plenty of waffle, plenty of waffle, plenty of waffle, plenty of waffle, plenty of waffle.

Abstract

plenty of waffle, plenty of waffle, plenty of waffle, plenty of waffle, plenty of waffle, plenty of waffle, plenty of waffle, plenty of waffle.

Contents

List of Figures

Chapter 1

Smart Contract and Distributed Ledger Technology

1.1 Introduction

With the arrival of Block chain technology as a form of Distributed Ledger Technology, the rule of bitcoin in this field is on increase. Block chain enables developer tp create diverse distributed application. One of the nove approach in this field is Ethereum platform which is the turing-compelete system used in smart contract.
Smart contract is the part of code in block chain without ....TODO. Block chain is the chain of block (participant computer) over network and provides a truth worthy mechanism to facilitate distributed contract in secure way.
TODO
In this paper, we attempt to investigate if Link data in the combination with block chain to realize the distributed semantic web application. TODO

1.2 Ledger

1.2.1 Destributed Vs. Undestributed

1.3 Distributed Ledger Technology

Is the method of keeping distributed ledger on networks of computer. DLT use consensus technique for adding new node(participant) over network. Similarly, This is the digital record that shared across participant and the records hold by each node or participant. The term 'block chain' is a most well-known type of DLT and are used by many best-known instances of DLT.
It goes without saying that distributed ledger can be either permission less or with

permission, regarding to be private or public. Most of the distrubuted ledger is permission less and public means any one can easily join to network and see all entries. But in financial fields, distributed ledgers are restricted to membership of each participant and they are 'commissioned ' and 'private' network. Furthermore, in financial fields exist also commercial sensitivity about the privacy of data related to each participant. In the other word, no one wants that data to be seen by the other participant. therefore, participants are able to see their own data and transaction on the network [44].
recently, blockchain as a type of distributed ledger become most popular and widely used in diverse fields. ledgers such as Ethereum extends initial bitcoin blockchain with features such as smart contracts that enable to distribute data on blockchain in secure way. In addition, there is incrementally need to integrate data stored in distributed ledger with other external sources. And also integrate smart contract with the service available on the web. This is where linke-data comes to play to provide sufficient access to data and smart contract also stored on Ehereum blockchain via semantic web technology stack.[p1431]

1.4 smart contract

Smart contract in computer science is the piece of code which is designed to execute certain terms if the predefined conditions are met. These terms are embedded and preformed on distributed ledger. As a compared to traditional contract that includes third party to execute terms of condition , smart contract has low transaction fees and is more profitable. Conception of smart contract is defined in distinct schools or terms: Smart legal contract that consider contract's legitimacy and includes right and obligations of different parties which are legally applicable. Second school focus on the main code of smart contract run successfully and required legal smart contract. One definition that performs this job well is that of Clack, Bakshi and Braine [] that is comprehensive meaning that covers both smart legal contract and smart contract code:
A smart contract is an automatable and enforceable agreement. Automatable by computer, although some parts may require human input and control. Enforceable either by legal enforcement of rights and obligations or via tamper-proof execution of computer code.[] Sometimes, smart contract and DLT are remembered as same thing, but actually not. They are different technologies which are complementary of each other. However, there are several platforms in DLT on which smart contract can execute. By

existing computer and capability of execurting code, why have not smart contracts developed?

This question indicates the highlighted role of smart contract and DLT and the relationship between each other. It is true that computer is capable of executing an event such as payment of a contact upon satisfaction of pre-defined conditions. But, that would have meant that both partied requires to have programmed on own computer. subsequently,they need different instance to run on their computers and the version of their code and program may differ that would be other issue. What DLT has done, is that creating unique versties a to bind to parties to each other by embedded code in distributed ledger. More importantly, DLT ensure both parties to have secure transaction and contract will executed automatically without any possibility of tampering from other sides. This is the accurate description of smart contract as 'self-executed' and 'self-enforced'. With the advent of distributed ledger, needing for efficient query of diverse data and indexing entries become more important. [44]

Within the blockchain, Smart contract is actually code inside the block and it triggers by receiving a transaction or massage and send the transactioninrespose of transaction that has received then it can read, write or create contract.

smart contract is the independent factor an be half of user to preform some operations the as long to users goals. These goals are programmed inside the contracts as code. Each contact controlled own Ether, key as long lasting storage to follow the constant variables.[29]

Let us clarify more by one example: Consider blockchain network where Bob, Alice and Carol are participant and there exist 2 assets X and Y. Box use the smart contract that defines three functions: *'deposit'*: store unit of X into contract, *'trade'* send back one unit if X instead of five unit of Y and *'withdraw'* to roll back all asset into the contract. Bob start activating smart contract by calling the *'deposit'* function and moving 3 unit of X asset into contract and record it in the blockchain. Alice has 12 unit of Y, and sending transaction and 'trade' 10 unit of Y and get back 2 unit of Y and recorded into blockchain as well. Bob signed transaction to *'withdraw'* function. Contract check signiture and *'withdraw'* is called by owner, then transfer all deposits 1 unit of x and 10 unit of Y to Bob. [14]

Let us extract the features of smart contact out of this example:

- Contract has its own states and controll over own assets.
- Contract allows us to do bussiness logic in code.
- Contract is deterministic means for the same input produce same output.
- Correct smart contract describes all possible outcome.

- Relationship between participants are explained by data.
- Smart contract is triggered by receiving transaction and sent transaction afterwards.
- Since contract stores on blockchain, that can be visible by every particpcant on network.
- Each participant can get cryptographically verifiable trace of contract operation due to *sign* message.

Smart contracts have state, which can be updated by a contract when it is executed. The blockchain keeps a record of all previous states of a contract beacause overwrittng previous state would involve overwriting records earlier in the blockchain. Smart contracts can be used to implement dynamic data storage with history in the Ethereum environment.[Ethgrap]

1.4.1 Validation of Smart contract38

1.4.2 Semantic of Smart contract34

Whereas Computer programmer deal with validation of contract, Then the semantic aspect of contract is considerable case in this field and thus the *meaning* of contract should not be ambiguous.Legal contract has 2 main aspect:

1. **Operational aspect**[44]
2. **Non-operational aspect** [44]

1.4.3 Different models of smart contracts

[44] There are two models of smart legal contract. However, most implementation of smart contract is near to the term operational clause, rather then non-operational clause.

1. external contract
2. Internal contract

1.4.4 Semantic Analysis38

1.5 Block chain

Blockchains are the distributed records for digital events and they are structured as chain of linked data stored in individual database or computer over network.[p1431]

Block chain are organized into blocks. Each block is identified by cryptography hash that refers to hash of previous block. this create links between block. Thus, it create the chain of blocks wherein Blocks are held the copy of blockchain structure. The initial block created manually, is known as *genesis block* and the other node will add to block chain after a process of consensus between nodes. The distributed consensus method which allow to add new block or item into blockchain that is verified as legitimate. This process would be done by some computational work called as Proof Of Work (POW) or mining. All blocks in blockchain hold small amount of data which need to be secure before distributing over all participating computer over network and are visible by having public key for all participant but not modifiable. These data time stamped and provide the time of adding that block.)

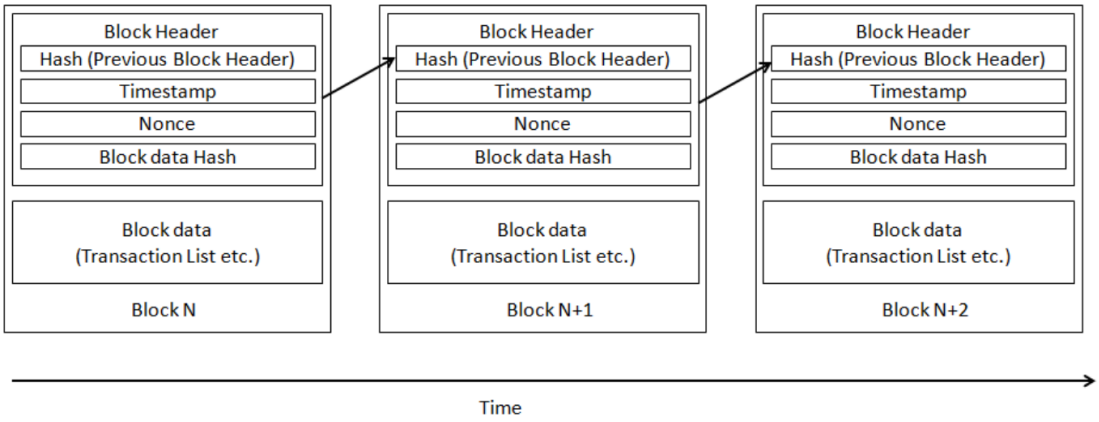

Figure 1.1: [18740]

Blockchain has four main concept: *P2P network:* node can interact with each other using private for sign transaction and public keys sre used as address that can be reachable on te network.

Open distributed ledge: It is like book containg all transactions that each node has the copy of data and there is no centrellized entity and any one can see assets and how much assets has each node.Therefor, they can decide about the validity of transactions.

Schronization: As, all node have one copy of ledger. Therefor, sycronization should prefome overall networks and by adddning new item or one new transaction should be performe in public and consensus needed to valid this transaction.

Mining: In the distributed ledger, all nodes can not receive transaction simultaneously. For adding new item in the blockchain, consensus of all nodes are needed. Thus there is need to prevent every onn node to add transaction into blockchain. Miners are the unique nodes that can add transaction to that chain. Miners attempts to validate the first transaction to add into chain. There are several distributed system based on consensus, but the one outstanding feature that makes blockchain more prominent then the others is that : Only single record stores in each participant computer. This provides transparency of transaction. Whilst storing whole records over all networks of participating computer mitigate the probability of losing infrastructure.[2016 book 491]

And blockchain is the sole technology which fulfill such properties:*(a)trustless*: There is no need to verify involved identities in transaction. *(ii) Permissionless*: there is neither permission nor controlling for participants in network. *(iii)censorship resistant*: Anyone can trades on blockchain and just cryptographic algorithm governs the operation that entities(participants) can trust it.[50]

this feature and above reason make the block chain as powerful and most secure technology in commercial events over network.[50]

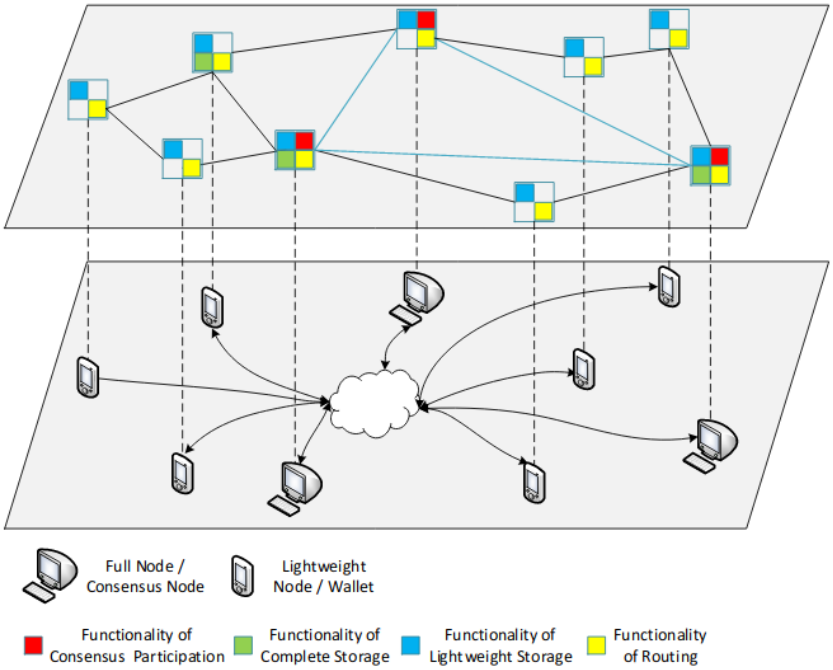

Figure 1.2: [60]

Permissionless / public Blockchain This blockchain allows any0ne to join to network and create consensus such Bitcoin nd Ethereum. oin permissionless blockchain any miner can create consensus mechnism such as proof of work, proof of stack to validate transaction. but as this mechanism is decentraliised the rate of validity of function and scalabilitiy is low.[30]

Permissioned/ Private Blockchain: this blockchain, only restricted particpant has right to validate transaction. Threfore, it provides better privacy and scalability. Unlike permissionless blockchain, this blockchain does not have mining computation to reach the consenus because all participant are known in this network. [30] **Privare Key/wallet**

1.5.1 Bitcoin

The basic goal of blockchain technolgy is to ensure people form truthworthy and ligimacy of transactions. Blockchain initially used to enhance commercial transaction through currency called Bitcoin.
Bitcoin is the digital records or cryptocurrency that is accept by users involved in transaction. in fact, Bitcoin is the financial use cases of this powefull technology.[50] Bitcoin is the list of blocks of transactions. Each block in blockchain is identifeid by hash algorithm on the top of the block. Bitcoin is introduced by consensus mechanism of blockchain, it is well known implimentation of decentralized cryptocurrency.
Bitcoin is the limit of block, wherein each block is verified by hash algorithm using SHA256 cryptgraphic on the top of block. Each block encompass the hash of its parent (previous block) in the own header that refuse to it. It forms block list wherein each block refers to previous block in the list name genesis block[1]. Altering in block implies to create new blockon the top. Since, each block contains the hash of its parent creating new block is expensive and needs proof of work that the miners allow the add new block.[1]

1.5.2 Proof Of Work

Is consensus algorithm in blockchain without any central algorithm that used to confirm transaction and add new block. With POW miners compete to complete its own transaction first into blockchain and get rewards(e.g: Bitcoin, Ether, ...).
Miners connected to blockchain and accomplish task validating transaction to add new block bc solving cryptographic puzzle[1]. TODO

1.5.3 Mining

Is an process if computation on the blockchain in order to verify and add block.[29] TODO

1.5.4 Merkler Tree

all transactions are stored in tree structure wherein leafs contain transactions and internal node contain the hash of its sub tree and single root also contains the hash of two children and represent top of tree. The purpose of bottom up hashing is that, if attacker attempts to create fake transaction into bottom of tree. This will change the node above and subsequently change the root. Thus altering the hash of block causing to register new block. Only the sequence of hash form root the leaf called the Merkler tree. [8a]

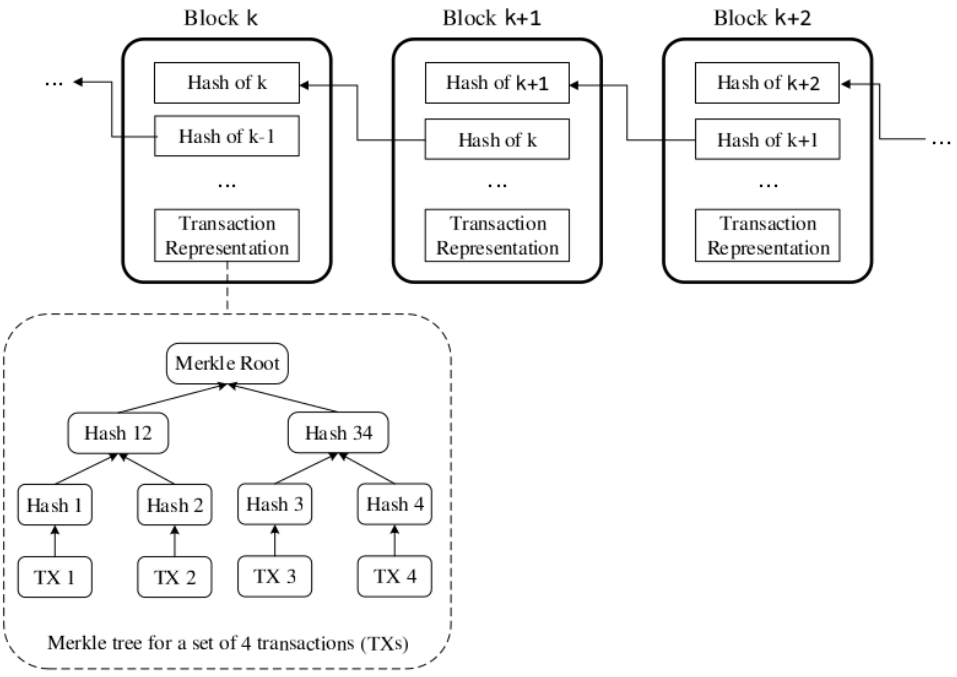

Figure 1.3: Illustration of chain of blocks and markler tree in single block [60]

1.5.5 Hash Function

It is mathematicaly method to applay cyptografic(secret writting) function. hash function is deterministic means for an input produce same output each time. altering data generate the different output. According to Wikipedia, the security level of hash function has different properties:

- Pre-image resistance: It s not easy, for initial input with /h / hash value find the output value {m} , in a way that {h = hash(m) }
- Second pre-image resistance: it is not easy, for input m_1 find another input m_2 in way that)$hash(m_1) = hash(m_2)$ and both have same result.
- Collisio Resistance : it is difficult to find two different input m_1, m_2) that have same output in way that$hash(m_1) = hash(m_2)$.

Hash function used in blockchain and it is the most secure hashing algorithm is SHA256 which that provide the combination for an given input with 256 bit length. Using SHA 256 in blockchain makes to be impossible to dublicate hash because there are diverse combination of this input with this length and it requires huge amout of computaional works. That means thare are 2^{256} hash values.

Input value	SHA 256, message hash
1	6b86b273ff34fce19d6b804eff5a3f5747ada4eaa22f1d49c01e52ddb7875b4b
2	d4735e3a265e16eee03f59718b9b5d03019c07d8b6c51f90da3a666eec13ab35
Blockchain	625da44e4eaf58d61cf048d168aa6f5e492dea166d8bb54ec06c30de07db57e1

1.5.6 Header

contains timetamp, nonce, previous hash, ...

1.5.7 Nonce

it stands for *number ued only once*. It is intger number and along with block data, number and previous hash use as input in SHA256 function to generate current block hash. We used this nonce to vary the hash of current block and without modifying the data inside the block.the By the usage of nonce, miner can generate valid hash, adding first own block into blockchain and get the reward.

1.5.8 Transaction

transfer data (asset) between users, an transaction contains inforamtion such: - The recipents which message to send. - Signiture of sender - A mount of ether to transfer -Structure value, the maximum number of computational step that transaction is allowed to do - Gaspreice, fees that should pay.[29]

1.5.9 Proof Of Stack

1.5.10 Cryptographic hash functionn

1.6 Ethereumt

Ethereum is the decentralized virtual machine based on block chain. Ethereum block chain stores codes of program and transaction data both on both on block chain. The code of contract is fed into block chain without address. Afterwards, the contract which already added to block chain and assigned address. As said before,the code of contract is unmodified, unlike contract's state that could be altered any time. In Ethereum virtual machine, node or block besides adding the transaction to ledger, render contract codes and control the state in virtual machine.

TODO Undoing smart contract etheruem ?? attack on ethereum smart contract.

1.6.1 Security in Smart contract

An analysis of existing smart contracts by Bartoletti andPompianu[7]shows that the Bitcoin and Ethereum platformmainly focus on financial contracts. In other words, mostsmart contract program code defines how assets (money) move.Therefore, it is crucial that contract execution is performedcorrectly. The direct handling of assets means that flawsare more likely to be security relevant and have greaterdirect financial consequences than bugs in typical applications.Incidents, like the value overflow incident in Bitcoin [8], or theDAO hack [9] in Ethereum, caused a hard fork of the blockchain to nullify the malicious transactions. These incidents showthat security issues have been used for fraudulent purposesruthlessly in the past. A survey of possible attacks on Ethereumcontracts was published by Atzei et al.[10]and lists 12vulnerabilities that are assigned by context to Solidity, the EVM,and blockchain peculiarities itself. Many of these vulnerabilitiescan be addressed by following best practices for

writing securesmart contracts, which are scattered throughout the Ethereumcommunity [11,12] and different Ethereum blogs. Most bestpractices mainly contain information about typical pitfalls toavoid and the description of favourable design and problemapproaches. The latter being the focus of this paper in orderto collate smart contract security design patterns.[scsecurity]

1.6.2 Vulnerabilities in Ethereum Smart Contracts

Author in this section categorized the vulnerabilities in smart contracts into three classes(solidity, Ethereum virtual machine, blockchain):

cal1	cal2
Solidity	Call to the unknown Gasless send Exception disorders Type casts Reentrancy Keeping secrets
EVM	Immutable bugs Ether lost in trasfer Stack size limit
Blockchain	Unpredictable state Generating randomness Time constraints

- Call to Unknown: One invokes the function and transfer the ether to the counterpart. If there is no signature in the given address, then the fallback function is executed.
for example: In the contract *Alice* and *Bob*:
Alice has **Ping** function and Bob invokes the Alice contract (**ping** function) via direct call. If there is any mistype in calling function such as *int* instead of *Uint* call to the **ping** return fallback function.

```
   contract Alice { function ping (uint ) returns ( uint ) ; }
contract Bob { function pong (Alice c) {c.ping (42); } }
```

Or in another contract:

```
c.call .  value (amount)(bytes4(sha3("ping(uint256)")),n);
```

function **ping** in contract **c**: called function is identified by hash signature.
amount is the much of *wei* that is to transfer to contarct **c**.
n is parameter in **c**. If the function with this signature does not exist in the contract

the fallback function is executed.

- Exception disorder: this is occurred in situation such as: executation run out of gas, call stack reaches to its limit and *throw* commend is executed. For instance in this contract:

```
contract Alice { function ping (uint ) returns ( uint ) ; }
contract Bob {uint x=0;
function pong ( Alice c ) { x= 1; c.ping (42); x=2; } }
```

There are two possibility of *call* function:

User invokes Bob function, therefore, **ping** throw exeption and execution stops. Side effect of all transactions are reverted and $x = 0$ after transaction.

Bob invokes **Ping** via call, Only side effect of invocation reverted, call returns fail, execution continue and $x = 2$ after transaction.

Gasless send: **send** function , transfer ether to contract. This function compiled the same way of *call* function without signature and returns *out of gas* exception. Assume *call* has no signature, so it invokes *callee's* fallbac function. However the upper bound of unit of gas is 2300 that is avaiable to *callee* and is executable.

```
1 -contract C {
function pay(uint n, address d) {
d.send(n) ;
  }
}
```

```
2 - contract D1 {
uint public count = 0;
function () { count ++; }
} contract D2 { function () {} }
```

Type casts Reentrancy Keeping secrets Immutable bugs Ether lost in transfer Stack size limit Unpredictable state Generating randomness Time constraints

Chapter 2

Block chain as infrastructure of semantic web

[paper6]

2.1 Distributed ledgers and indexing

[p1431] Distributed ledger based on blockchain do not have central control. Blockchains are organized into multiple blocks that initial block created manually and the other blocks are added by some consensus process between nodes. Ethereum smart contract provides possibility of to control automatically what happen with cryptocurrency on the blockchain without involving the untrusted external sources. Ethereum smart contracts has account which can normally store, update or making function with the input and output. The concept*accounts* are widely used in this study refer to agent such human and the concept *balance* refers to cryptocurenccy in blockchains.
As already said, smart contracts are time-ordered where data are stored into blocks. therefore it requires the data to index. Indexing the smart contract, gives us capability to access the data, search, analysis services on the distributed ledger and expose them to outside the worlds for more of interactions. There are different levels of indexing smart contracts: Basic level is the fundamental level for next step.It index basic entities such as account, blocks related to distributed level and data can be stored or retrieved here. In functional level, smart contracts contains alot of functional interfaces that depict the other functionality of platforms such as Ethereum.
[p1431] There are not such comprehensive vocabularies to describe such concepts but some vocabularies are purposed in this case such as:
FlexLedger BLONDiE and EthOn are formalisations of blockchain concepts as ontologies, generic across Bitcoin and Ethereum blockchains and specifc to Ethereum,

respectively. It discuss initial approaches to Linked Data indexing of blockchains, and the certi

cation of Linked Data temporal streams on blockchains, but both are very preliminary.[53]

2.2 Vocabularies

[p1431]

2.2.1 Why do we use ontology for Blockchain?

GEnerally, Blockchain is the destrubuted database that replicated over all nodes as s cloud computing arcitecture. These databases are distributed across noumerous organizations. That's why standard interpretation is needed that the data would be understandable by organizations. Interpretations are applicable via formal specification that enable verification and inference within software and applications executed on network.

This is where ontology to play to ensure common interpretation of data of shared database among different intersperses. Blockchain modeling is the one form of modeling among enterprises. blockchain modeling used different type of ontology: informal ontology, semi ontology to facilitate search and enhance better understanding of business process for developing and applying on blockchain. Blockchain modeling based on formal ontology help the formal specification to verify the operation of blockchain. On the other word, blockchain modeling based on formal ontology can help the development of smart contract to execute on blockchain.

Also, we can use ontology tocapture data within blockchain: From one hand, It facilitates the better understanding of blockchain concepts for human.On the other hand, enables interlinking with other link data to convey deductions and formal reasoning.[2]

Vocabulary used within ontology increase the transparency of transaction in a way that by describing transaction in the context of linked data comfort the graphical represantation of location of such transaction. Thus , it increases also the capability of analysis by users.

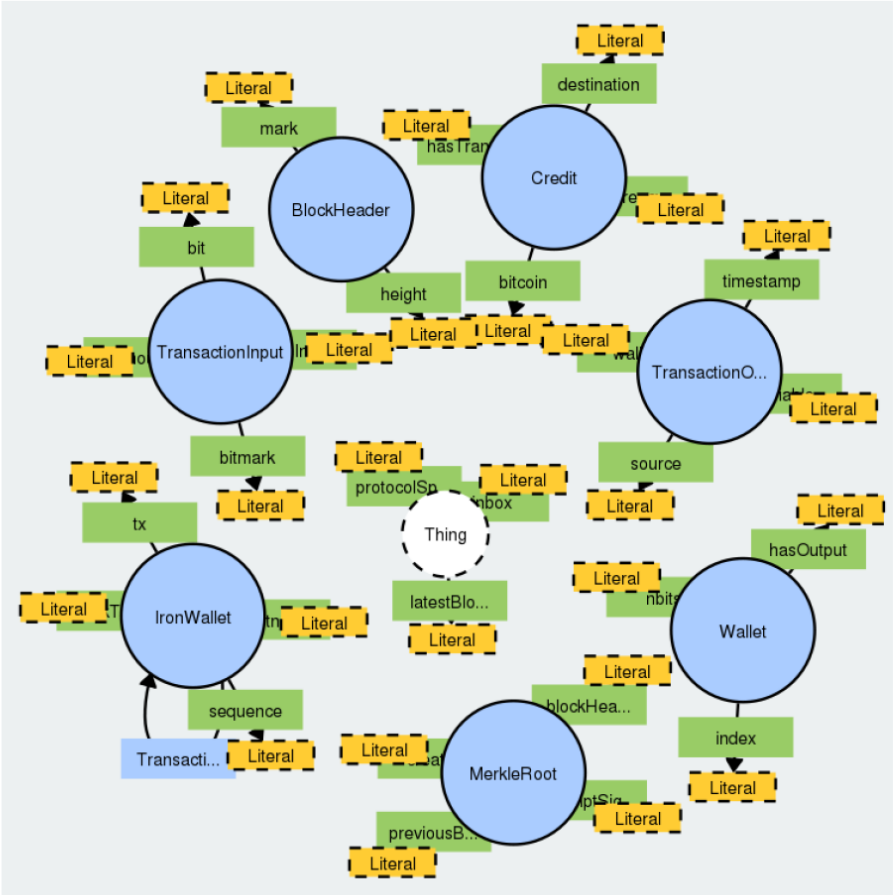

Figure 2.1: Illustration of Ontology diagram(2)

2.2.2 Linked Data

TODO when information can represenr in linked data, querying about realted information can be easily discovered in linked data. It is based on 4 rules:

URIs(Uniform Resource identifier) as names.

HTTP to search for names.

(SPARQL, RDF) when a user search for something, provides related information.

Link to other URLs to provide more information.

2.2.3 Resource Discription Framework

It is w3c specification that model information. TODO

2.2.4 SPARQL

TODO

2.2.5 Ontology and OWL

TODO Ontology Wen Language is made to represent knowledge bout things and the relations among them. OWL is computational logic-based language ,means the language modeled in OWL can operated in computer program like negation, intersection and so on. TODO

2.2.6 Vocabulary in Distributed Ledger

In ord is replecated over allnr to generate link data, it requires to use a standard ontology or vocabulary to explain the blockchain concepts. Interfaces between distributed ledgers and the Semantic Web are still on early stage in this. there are some systems that have such voicabularie includes: Flex Ledger, EthOn, BLONDiE[p1431].

FlexLedger: describes HTTP interfaces to blockchains, with a standard vocabulary and responses of these interfaces[p1431]. FLexLedger is a protocol for decentrelized ledger and interfaces which represent ledger creation, querying and data model using JSON-LD. However, Flex Ledger does not have explicit vocabulary about ontology nor having concrete ontology for itself.
TODO It is also worth mentioning that the Flex Ledgermeta and the content data are stored together in the same graphwhereas the GraphChain blocks content is store outside the blockin a separate graph. For all the reasons stated above, GraphChaincannot be considered as an implementation of Flex Ledger.[p1171]

EthOn is an OWl ontology that describes blockchain concepts such as *"blocks, accounts, message"*and relations such *"has parent block"*.[40] TODO

Blockchain Ontology with Dynamic Extensibility(BLONDiE)

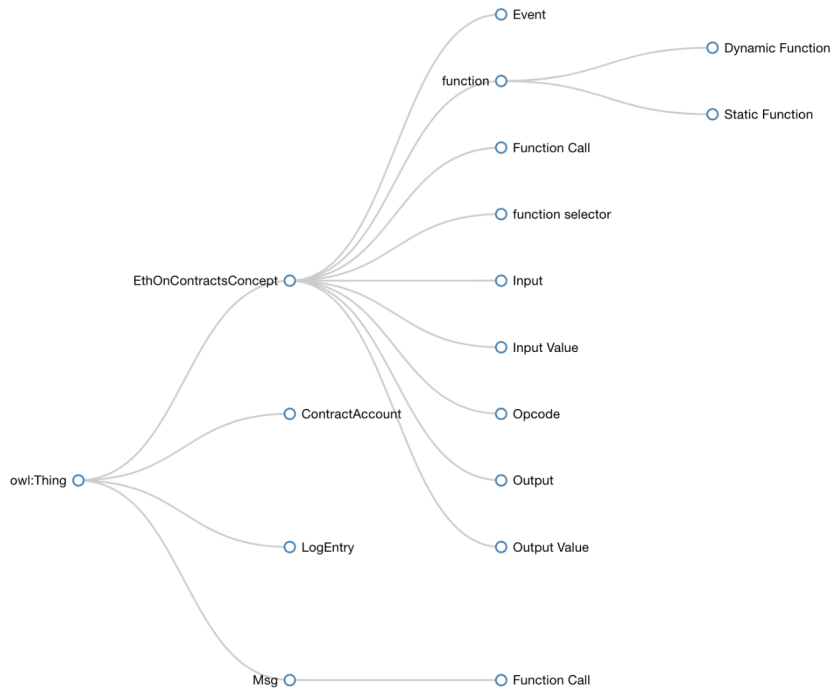

Figure 2.2: EthOn Classes(negotitaion)

BLONDiE is another OWL ontology for describing the Blockchain structure like EthOn. But is is more generic then EthOn. For example EthOn and BLONDiE bothe defined some terms such 'account', 'block', 'transaction' and some attributes such as 'transaction payload' or 'miner adddress'. BLONDiE defines some other concept for different Blockchain such as 'BitcoinBlockHeader' and 'EthereumBlockHeader' as sub classes of 'BlockHeader'. At the moment, BLONDiE supports two crypocurrencies like bitcoin and Ethereum where all link and relation between objects and attribute represent in RDF(Resource Description Framework).[p1431]

2.2.7 Linking Blockchain in BLONDiE[45]

TODO On a general point of view, the Web is a huge global graph of connected hypertext documents by hyperlinks. On a similar way, there are some projects working in the idea of connecting different Blockchains. Where the main goal is to create the Internet of Blockchains. Since payments are different from plane information, it can be copied and replicated, but money must not be. The interledger protocol (ILP)8 is for payments across payment systems. ILP models the world of finance as a giant

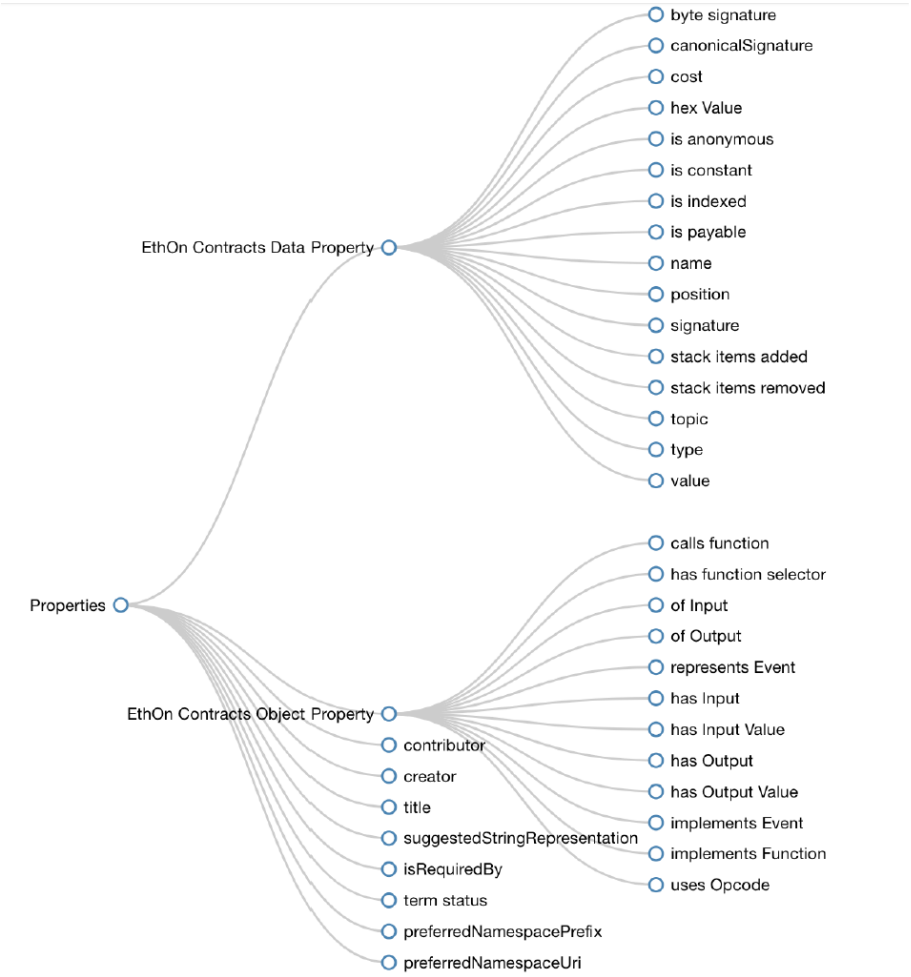

Figure 2.3: EthOn classes(negotition)

global graph of ledgers connected by liquidity. The systems providing this liquidity are called connectors and a key feature of ILP is that these connectors do not need to be trusted, meaning anyone can create one [21]. Currently, centralized exchange platforms are used to exchange one Blockchain-based currency for another. Cosmos9 is a network of Blockchains, organized on hubs and zones. Zones plugged into a central hub and each zone maintain its each governance. Allowing the decentralized exchange of tokens from one Blockchain to another. Polkadot10 is a new network that aims to provide interoperability between private and public Blockchains. Allowing extensibility and scalability. It defines a heterogeneous multi-chain, provided to an absolute minimum of security and transport. Scalability is addressed through a divide-andconquer approach. The heterogeneous nature of this architecture enables many highly divergent types of consensus systems interoperating in a trustless, fully decentralized federation, enabling open and closed networks to have trust-free access

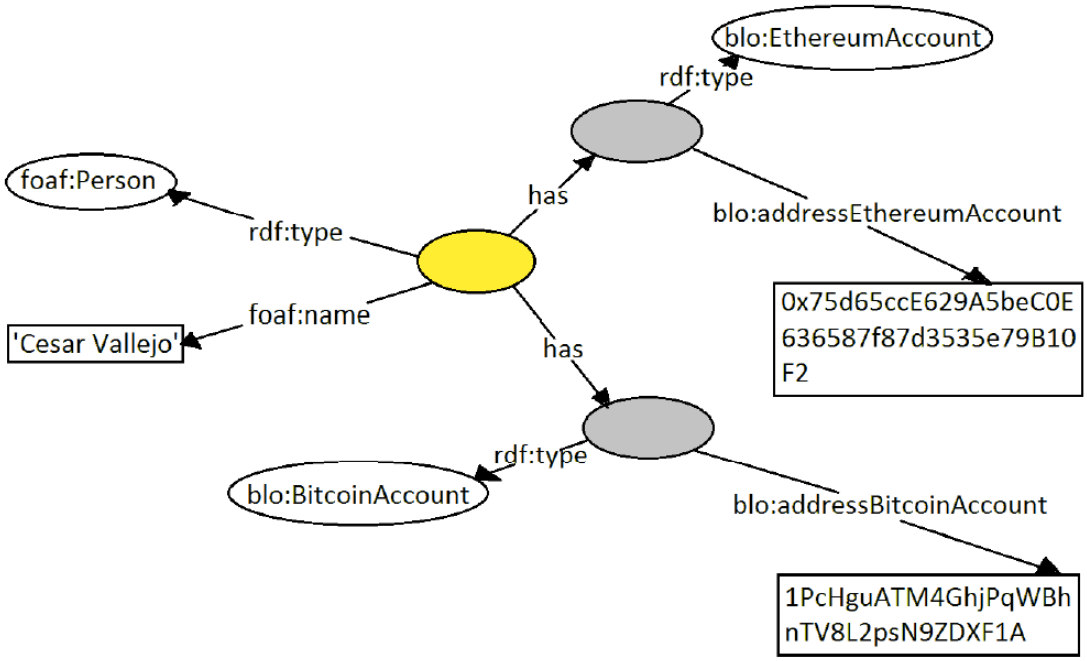

Figure 2.4: BLONDiE usage example[45]

to each other [23]. Two more similar projects are Blocknet11 and Supernet12. All these projects show evident need of semantic empowerment. The same need that the web still has to really jump from an only Syntactic Web (Web of documents) to a Semantic Web (Web of data). While how and where to locate and enable semantic data is not completely clear, it is obvious that Semantic Web fits as a prominent set of technologies to be used here. In Figure 4, we present datasets that have been published in Linked Data format by contributors to the Linking Open Data community project and other individuals and organizations. In a similar way, Blockchains-based systems and its corresponding datasets can be published and linked.

2.2.8 Ontology-based smart contract in BLONDiE[45]

Data models are used on Ontology-Based Object-Oriented Enterprise Modelling [24]. In a similar manner, Kim and Laskowski [25] propose that ontologies can contribute to Blockchain design. Their approach can aid in the development of Smart Contracts that execute on the Blockchain. They come up with a proof-of-concept using

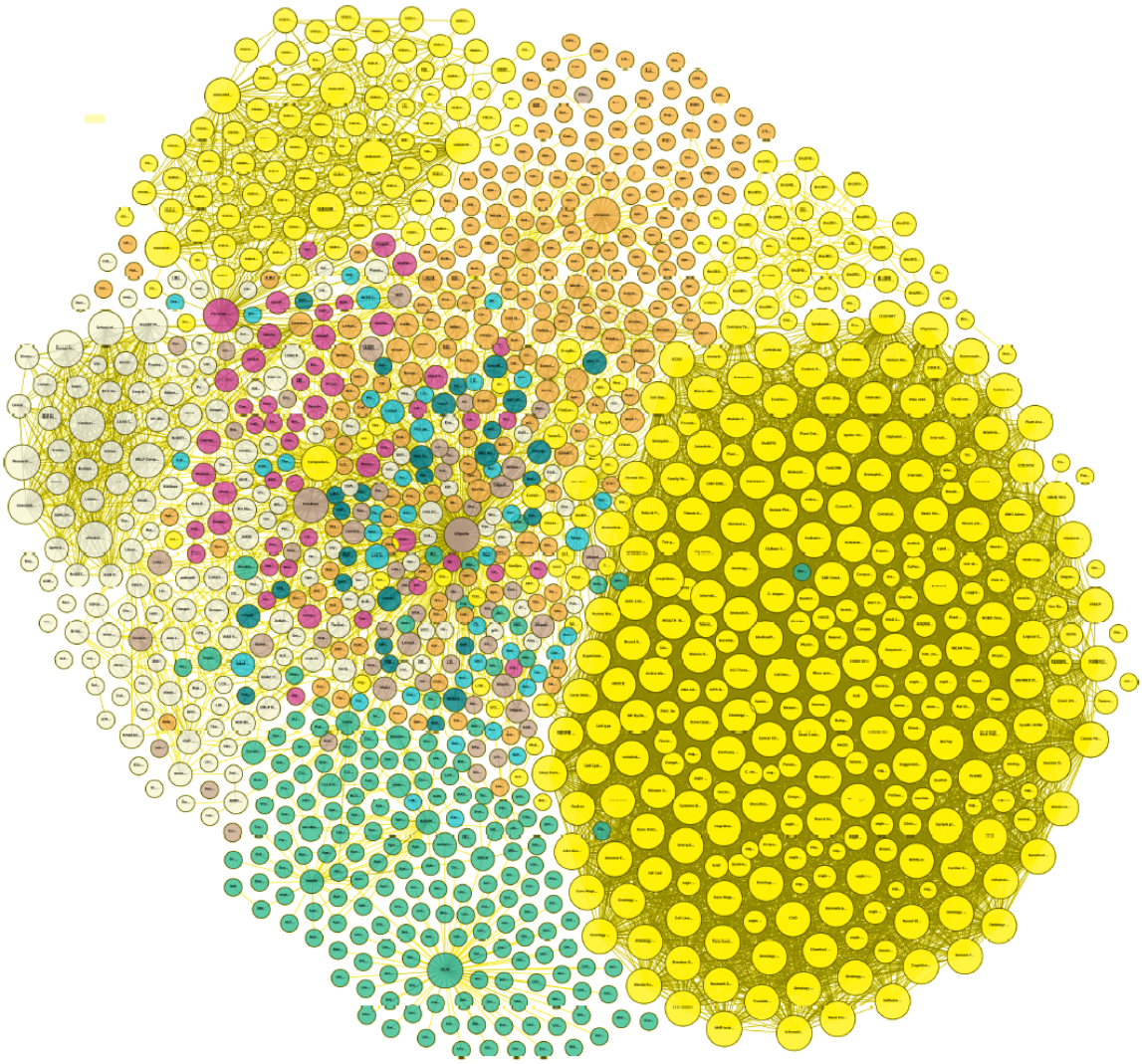

Figure 2.5: Linked data Diagram[45]

Ethereum and TOVE Traceability Ontology. This interesting idea can be replicated for other ontologies too. Better data standards, business practices and processes for developing and operating Blockchain are achieved. It also aids in the formal specification for automated inference and verification in the operation of a Blockchain.[45] TODO

2.2.9 Storage and RDF

The main goal of data over web is to make data machine readable format on web. It causes to interpret data in suitable formats to be useful in a way that be human

readable too.
JSON-RPC is remote procedure call protocol in JSON. his protocol allows to send data to server without needing to response. Ethereum blockchian which has properties such as pay fee with Gas, there are different was of storing data in RDF triple Most wallet in Ethereum in in JSON format that is seasily convertble to RDF and fornt-end is made up with HTML technology that we can use it to embed data on website interface as Micro data, JSON-LD. The main focus of research is the storage of data on blockahain itself.There exist limitation of storage in Ethereum blockchain and to fees. that's why, it is suitable to use compact RDF serialization.[45] There are some methods to store data on Ethereum blockchain that are summarized in table bellow:

```
{
"id":1,
"jsonrpc":"2.0",
"result": {
    "number": "0x1b4", // 436
    "hash": "0xe670ec64341771606e55d6b4ca35a1a6b75ee3d5145a99d05921026d1527331",
    "parentHash": "0x9646252be9520f6e71339a8df9c55e4d7619deeb018d2a3f2d21fc165dde5eb5",
    "nonce": "0xe04d296d2460cfb8472af2c5fd05b5a214109c25688d3704aed5484f9a7792f2",
    "sha3Uncles": "0x1dcc4de8dec75d7aab85b567b6ccd41ad312451b948a7413f0a142fd40d49347",
    "logsBloom": "0xe670ec64341771606e55d6b4ca35a1a6b75ee3d5145a99d05921026d1527331",
    "transactionsRoot": "0x56e81f171bcc55a6ff8345e692c0f86e5b48e01b996cadc001622fb5e363b421",
    "stateRoot": "0xd5855eb08b3387c0af375e9cdb6acfc05eb8f519e419b874b6ff2ffda7ed1dff",
    "miner": "0x4e65fda2159562a496f9f3522f89122a3088497a",
    "difficulty": "0x027f07", // 163591
    "totalDifficulty":  "0x027f07", // 163591
    "extraData": "0x0000000000000000000000000000000000000000000000000000000000000000",
    "size":  "0x027f07", // 163591
    "gasLimit": "0x9f759", // 653145
    "gasUsed": "0x9f759", // 653145
    "timestamp": "0x54e34e8e" // 1424182926
    "transactions": [{...},{ ... }]
    "uncles": ["0x1606e5...", "0xd5145a9..."]
  }
}
```

Figure 2.6: Ethereum structure

2.2.10 Semantic Blockchain

By increasing of the usage of blockchain technology recently, the need for semantic reasoning on distributed ledger is on crease as well. The blockchain is the best platform to utilize semantic web principle inthis technology and add new trusted property to dataset. that makes new dataset so thruthworthy. Using semantic web technology on blockchain in novel idea and the way of how apply this technology in blockchain and smart contract is also as controversial issue.

Way	Short explanation	Advantages	Disadvantages
Transaction Data Property	Property existing in each transaction on Ethereum	- Not fixed size - Cannot be modified	- Expensive - Stored on hexadecimal format - Is not SPV friendly
Contract Storage	Contract state flexible database. Key-value store	- Not fixed size - Easily accesible	- Expensive - Information is modifiable
Event Logs	Historical raw data	- Cheap	- Not accesible for smart contracts. - Data generated by the Smart Contract
External Storage (IPFS)	Storing it externally and keeping the identifier using one of the above methods	- Unlimited size	- Not guaranteed that data will not be removed

Figure 2.7: Storage of RDF data on Ethreuem summery

There are some **definition of semantic blockchain:**
Semantic blockchain is the representation of stored data on distributed ledger using linked data.
Semtcic blockchian is the applying semantic web standard on blokchain that these standards is based on RDF.

2.2.11 semantify Blockchain process

Semantic blockchain or semantic distributed ledger has effect on industrial world and subsequently , the result lead to start developing new application and framework to combine two worlds: thee are some way to semantify blockchain: -Mapping the basic blockchain to RDF making usege of vocalbularie, ontology and so on. - Storage of data in blockchain is expensive, The only wey is to store the hashing point to data set in blockchain and then share RDF on blockchain. - Create semantic blockchain that internal data exchange protocol is based on RDF.

TODO SemBlocRouter [45]

2.2.12 Semantic Ontology Mapping

[p1341] To generate RDF, ot is need to ap the basic blockchain etities to relevant semantic web terms, concepts and ontology. In order ot make query more efficent BLONDiE used some ways such as:
Firstly, records realting to block and transactions both, have been completed with attribute for hashing to provide direct mapping between contents of index and address

of entities on blockchain, Secondly, transaction with link to blocks or smart contract and input to smart contract.
Blockchain stores just binary form of each contract with metadata. it requires to have the relevant Application Binary Interface (ABI) specification in JSON form to interact with such contract. This specification is created when the smart contract is compiled and stored in blockchain. Indexing of smart contract on blockchain is in binary form. IN order to interact with contract Application Binary(ABI) Interface specification is needed. This specification is in the form of JSON and is created when smart contract is compiled and stored in blockchain. The ABI determines all functions of contracts and description about input, output parameter for each contract.

```
{ "badge_abi": [
    {
      "constant": false,
      "inputs": [ { "name": "imageurl", "type": "string" } ],
      "name": "changeImageURL",
      "outputs": [],
      "payable": false,
      "type": "function"
    },
    {
      "constant": false,
      "inputs": [ { "name": "tag", "type": "string" } ],
      "name": "removeTag",
      "outputs": [ { "name": "success", "type": "bool" } ],
      "payable": false,
      "type": "function"
    },
    ...
```

Figure 2.8: Smart contract of ABI

By the use of ABI, it generate RDF to index smart contract. Author()model smart contract as follow:
- A contract as *msm:Service*
- A function as *msm:operation* with (*msm:hasInput* or *hasOutPut*) related to suitable *msm:MessageContent*
- A msm:Service records contains the blockchain address as an attribute to keep the index into blockchain.

2.2.13 Vocabulary in Smart Contracts

[p1341] As already menthiond, EthOn and BLONDiE are both similar concept that can be used for smart contract. as, smrt contract is the executable software, so the semantic and vocabularies that are applicable for other software too.
There are many works on semantic annotation of web (see [AnnaFensl]) and HTTP APIs which may enable us to annotate smart contract as well. However, the caontracts are not Web API and implementation ma differ but the main concept does not differ. On the other word, the vocabularies used to annotate web services, are usable to annotate smart contract too. It is seems that the Combination of distributed ledger with smart contract and web service due to profitability become common.
Here, Author(TODO negotation author) used Minimal Service Model(MSM) with Ethereum Ontology(EthOn) to describe some concept of smart contract like Gas and son on that make us enable to answer queries such as 'finding a smart contract with the minimal gas payment'.

2.2.14 Minimal Service Model

[negostation] The MSM defines a service that has Operations. Operations have input, output and fault MessageContent that may include mandatory or optional MessageParts. MessagePart provides support for finer-grained input/output discovery, as available in SAWSDL, OWL-S and WSMO

We present one example that Author() used to clarify more the meaning of MSM in blockchain: Inthis example, educational data is used to store in blockchain using Ethereum *web3* library.
Author()considered that every block add to blockchain and retrieve trascation within block. If transaction contains smart contract.Then it retreive contract address using Ethereum API. Then, it sves the smart contract address, Application Binary Interface(ABI) as triple in RDF. ABI describes the name of smart contract and way of calling them. Auther, then saved each method in ABI as an RDF triple based on MSM ontology as follow:

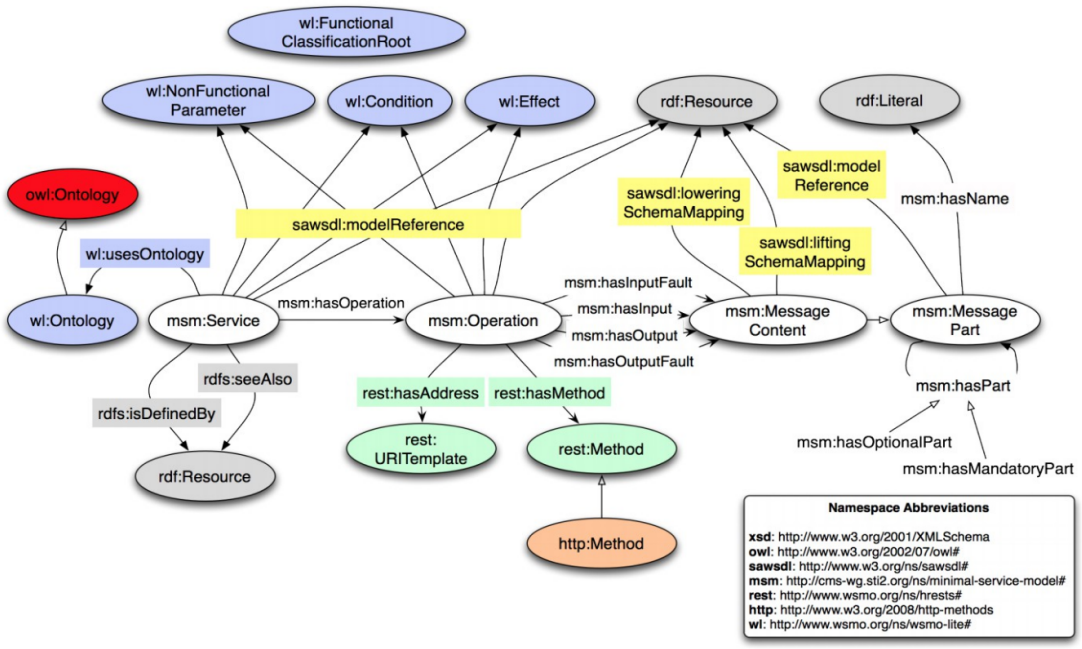

Figure 2.9: Illustration of Minimal Service Model Ontology nrgotation

Smart Contract Methods	MSM
Smart Contract ABI	msm:service
Function	msm:opreation
Input	msm:messagecontent
Output	msm:messagecontent
Gas	msm:gas(from EthOn Contract DataProperty 'cost')

TODO MY example

2.2.15 Implementation of BLONDiE

[p1341]

2.3 Link Data TODO

[45] According to Tim Berners-Lee, Linked Data is the Semantic Web done right, and the Web done right. When information is presented as Linked Data, other related information can be easily discovered and new information can be easily linked to it. It is based on these 4 rules [8]: i) Use URIs (Uniform Resource Identifiers) as names for things. ii) Use HTTP URIs so that people can look up those names. iii) When

someone looks up a URI, provide useful information, using the standards. (RDF, SPARQL) iv) Include links to other URIs. so that they can discover more things.

2.3.1 RDF[45]TODO

The Resource Description Framework (RDF) is a family of W3C specifications, it is a foundation for processing metadata [9]. On web resources, RDF is used as the standard way to describe and model information. Three object types conform the basic model [9]: i) Resources. The things that where RDF expressions are used to describe them. ii) Properties. The specific description of a resource, it can be an attribute or a relation. iii) Statements. The conjunction of a resource, a named property and the value of that property. These three elements form the RDF statement of a specific resource. They are expressed in the form of subject, predicate, object and commonly called triples. Triples create a basic graph structure of data. Definition 4. RDF triple. An RDF triple t is defined as a triple t = (s,p,o) where s 2 U [B is called the subject, p 2 U is called the predicate and o 2 U [B[L is called the object. where: U (Set of all URIs), L (Set of all literals) and B (Set of all blank nodes) [10].

2.3.2 SPARQL[45]TODO

SPARQL (recursive acronym for SPARQL Protocol and RDF Query Language) is a W3C recommended semantic query language for datasets, it is made for retrieving and handling data stored in RDF format. Thus, the queries are working over a graph structure defined by the RDF data, where the result will also be a graph or a subset of it.

2.3.3 OWL[45] TODO

An ontology is a set of explicit formal specifications of the terms (classes or concepts) in a domain and relations (properties or roles) between them [11]. When a set of individuals (instances) is available, it is known as a knowledge base. Ontologies define a common vocabulary for researchers that want to share information in a domain and they include machinereadable definitions of basic concepts and its relations [12]. Web Ontology Language (OWL) is a language made to represent complex and rich knowledge about things, sets of things and existing relations between them. OWL documents or OWL ontologies are usually published on the WWW and make reference or be referred from others OWL documents. OWL is a computational logic-based language meaning that knowledge modeled in it can be exploited by computer

programs, e.g. to make implicit knowledge explicit available [13]. It has a rich set of operations like union, negation, intersection, etc. Its logical model allows the use of reasoners that are checkers of consistency between ontology elements.

2.3.4 WEB3 [45]

2.4 Retrieve Information form semantic blockchain

[paper6] Reasoning and knowledge representation in semantic web are based on Description Logic(DL). Description logic has some elements such: classes and properties , link between different object and etc. Ontology is the study about the objects and relations between these objects or entities. An individiusl axiom also known as *fact* forms a assertion box. Asssertion box and ontology together create knowledge base. Our work extens, in the keepong with the retrieving thge most relevent resources for given query by requester, where both query and resourcse are matched called as semantic matchmaking. This proccess leads to two approach *full macth*, *no match*.TODO First, Resources and query may not matched. So one contradiction determines the location of infliction with resources. if one retracts, query TODO Second, if resources and query matches. But resources are not full match in .....

Logic based matrix

2.4.1 Semantic Blockchain Architetuer

This method purposed framework based on semantic discovery layer built up on basic blockchain. It is striking to see the main features of such a blockchain as follow:

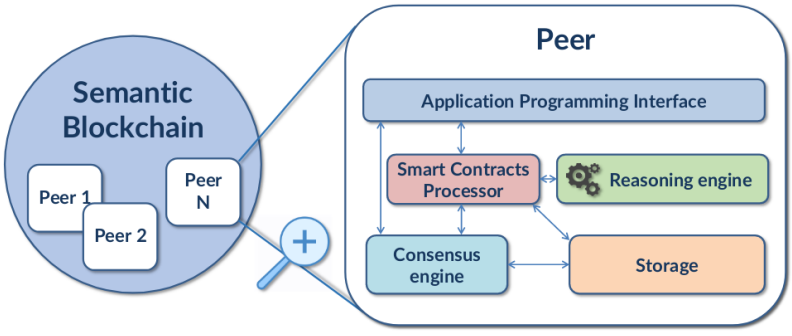

Figure 2.10: Framework Arciteture (semantic web of thing)

Peer agent is identifed bx public key and its acounts. Each agent is able to enforce semantic discoveryto transfer asset between annotations that are stored in blockchain. And each peer is the integrated of matchmaking and reasoning engine for semantic discovery.

Asset represents the resources and service instances based on domain ontology.
Smart Contracts are the integration of matchmaking and reasoning engine.
Consensus Engine allows to validate transaction in blockchain.
Storage built up *markle tree* out of transactions including semantic ones to efficient detection of erroneous change in transaction.

2.4.2 Semantic in Consensus Protocol

[paper6]

In semantic blockchain, EthOn or BLONDiE concepts and vocabulary to facilitate integrity and develop resource discovery. EthOn concept used W3CRDF schema and web ontology language to describe blockchain contract concepts. By the use of This concepts, it is possible to compare request with different recourse with respect ot semantic annotation of shared domain ontology. Smart contract semantic is represented by service layer that gives correct description of discovery outcome. Thus, it raise the trust of discovery process for users. In semantic blockchain, Author() used domain ontology to maps to EthOn contract. These concepts used for OWL-s ontology, indexing and invoking smart contract on Blockchain via URIs.[semdec] Semantic blockchain also preform operations such as: registration, discovery, selection and execution that are implemented as smart contract. Author here focused on semantic mismatching as a significant feature of semantic blockchain with respect to basic blockchain. This allows to compute the semantic distance between resources and queries with the same ontology. The logic-base matrix enforce the semantic ranking liat of element of the query.

A:Resource Registration: As already mentioned, Smart contract desined using OWL specification. It used concepts and entity defined as class in OWl language and relationship between entitites defined as object properties with regard to ontology. In order to construct smart contract semantic , requires framework to design such contracts.
Multiple resources stored on blockchain and Each domain is related to a different ontology that provide vocabularies for annotating resources. Resources also are categorized by attributes such as: URI of reference ontology to retrieve the resource,

semantic annotation in OWL that describe resources, resource price. In order to register resource on blockchain, a user is required to register an account with public address and related private key to call the smart contract as parameter. In ontology, a user describes as class corresponding to an account along with other attributes such as address, private key and other details.
To acheive ths, A new ontology called *EthereumContractConcepts* is implemented containing *EthereumContract* data prperty. A *ContratcAccount* consist of lisencenumber, date, smart contract owner and state. After all, smart contract is developed with vocabulary as domain-based-ontology. [semdec]

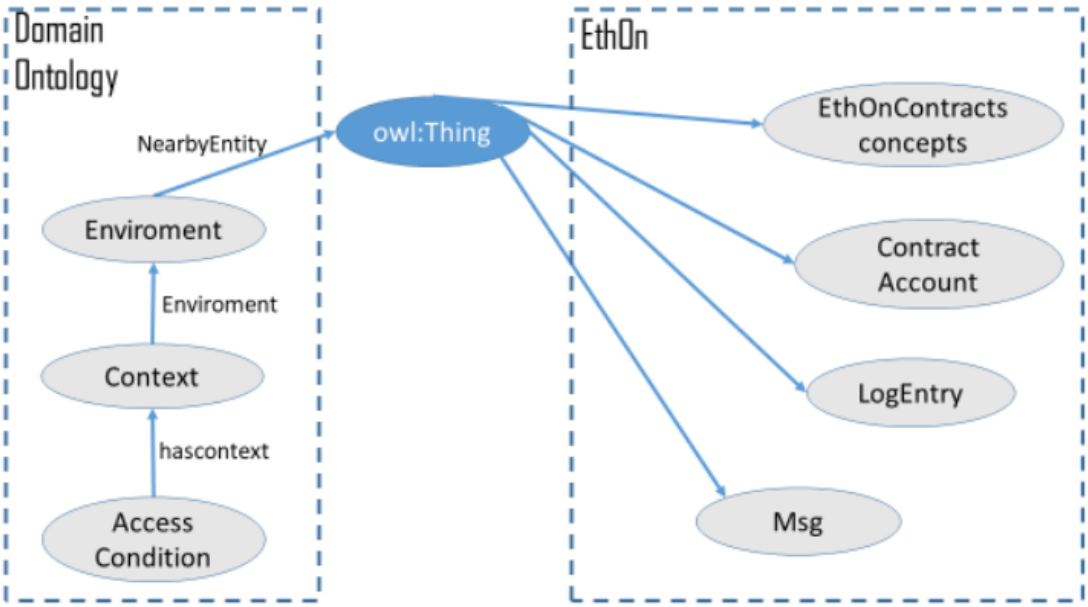

Figure 2.11: Domain ontology semdec

B: Smart Contract Author used the EthOn and OWl concepts build semantic web service for smart contract. Autho() used OWL-S ontology that makes functionalities such discover, invoke and monitor by providing some additional vocabulary along with smart contract concepts facilitate query to finding smart contract. The OWL-S ontology provides three type of description about service as follow in figure():

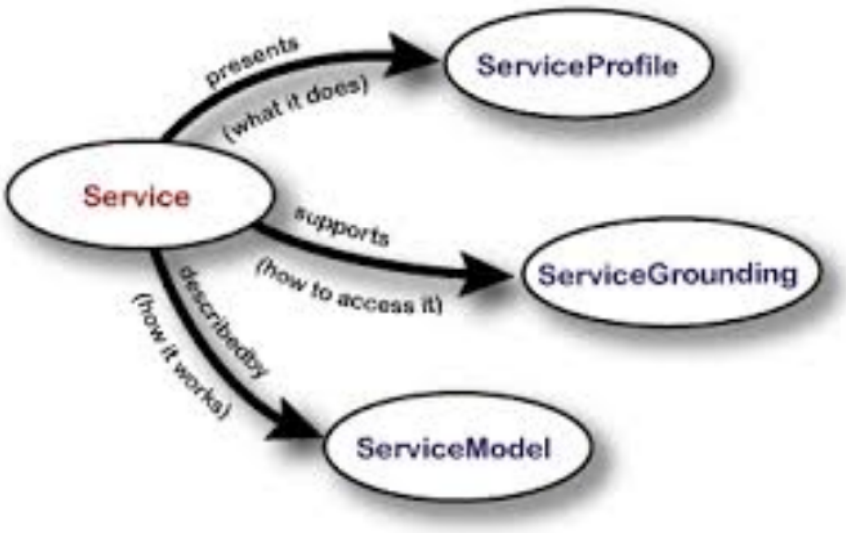

Figure 2.12: Service-based smart contract

By the use of EthOn concepts, we can monitor the added block to blockchain. When a transaction has a smrt contract, ir is retreived the store contract address via API, then the address of smart contract and Application Binary Interface(ABI) as a triple in RDF store. ABI consist of the smart contract method and way of calling it. Methods are stored in ABI as an RDF triple with regard to OWL-S ontology. Table bellow show the OWL-s vocabulary to describe the smart contract[semdec].

TABLE I
EXTENDED OWL-S ONTOLOGY FOR SMART CONTRACTS

Smart Contract Methods	**OWL-S**
Smart Contract ABI	owls:service
Input	owls:ServiceModel
Output	owls:ServiceModel
Function	owls:ServiceProfile

Figure 2.13: OWLS ontology for smart contract

Semantic Discovery of Semantic Smart Contract: In order to search for a smart contract or an item, The requester send the request to n nodes randomly specifying:
- URI of domain ontology to determine resource and vocabulary of both resource and request which to be retrieved.
- Semantic annotation of smart contract in OWL language.
- Maximum price p_max that requester pays.
- Minimum semantic s_min threshold in $[0, 1]$ interval, 1 is related to full match and 0 is the mismatch. - maximum result r_max is to be returned. - Address of requester.
Author() here used the *gossip* approach to propagate the request. The nodes which received the request preform 2 operations at the same time: First, preform semantic matchmaking of own resource with the request and provides a list of max result r_max satisfying both semantic relavance score s_min and cost $p_i <= p_{max}$, is returned.
Second, send the request to other n nodes randomly, the other nodes preform at the same way until reach the m thresholds. the request continue until reach the $\sum_{i=1}^{m} n^i$ with n and n parameters. Then do not forward the request and preform match making locally.
Afterwards, the nods send back the result to main requester at the specified address[semdec].

2.4.3 Resource Selection:

After receiving all results, requester selects the best resource and smart contract, sending message to resource owner and contextually payment.the receiver responses with the proper resource representation relied on meaning of uri[4]. The resource discovery and retrieval interaction is shown in Figure() and Each associated transaction is recorded on the blockchain[4].

2.4.4 Indexng of Smart Contract

As already mentioned, distributed ledger, does not have central registry due to their structure. Contract in this distributed ledger is not directly queried. and blockchain is the time-ordered structure where data exist on multiple blocks. In such distributed structure, developing the indexing process for smart contract is necessary. Indexing system provides a bility to analysis and search service on blockchain and displays the

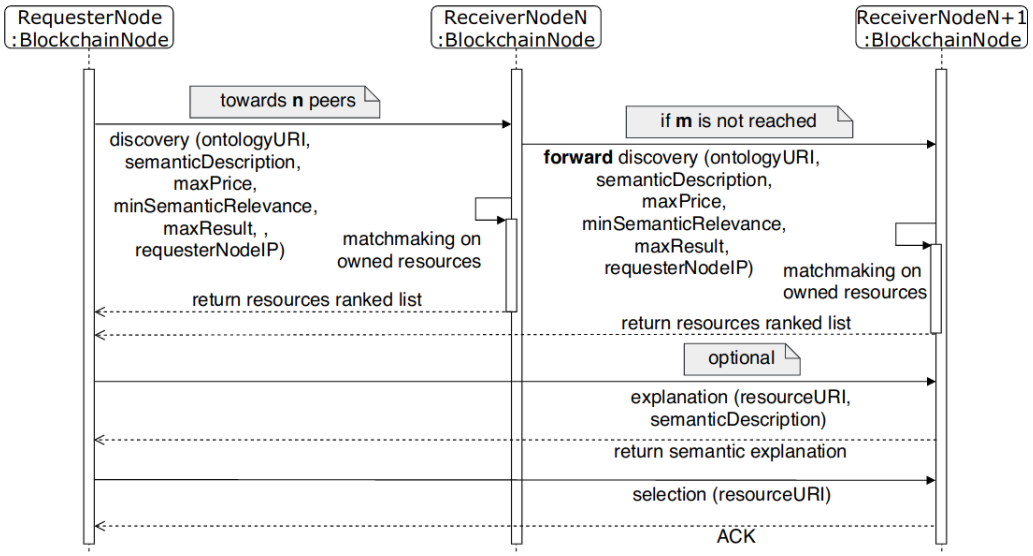

Figure 2.14: Resourse Discovery (4)

outside. There are differnet level of indexing such as low level to index basic entities such as account, transaction, block and top level for more functional operations is indexed.

2.4.5 EthOn

Is an ontology that describe blockchain concepts such transaction , account, block using w3RDF schema and ontology web language OWL to describe the relation these objects on blockchain. In order to model data and be understandable for resource, EthOn and some semantic such as "hasParentBlock". EthOn in at infancy and should develop more to have smart contract concept, functions, events, input and output and etc. EthOn accepts consider just some relation between smart contract and Ethereum framework. There are multiple tools and language that can annotate smart contract. In this study Author() used OWL-S ontology due to some significant feature such as felxibility, widly discription of servic e and non- restriction .... The main goal is to develop to support the smart contract too with respect to semantic web service context. For reaching to this aim, web Araph chain is PI and smart contract can be represent as a executable function in distributed environment. Semantic web describes message, service and entities in machine readable format that can support logical reasoning based on semantic web service description. It helps mapping between different ontologies which faiciliate the concepts transformation among ontologies[semdec].

2.4.6 Graphchain

Graphchain is blockchain mechanisim on the top of the RDF graph abtratc data type. An absttract RDF graph is the set of values and operations permitted on these value. This behaviour allows us to create be on the top of abstratc RDF graph data type. that gragraphsph hcian is: *linkde chain of RDF graph is ordered linearly by functional relation refering to pervious graph in the chain.. THe first graph called genisis unit. the linkf chain of RDF graph that used the blockchain mechnisim such as hashing, linking of graphs, acheiving consensus, signiture and so on.On the stored graphs in the multiple distributed database of chain RDF graph. The main benifit of this approach is the user can work with the chain of graphs with fully developed methods succh : SPARQL for querying, linked data for access nodes on the graph and reasoning by the use of ontology and son on.*

As a mean of implementing a service, each SC with cor-responding distributed ledgers contain variety of information related to the business logic of the service. However, such distributed ledgers do not have a global registry; and due to their structure, contents in such ledgers are not straight-forward to be queried [14]. As BCs are strictly time-ordered structures where data exists across multiple blocks (as inevitably it must),there currently is no convenient way to identify, group or query it. Thus it is necessary to develop an indexing mechanism for SCs and distributed ledgers. This indexing system, where present, provides the ability to search and analyze any services deployed onto a BC, and potentially to expose them to the outside world for interoperability. As BC-based IoT services (i.e., IoT DApps) contain millions of connected devices operating in different scenarios, a BC-based platform for these IoT DApps needs to manage a considerable number of SCs and distributed ledgers in different domains and contexts. For example, in a smart city, IoT DApps(i.e., corresponding SCs) which utilize crowd sensors, parking sensors, etc. need to be managed for efficient co-operation.For this purpose, there are different levels of granularity atwhich indexing can be carried out. At a low level, it necessary to index the basic entities of a distributed ledger, blocks,transactions. At a higher level, more functional orientation for SCs should be also indexed (i.e., SC indexing)[semdec].

2.4.7 EthOn

There is limited research on integrating and developing se-mantics in BC, particularly for SCs. The only notable work isthe Ethereum ontology (EthOn) which describes BC

concepts(e.g.blocks, transactions, contracts) and relations using W3CRDF Schema and the Web Ontology Language [15]. EthOnenables some basic semantics in BC such as has parentblock? query. Its main goal is to serve as a data model andlearning resources for understanding Ethereum. While activelydeveloped, EthOn is, at the time of writing, at an early stage. Ithas recently been envisioned that the EthOn ontology shouldinclude extension for SCs, which further describes valuableinformation about concepts and properties specific to SCs suchas Functions, Events, Inputs, Outputs and Opcodes. Classesand properties in Ethereum ontology are presented in Fig. 1.The EthOn concept covers only the relations between SCsin Ethereum framework. Given that SCs themselves are es-sentially executable software, it is necessary to represent theirsemantics using ontologies leveraging vocabularies that arealready defined in other forms of software. Indeed, there is Fig. 1.EthOn Message concepta wealth of existing work on the semantic annotation of Webservices that can be used for SCs semantics. Also, there aresufficient ontology language, existing ontologies, and toolsthat can be utilized for annotating SCs. For instance, somewell-known ontologies for semantically annotating servicesare OWL-S, WSMO, SAWSDL, WSMO-Lite when it comesto WSDL services, and MicroWSMO, and SA-REST forWeb APIs. We choosed OWL-S as a service ontology basedon five reasons: i) Recommended by W3C ii) Flexible, norestriction of the way to implement the web of services.iii) Rich description of the service composition process. iv)The implementation is an orchestration mechanism and finallyv) The service composition process description is near thelanguage programming so that it will be accessible to aimplementation orchestration program.As being too simple and limited usage of the EthOn ontol-ogy, our ultimate goal is to extend the existing EthOn witha service ontology to support Ethereum SCs by consideringthe SCs under the context of semantic web services. For thispurpose, both Web APIs and SCs can be seen as executablefunctionality exposed in a distributed environment for arbitrary(suitably authorized) third-parties to call. The notion of aSemantic Web Service [16] [17] is an important concept inour paper. It describes services, messages, and concepts in amachine-readable format that can also facilitate logical rea-soning. Thus, service descriptions can be interpreted based ontheir meanings, rather than simply a symbolic representation.Provided that there is support for reasoning over a SemanticWeb Service description (i.e. the ontologies used to groundthe service concepts are identified, or if multiple ontologiesare involved, then there exist alignments/mapping betweenontologies to facilitate the transformation of concepts from oneontology to the other) [18]. Section III will describe the OWL- S ontology and how it is mapped to the SC EthOn extension[semdec].

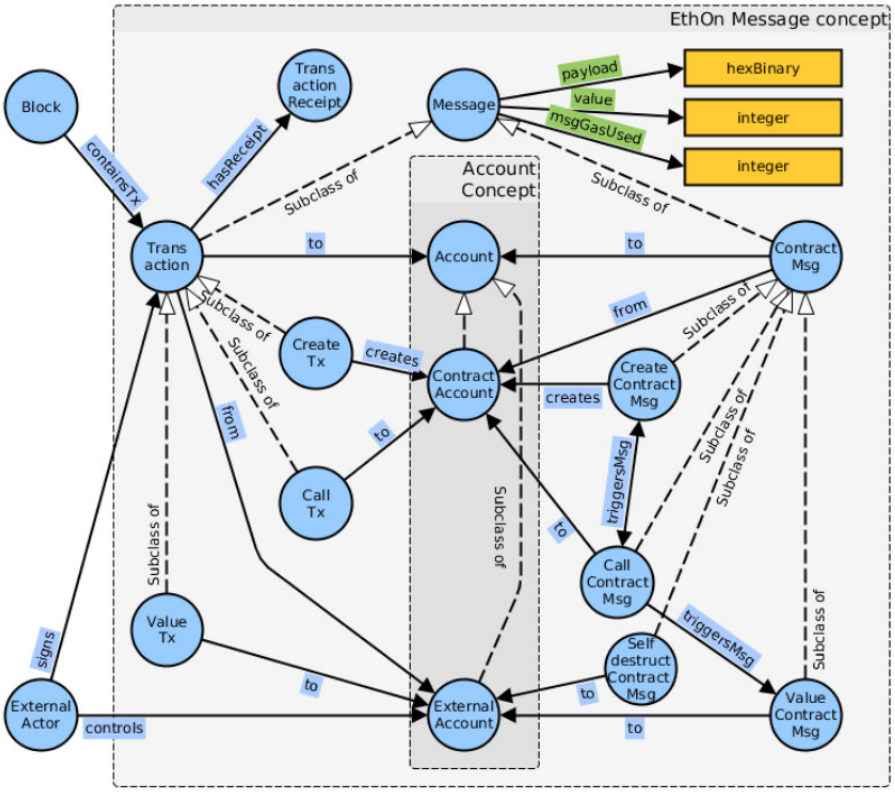

Figure 2.15: EthOn message concept

2.4.8 Smenatic Smart Contract

2.5 Web 3.0

o

2.6 TemporalGraphsontheEthereumDistributedLedg

The Ethereum blockchain is not suited to storing large amounts of data thespeed of execution is unlikely to be fast enough to support high volume datastreams. However, in order to achieve the goal of validation of data integrity, itis not necessary to store the data itself on the blockchain; all we need is to storesufficient metadata to allow

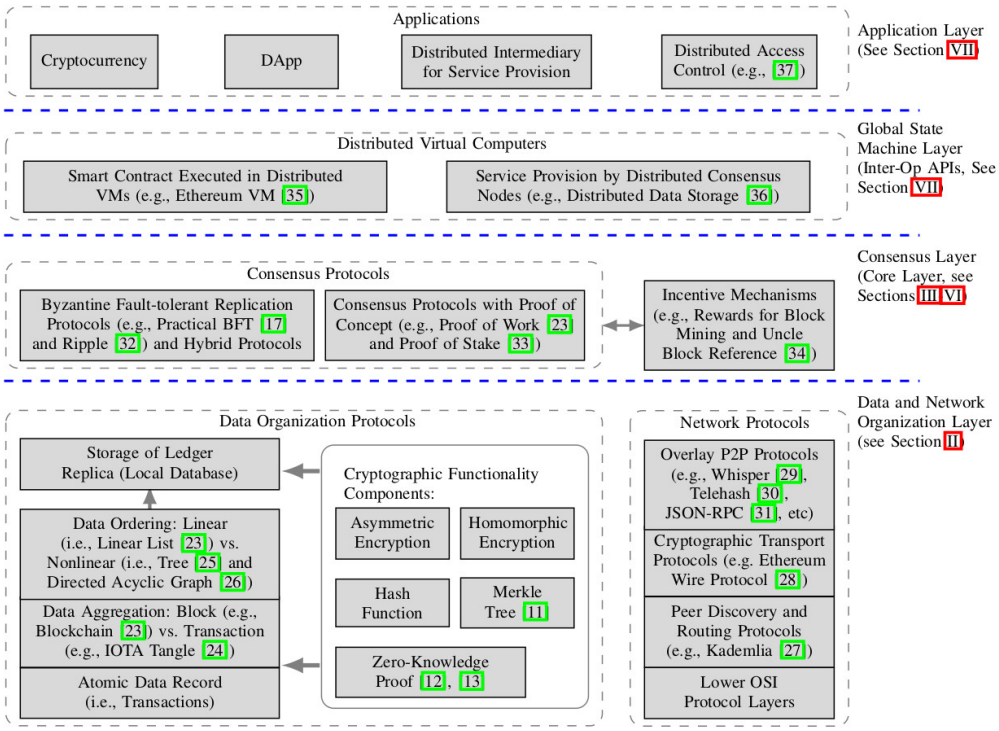

Figure 2.16: Blockchain technology stack

anyone who does possess a chunk of the data to verifythat its contents are intact. We need, therefore, a canonical representation of thedata which can be hashed reproducibly to provide a verification for example,the RDF serialisation of the source and a reliable form of punctuation in thedata stream, to identify complete chunks of data to be used for the hash.The form of punctuation used depends on the temporal model used in thedata. Multiple approaches have been taken to the representation of temporaRDF streams. Broadly, they vary as to whether temporal information is associ-ated with each triple individually (triple-based) or with RDF graphs, wherethe triples in an individual graph share the same temporal information. In thelatter approach, a graph usually corresponds to a timeinterval. In the former,the temporal information attached to a triple may either be an interval or atimestamp representing an instant. [3,6,7,10] The difference between intervaland timestamp is not deeply relevant; with an interval representation, one canalways simulate a timestamp by setting the start and end points of the intervalto be identical, with no loss of information. For the purposes of this work, giventhe requirement to group sections of the data stream in order to implement whatis needed for verification, it seems most appropriate to use

the graph approach.We therefore ensure that the data streams generated from sensors are seg-mented, at source, into named graphs corresponding to time intervals, with thelength of intervals to be determined also at the source, and indicated within thedata itself. Variable rather than static intervals provide more flexibility.

Smart contracts running on (a private instance of) the Ethereum blockchainhave been written to receive the data, with each remote client provided with theaddress of the relevant contract(s). Each time data is sent, the contract identifies graphs specified by the source, and calculates a verification hash, extracting the start and end times of the interval covered by each graph. Four items graphURI, hash, start and end time are stored in the state of a new smart contract,the address of which is stored in a master contract and which, along with the original data, is forwarded onto a traditional RDF store.At the same time, clients performing event detection construct an RDF repre-sentation of each event detected, and send it to a separate smart contract, alongwith the names and hashes of the relevant temporal graphs. The duplication ofhashes provides a separate source of validity information for each graph.In order to support the verification of data in standard SPARQL queryingscenarios, a custom SPARQL endpoint, running off-blockchain, is being writtento respond to federated SPARQL requests using the SERVICE keyword. Anytriple patterns passed to this endpoint specified to be in a temporal graph knownto be hashed on the blockchain are queried from the full dataset, and eachrelevant graph is hashed, and compared to the entries stored in the blockchainboth from the streaming data contracts, and any relevant contracts from eventdetection. The custom endpoint returns a triple stating whether verificationsucceeded, allowing at least a base level of verification within SPARQL[rdfgraph]. TODO

2.6.1 Graph Blockchain Definition[p1171]

Let us start with introducing a few definitions related to the RDFgraphs.AnRDF triple6contains three components:the subject, which isan RDF URI reference or a blank node,the predicate, which is an RDF URI reference andthe object, which is an RDF URI reference, aliteral or a blank node. AnRDF graphis an unordered set of RDFtriples and anamed RDF graphis an RDF graph which is assigned aname in the form of a URI [5].The main idea behind GraphChain is to use Blockchain mechanisms on top of an abstract RDF graph data type. Following the general definition of the abstract data type (ADT) [6], we definethe abstract RDF graph data

type by its behaviour in computersystems, the set of possible values and the operations permitted onthese values. Such an approach allows us to create a Blockchain-compliant system on top ofanyconcrete data structure used torepresent and expose the graph. In general GraphChain does notassume any specific serialisation of the RDF graph (like RDF/XML,Turtle or JSON-LD) nor a mechanism for storage (triple store, files,memory) or exposing the graph (HTTP, file read etc). GraphChainis thus defined as:(1)A linked chain of named RDF graphs specified by the Graph-Chain ontology.(2)A set of general mechanisms for calculating a digest of thenamed RDF graph.(3)A set of network mechanisms that are responsible for thedistribution of the named RDF graphs among the distributedpeers.The following two definitions make the idea of GraphChain moreprecise:Definition 2.1 (Linked chain of named RDF graphs).A linkedchain of named RDF graphs is a linearly ordered collection (a chain)of cryptographically secured named RDF graphs. The collectionis ordered by a functional relation pointing to the previous graphin the chain. With exception to the first graph in the chain, therelation is asymmetric and irreflexive.Definition 2.2 (GraphChain).A linked chain of named RDF graphsstarting from the first graph called genesis unit7is called the Graph-Chain. The GraphChain acts as a consistent named graphs historyon which all nodes eventually agree8.[p1171]

2.6.2 Gragh chain Arcitecture[p1171]

A single node consists of several parts: a web interface for communi-cation with clients (via the HTTP protocol), a web socket endpointfor communication with others nodes, a cryptography module forhandling of digest calculation, a triple store repository managerfor storing blocks as sets of triples and obtaining blocks from therepository, and services which bind all these parts together.In figure 1 a diagram presenting nodes interaction is depicted.Clients can connect to a node via HTTP and to a triple store viaSPARQL over HTTP. The interaction with the node itself enables aclient to create new blocks (a process of block creation is explainedbelow), add new peers, and retrieve block meta-information. Inter-action with the triple store enables reading data from it with theusage of the SPARQL query language. Interaction among nodesis implemented by means of the web sockets protocol. Nodes caninteract with their triple stores by various ways. These ways canbe determined for example on the basis of required performance.For now we use SPARQL over HTTP.The block creation process begins with an HTTP request fromthe client. The HTTP request contains a serialised RDF graph (seefigure 2). For now, we assume that this graph is serialised in theTurtle format, but it should be fairly easy to extend the application so that it would accept

other types of serialisation formats. A sampleHTTP request is presented in the listing 1. For now we have notdecided to limit the size of the input RDF graph, but we will do thatin the future works. After the application receives the graph, it is deserialised andits hash is calculated with one of the methods we will discuss insection 4.3. Then a hash of a string being the concatenation ofthe following elements: a new block index, a previous block IRI, aprevious hash, a current time stamp, a received graph IRI, and theaforementioned graph hash, is computed. The hash is consideredto be the hash of the block.After the hash for the new block is calculated, a new block objectis created. It consists of the counterpart Blockchains header andcontent (see section 5). The block is then serialised as two sets oftriples: one for the header and one for the content. The first set of triples is stored in agraph ledger(which is a named graph in arepository) and the second one as a separate named graph. Finally,information about the change is broadcast to the other nodes.[p1171]

2.6.3 RDF Graf

TODo

2.6.4 mporalGraphsontheEthereumDistributedLedg

The Ethereum blockchain is not suited to storing large amounts of data thespeed of execution is unlikely to be fast enough to support high volume datastreams. However, in order to achieve the goal of validation of data integrity, itis not necessary to store the data itself on the blockchain; all we need is to storesufficient metadata to allow anyone who does possess a chunk of the data to verifythat its contents are intact. We need, therefore, a canonical representation of thedata which can be hashed reproducibly to provide a verification for example,the RDF serialisation of the source and a reliable form of punctuation in thedata stream, to identify complete chunks of data to be used for the hash.The form of punctuation used depends on the temporal model used in thedata. Multiple approaches have been taken to the representation of temporal RDF streams. Broadly, they vary as to whether temporal information is associ-ated with each triple individually (triple-based) or with RDF graphs, wherethe triples in an individual graph share the same temporal information. In thelatter approach, a graph usually corresponds to a timeinterval. In the former,the temporal information attached to a triple may either be an interval or atimestamp representing an instant. [3,6,7,10] The difference between intervaland timestamp is not deeply relevant; with an interval representation, one canalways simulate a timestamp by setting

the start and end points of the intervalto be identical, with no loss of information. For the purposes of this work, giventhe requirement to group sections of the data stream in order to implement whatis needed for verification, it seems most appropriate to use the graph approach.We therefore ensure that the data streams generated from sensors are seg-mented, at source, into named graphs corresponding to time intervals, with thelength of intervals to be determined also at the source, and indicated within thedata itself. Variable rather than static intervals provide more flexibility.Smart contracts running on (a private instance of) the Ethereum blockchainhave been written to receive the data, with each remote client provided with theaddress of the relevant contract(s). Each time data is sent, the contract identifiesgraphs specified by the source, and calculates a verification hash, extracting thestart and end times of the interval covered by each graph. Four items graphURI, hash, start and end time are stored in the state of a new smart contract,the address of which is stored in a master contract and which, along with theoriginal data, is forwarded onto a traditional RDF store.At the same time, clients performing event detection construct an RDF repre-sentation of each event detected, and send it to a separate smart contract, alongwith the names and hashes of the relevant temporal graphs. The duplication ofhashes provides a separate source of validity information for each graph.In order to support the verification of data in standard SPARQL queryingscenarios, a custom SPARQL endpoint, running off-blockchain, is being writtento respond to federated SPARQL requests using the SERVICE keyword. Anytriple patterns passed to this endpoint specified to be in a temporal graph knownto be hashed on the blockchain are queried from the full dataset, and eachrelevant graph is hashed, and compared to the entries stored in the blockchainboth from the streaming data contracts, and any relevant contracts from eventdetection. The custom endpoint returns a triple stating whether verificationsucceeded, allowing at least a base level of verification within SPARQL.[Ethgrapf.pdf]

2.6.5 Knowledge Graph

One of the cornerstones of the Semantic Web is the ability to represent data semantically such that the meaning of a piece of data can be read, by human or machine, from the data itself and not from its position in a structure such as a relational table. The most commonly used standard is the Resource Description Framework (RDF) [26], in which data are represented as triples semantically, predicate objec subject predicate ob- ject" sentences where graphs serve to group triples. In both cases, subject and predicate (and graph, where present) are URIs, and objects may be either

URIs or (optionally typed) literal values, e.g., numbers or strings. RDF has a number of concrete representation formats such as RDF/XML [6], Turtle [7], NTriples [5] and NQuads [10], JSON-LD [18], and so on, but each implement the same semantic model. Querying of Linked Data is designed around the idea that data from multiple sources can be linked easily no matter where it comes from, and without the need for coordination among data publishers beyond the use of common vocabularies and ontologies for common concepts. This model allows simple data integration and on-the- y construction of relevant knowledge graphs from multiple sources by querying. The dominant standard for querying Linked Data is SPARQL [25]. A data source publishes a SPARQL endpoint, an HTTP-accessible interface which responds to queries in the SPARQL language, a query language which is somewhat similar to SQL [13], with SELECT, DELETE and INSERT operations (among others), and a query pattern syntax adapted to the RDF data model. It sup- ports sophisticated

ltering and aggregation, and, crucially, via the SERVICE keyword, the ability to pass subqueries to other SPARQL endpoints to achieve federated querying that is, querying of multiple data sources at once. The downside of SPARQL is its comparative weight on the server side, and its complexity. Many of the advanced

ltering and aggregation features can be computationally expensive for the endpoint server executing them, and the im- plementation of SERVICE-based federated querying requires all communication between other data sources to be carried out by the initially targeted endpoint. To address these issues, the Linked Data Fragments (LDF) approach to querying has recently been proposed [1]. A data source for an LDF query may be a SPARQL endpoint, but can be any source which provides valid RDF. Cur- rently the only query patterns supported are Triple Pattern Fragments (TPF), although there is scope for extension. The basic query structure it handles is the Triple Pattern, which is an RDF triple subject predicate object but where one or more terms are replaced by variables in the form of alphanumeric strings beginning with a question mark. So where [53]

Chapter 3

my idea , Usecase

Chapter 4

Conclusion

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Appendix A

Sample Title

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Appendix B

Sample Title

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Bibliography

[1] Albert Einstein. Zur Elektrodynamik bewegter Körper. (German) [On the electrodynamics of moving bodies]. *Annalen der Physik*, 322(10):891–921, 1905.