Semantic Smart Contracts And Their Integration With Semantic Data Licesing

Zahra Jafari
University of Innsbruck

A thesis submitted for the degree of
Master of Informatic

2019

This page is intentionally left blank.

This page is intentionally left blank.

Abstract

Smart contracts as computer code which reside on blockchain technologies are receiving great attention in new business application because they allow parties to represent contract terms in program code and thus eliminate the need for a trusted third party. The creation process of writing valid contracts in Ethereum is difficult task. Blockchain as distributed ledger technology is increasingly used as transnational data storage between parties and it gains good popularity among new industries in last few years. Blockchain implemented in different areas of applications such as social, healthcare, logistic and etc. It is also capable to execute smart contract and prevents data tampering by validating transaction through consensus protocol. Research on this topic is still on early stage in science. Based on an analysis of collected data and our works, we divide this research into 3 sections: first we focused on how smart contract build up on blockchain. Second, we demonstrate how blockchain can integrate with semantic web technology. It is focused on some solutions for indexing and executing smart contract on Ethereum blockchain. And third section, It is focused on use case: supply chain management system and how smart contract facilitates supply chain process and represented different models of ontology in this field, then implemented Etherum ontology concepts to model in semantic blockchain using RDF triples and SPRQL query.

Contents

List of Figures

0.1 Introduction

Smart contract is computer program that expressed the content of agreements and preform transaction on blockchain when specific conditions are met. Smart contract preform verified transaction on blockhain without third party or any supervisors, thus ensuring us to transparent, valid and secure transaction. As blockain is distributed ledger which store data and transactions, querying them becomes challenges task. This due to the fact that blockchian can allow transaction and payment without needing intermediary. Moreover there is need to integrate blockchin with semantic web service, thus making use of some linked data tools to index blocks and transaction according to Ethereum ontology.

In this paper, supply chain management is regarded as use case where blokchain is fit for some reasons. During product life cycle in every step, data can be documented in blockchain. Blockchain technology can contribute to record single asset as it flows through supply chain node, track orders, payment, product and track digital asset. Blockchain can contribute through distributed nature in sharing information about product process, delivery between from supplier to customers. In today's world, supply chain is complicated structure with multiple involved participants with amount of activities. Security and organizational issue cause to improve the need to build blockchain based supply chain management. In spite of some features of supply chain, blockchain also offers some advantages by indexing, registering products, increase transparency and trust of participants. Besides elimination of third party which allow for growing number of participants, it increases innovation by deploying smart contract with low fee transactions, without cost of third party.

The literature provides some supply chain management ontologies for range of activities and industries. Many studies claim the benefit of ontology in supply chain industries which are suggested multiple models to supply chain ontology another application of these ontologies by some Enterprise Modeler program model ontology. This reports different ontologies models in supply chain management fields. Then it works on how semantic web service can be applicable in this field. Existing supply chain ontology has several gaps with respect to domain accuracy, consistency and development approach. But we attempted to semantify blockchain using ontologies and some semantic techniques.Then we purposed proof of concept developed in the concept of supply chain

management and semantic blockchain.

As this is important to gain advantages, we used Ethereum ontology which provide some specifications of phenomenon, adapt semantic principles for developing semantic blockchain and mapped these concepts using RDF triple and model based on Sparql query to produce blockchain models based on semantic web ontologies.

Chapter 1

Smart Contract and Distributed Ledger Technology

1.1 Ledger

Ledger is a principle book or computer file for recording transactions measured in terms of unit account with specific balance for each account[22].

1.1.1 Distributed Vs. Undistributed

the difference between decentralized and distributed is made by Baran(1964). decentralized means there is no single point to make decision. It contains of central hierarchy nodes. Each node make up decision for own self and system gather all responses as resulting behavior. In distributed system, process spread across all nodes and there are no central nodes. The main difference is that decentralized database is the collection of independent databases, while distributed is the single database which is spread multiple inter-connected computers on different locations. *Ozsu and Valduriez* define distributed database as a "collection of multiple, logically interrelated databases distributed over a computer network and distributed database makes a transparent distribution to all users"[27]. Based on this definition Blockchain technology adheres to both definitions, as it appears as single system to its users and perform an task in a network. Thus, Blockchain is a form of a distributed computing system.[22].

1.2 Distributed Ledger Technology(DLT)

DLT is the method of keeping a distributed ledger on networks of a computer. DLT uses a consensus technique for adding a new node(participant) over the network. Similarly, This is the digital record that shared across participant and the records held by each node or participant. The term 'blockchain' is the most well-known type of DLT and are used by many best-known instances of DLT.
A distributed ledger can be either permissionless or with permission, regarding be private or public. Most of the distributed ledger is permissionless and public means anyone can easily join to network and see all entries. But in financial fields, distributed ledgers are restricted to members of each participant and they are 'private' networks. Furthermore, in financial fields exist also commercial sensitivity about the privacy of data related to each participant. In another word, no one wants that data to be seen by the other participant. therefore, participants can see their data and transaction on the network.
Recently, blockchain as a type of distributed ledger become the most popular and widely used in diverse fields. ledgers ?????such as Ethereum extend initial bitcoin blockchain with features such as smart contracts that enable to distribute of data on the blockchain securely. Also, there is an increasing demand to integrate data stored in distributed ledger with other external sources. And also integrate smart contracts with the service available on the web. This is where linked data comes to play to provide sufficient access to data and smart contracts also stored on Ehereum blockchain via semantic web technology stack[3].

1.3 Blockchain

Blockchains are the distributed records for digital events and they are structured as a chain of linked data stored in individual database or computer over network[3]. Blockchain is organized into blocks. Each block is identified by cryptography hash that refers to a hash of the previous block. this creates links between blocks. Thus, it creates the chain of blocks wherein Blocks have held a copy of the blockchain structure. The initial block created manually, is known as *genesis block* and the other node will add to the blockchain after a process of consensus between nodes. The distributed consensus method allows adding a new block or item into the blockchain that is verified as legitimate. This process would be done by some computational work called Proof Of Work

(POW) or mining. All blocks in blockchain hold a small amount of data which need to be secure before distributing over all participating computer over the network and are visible by having the public key for all participant but not modifiable. These data get timestamp to provide the time of adding that block.

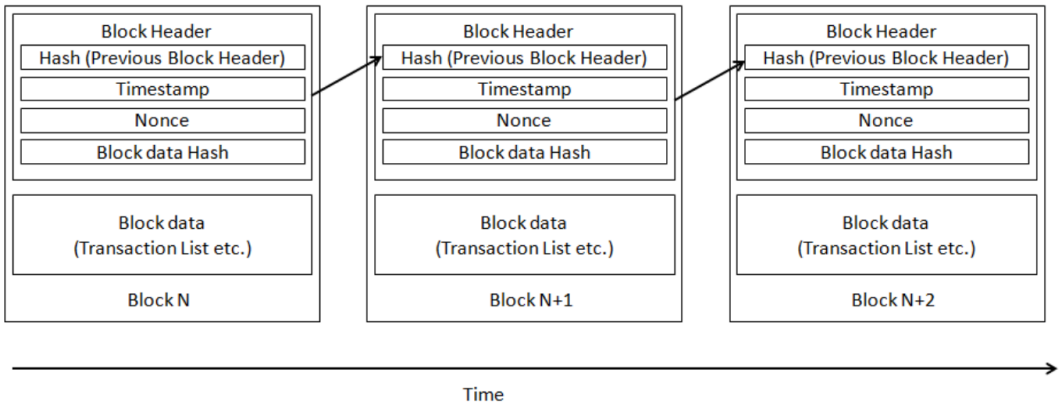

Figure 1.1: Generic chain of blocks

Blockchain has four main concepts:

P2P network: nodes can interact with each other using the private and public key. Private keys are transactions signature and public keys are address of transaction that can be reachable on the network.

Open distributed ledger: It is like a book containing all transactions that each node has a copy of data and there is no central entity and anyone can see assets and how much assets has each node. Thus, they can decide about the validity of transactions.

Synchronization: As all node has one copy of the ledger. Therefore, synchronization should be done throughout networks whenever new item or one new transaction added to network and finally consensus will need to validate these transaction.

Mining: In the distributed ledger, all nodes can not receive transaction simultaneously. In order to add a new item in the blockchain, the consensus of all nodes are needed and it prevents every node to add a transaction into the blockchain. Miners(buyer) are whom attempt to validate the transaction to add to the blockchain.

1.3.1 Advantage Of Blockchain

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 to the transaction.
- Storing whole records overall networks of participating computers mitigate the probability of losing infrastructure. - One new item is confirmed and added to blockchain, can not be altered. - Events are stored and accessible to any participant but required public key to be reachable[29].
- Blockchain is the sole technology that fulfills such properties:*(a)trustless*: There is no need to verify involved identities in a transaction. *(ii) Permissionless*: there is neither permission nor controlling for participants in the network. *(iii)censorship resistant*: Anyone can trade on blockchain and just cryptography algorithm governs the operation that entities(participants) can trust it. this feature and above reason make the blockchain as powerful and most secure technology in commercial events over network[2].

1.3.2 Limitation of Blockchain

There are some limitations and need to cope with before blockchain can be more applicable: the proof of work which generate blocks are wasteful. Although, it helps us to remain integrity in blockchian and prevent form attacks, but as blockchain get longer this problem deteriorates.
Also, when blockchain grows,processing speed affects adversely. Because it needs more time to verify transactions. However, blockchain rlies on consensus of majority to ensure security in blockchain. there is probability of being violated this security. Requiring the consensus cause some malicious attackers can tamper the blcockhian by achieving %51 of consensus and introduce their version into blockchain and validate it. In order to overcome this limitations, proof of stack(PoS) used over proof of work. In POS, miners are not compulsory to solve computational puzzle. Instead, they would be selected based on their wealth or stack in cryptotoken. This will decrease the probability of a %51 attack and wasting the resources.

1.3.3 Type of Blockchian

Permissionless/ public Blockchain This blockchain allows anyone to join to network and create consensuses such as Bitcoin and Ethereum. In a permissionless blockchain, any miner can create consensus mechanisms such as proof of work, proof of stack to validate the transaction. but as this mechanism is decentralized, it has low rate of validity function[30].

Permissioned/ Private Blockchain: This blockchain, the only restricted participant has the right to validate the transaction. Therefore, it provides better privacy and scaleability. Unlike permissionless blockchain, this blockchain does not have mining computation to reach the consensus because all participants are known in this network[30].
Wallet contains the ether which will be used to execute transactions[35].

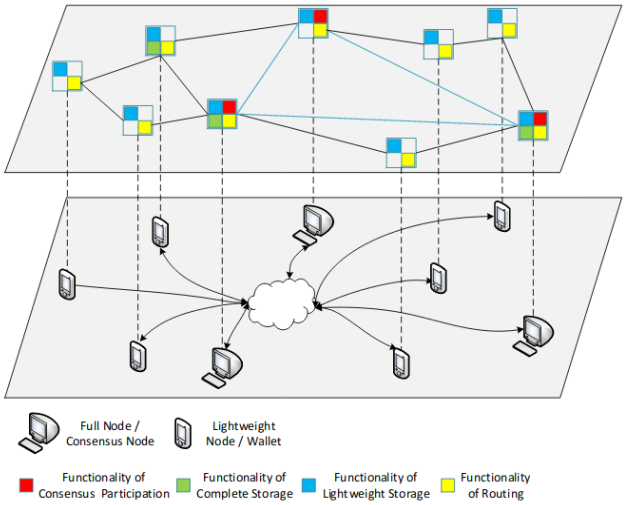

Figure 1.2: Permissionless blockchain network. The P2P links between consensus nodes are shown in blue.][60]?????

1.3.4 Mining

It is a process of computation on the blockchain to verify and add block. Miner add new block and others check the validity of new block. Any participant can take part

in mining pool, But chance of finding valid depends on power of computer to preform calculations. Sometimes a miner will find an uncle block; an uncle block is a block which is initially valid but is surpassed by another, faster block. Uncle block is rewarded with $\frac{7}{8}$ of full block value and hash will be added to valid block. A max of two uncle blocks can add to valid block and the miner of valid block also receive $\frac{1}{32}$ extra ether for each uncle block[35].

1.3.5 Mining Pool

Mining can be done alone or in mining pool. Mining pool is better way to solve block and get rewards as compared to mining alone.Miner in pool mine together and rewards will split to all members in pool[35].

1.3.6 Ether

Is the form of payment and as fuel for Ethereum. The base fo mining(find the solution and add block) successfully mining block is five ether. If miner find solution but not fast, it becomes less ether like 4.375 ether and will be uncle block. Each block can contains just two uncle block and receive $\frac{1}{32}$ per uncle block. If another miner also find solution. this block can not be added into blockchain and miner just receives 2-3 ether[35].

1.3.7 Merkler Tree

all transactions are stored in tree structure wherein leaves contain transactions and the internal node contains the hash of its sub tree and a single root also contains the hash of two children and represents the top of the tree. The purpose of bottom-up hashing is that if an attacker attempts to create fake transactions into the bottom of the 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[Buterin].

Merkler tree helps the blockcchain by providing tree-structure of transactions. The structure can be described the branches b and main branch is root $b_1...n$ and is connected by

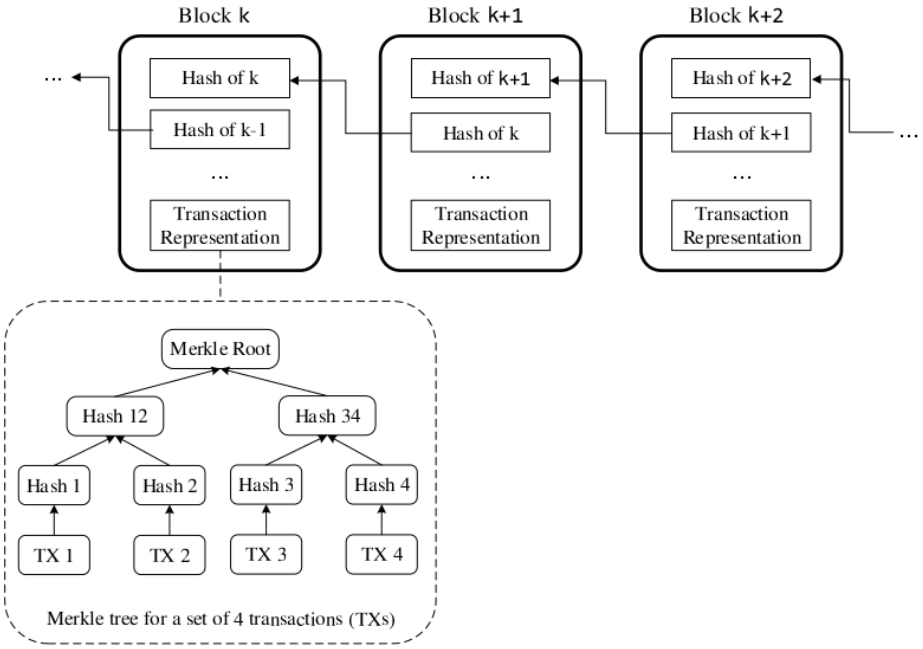

Figure 1.3: Illustration of chain of blocks and markler tree in single block [60]

its concatenations hash $b_{1..\frac{n}{2}}, b_{\frac{n}{2}+1...n}$, this is:

$$b_{1,...,n} = h(b_{1,...,\frac{n}{2}} + b_{\frac{n}{2}+1,...,n})$$

And branches are :

$$(b_{1,...,\frac{n}{2}}, b_{\frac{n}{2}+1,...,n}) = (h(b_{1,...,\frac{n}{4}} + b_{\frac{n}{4}+1,...,\frac{n}{2}}), h(b_{\frac{n}{2}+1,...,\frac{3n}{4}} + b_{\frac{3n}{4}+1,...,n}))$$

Then , Binary branching continue down:

$$b_1 = h(T_1), ..., b_n = h(T_n)$$

Where T is the transaction in blockchain. Generally, it takes $\log_2(n)$ branches to check transaction is $h(T) \in b_1, ..., n$[7].

1.3.8 Hash Function

It is a mathematical method to apply cryptographic (secret writing) function. A hash function is deterministic means for input to produce the same output each time. Altering

data generate different output. According to Wikipedia, the security level of the hash function has different properties:

- Pre-image resistance: It is 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.
- Collision Resistance : It is difficult to find two different input (m_1, m_2) that have same output in way that$hash(m_1) = hash(m_2)$.

The hash function used in blockchain and it is the most secure hashing algorithm is SHA256 which that provides the combination for a given input with 256-bit length. Using SHA256 in blockchain makes to be impossible to duplicate hash because there is a diverse combination of this input with this length and it requires a huge amount of computational works. That means there are 2^{256} hash values[36].

Input value	SHA256, message hash
1	6b86b273ff34fce19d6b804eff5a3f5747ada4eaa22f1d49c01e52ddb7875b4b
2	d4735e3a265e16eee03f59718b9b5d03019c07d8b6c51f90da3a666eec13ab35
Blockchain	625da44e4eaf58d61cf048d168aa6f5e492dea166d8bb54ec06c30de07db57e1

1.3.9 Blocks

Blockchain consist of blocks to record series of transactions. Blocks are like container that every on see metadata but only a owner has key. Each block contains a list of transactions which once the block is accepted by the network, by some consensus algorithm, will be broadcasted to the network and included in the blockchain[**?**].

1.3.10 Header

It is like metadata on the top of the block which includes which includes a reference to the previous block, timestamp containing the time of creation, Merkle root which is an efficient data verification structure, nonce[7].

1.3.11 Nonce

it stands for *number used only once*. It is integer number and along with block data, number and previous hash use as input in SHA256 function to generate the current block hash. We used this nonce to vary the hash of the current block and without modifying the data inside the block. By the usage of the nonce, a miner can generate a valid hash, adding first own block into blockchain and get the reward[36].

1.3.12 Merkler Root

All transactions contained in the block can be hashed together and create a one-line hash text which is the concatenation of all transactions in block. This is the root hash of the Merkler tree that gives a short representation of each transaction in a block????.

1.3.13 Transaction

transfer data (asset) between users, an transaction contains information such:
- The recipients which message to send
- Signature of sender
- A mount of ether to transfer
- Structure value, the maximum number of computational step that transaction is allowed to do
- Gas price, fees that should payâĂę[35].

1.3.14 Consensus Algorithm

Consensus algorithm is used to get agreement of all participants about what block appends to the chain. By the use of this algorithm, real participant and malicious can be recognized. A blockchain is consensus, if block b_t is before blockb_{t+1} while no other participant can append block in reverse order. In this section, i will go through some important consensus algorithm which is better suited for blockchain[7]:

1.3.14.1 Proof Of Stack

It is an alternative to proof-of-work that fewer CPU computation for mining. In proof of stack, the chance of mining the next block chances of a node mining the next block depends on node balance. In private networks however, where the participants are known,

costly consensus mechanisms such as proof-of work are not required. This prtically removes need for mining and give wide ranges of consensus protocol for picking node[17].

1.3.14.2 Proof Of Work(POW)

It is mechanism that ensure consensus is done without any central control. With POW miners compete to complete its transaction first into blockchain and get rewards(e.g: Bitcoin, Ether).
Miners connected to blockchain and accomplish task validating transactions to add new block by solving cryptographic puzzle and any body who complete own task sooner, can add own block first in blockchain[28].

1.4 Bitcoin

The basic goal of blockchain technology is to ensure people form trustworthy and legitimacy of transactions. Blockchain initially used to enhance commercial transactions through a currency called Bitcoin.
Bitcoin is the digital records or cryptocurrency that is accepted by users involved in the transaction. Bitcoin is the financial use case of this powerful technology[2]. Bitcoin is the list of blocks of transactions. Each block in the blockchain is identified by the hash algorithm on the top of the block. Bitcoin is introduced by a consensus mechanism of blockchain, it is a well-known implementation of decentralized cryptocurrency.
Bitcoin is the limit of the block, wherein each block is verified by a hash algorithm using SHA256 cryptography on the top of the block. Each block encompasses the hash of its parent (previous block) in the own header that refers to it. It forms a blocklist wherein each block refers to the previous block in the list name genesis block. Altering in block implies to create a new block on the top. Since each block contains the hash of its parent creating a new block is expensive and needs proof of work that the miners allow the add new block[28].

1.5 Ethereum

It is another cryptocurrency similar to Bitcoin that built on the top of the blockchain. the participant publishes the transaction on the network that is then divided into a node (called miner) and add to the blockchain using consensus mechanism. The state of the

system refers to the sate of account that can be an external account related to the user of the system(that contains information about balance) or contract account that obtain contract code or constant storage of that account. The virtual currency in this system is *Ether*. The transaction can change the state of the system by creating a new contract or invoking an existing contract. calling the external account just transfer the Ether but calling the contract account to execute the code of that contract and may perform a transaction or change the storage of that account[14].

1.5.1 Gas

Transaction in Etheruem platform needs fuel to execute called gas which is used internal and paid in advance to execute a transaction. If transaction get run of gas, means transaction is executed. If transaction rolled back but consumed gas will not be returned.
To enable easier calculations, Ether also has some sub-denominations[35]:

- Wei - 10^0
- Szabo - 10^12
- Finney - 10^15
- Ether - 10^18

1.5.2 Account

There are two types of accounts in Ethereum:
- *Normally controlled* is an account controlled by private key. if Person has private key can send message and ether from this account.
- *Contract* is an account controlled by code. It is an normal account with extra option of containing code. Ethereum blockchain starts firing transaction from an account which this transaction is the response of receiving transactions by an account[35].

1.5.3 Message

As already said blockchain fire transaction when receive a transaction. When an account send transaction mean send a message. Message contains all attribute the same as transaction, but $gasPrice$. the only difference between message and transaction is that message is fired by contract[35].

1.5.4 Ethereumt Virtual Machine(EVM)

It handels the state of contracts and it build on stack based language with some instruction like opcode. Actually, contract is collection of opcode statteents which are executed on EVM. EVM can be assumed as decentrelaized computer which all smart contract run. It is network of smaller interconnected machine but sounds to be a big computer. Ttransactions executed smaert contrat on each node on the network and each node gather transactions sent from userts to block in otder to append in blockchain to collect rewards.
To ensure resource handling of the EVM, every instruction the EVM executes costs gas. Operations with more computational resources cost more gas, than operations with less computational resources. Therefore, the using gas is beneficial due to encourages developers to write quality applications and avoiding wasteful code. When it comes to paying for gas, a transaction fee is in small amounts of Ether,and the token with which miners are rewarded for executing transactions and producing blocks[37].

1.5.5 Solidity

os high level truing complete language with Java script similar syntax. Contract is similar to classes in object oriented language. Contract contains the fields as persistent storage of contract and methods to be invoked by internal and external transactions. For interacting with another contract, either need to create new instance of this contract or make transaction to known contract address.
In particular, Solidity provides for accessing the transaction and block information like: *msg:sender* or *msg:value* to access the amount of *wei* transferred by transaction that invoke the method. Solidity uses some fuctions to rransfer money ro another contracts suhc as *call* and *send*. A value trnsfer using this function to internal call transaction which implies calling contract also execute code or may fail to execute due to insufficient gas. Another contract is *fallback* function that get execute via *call* and *send* function was preformed [14].

1.6 smart contract

A smart contract in computer science is the piece of code which is designed to execute certain terms by predefined condition. These terms are embedded and preformed on a

distributed ledger. As compared to a traditional contract that includes the third party to execute terms of condition, the smart contract has low transaction fees and is more profitable without needing to third party.

A smart contract is an self-executed and self-enforced agreement.

Sometimes, smart contracts and DLT are remembered as the same thing, but not. They are different technologies that are complementary to each other. However, there are several platforms in DLT on which smart contracts can execute. By existing computer and capability of executing code, why have not smart contracts developed?

This question shows the smart contract and DLT and the relationship between each other. A computer is capable of executing an event such as payment of contact when pre-defined conditions met. But, that would have meant that both parties requires to have programmed on own computer. subsequently, the version of codes or using program may differ that would be another issue.

What DLT has done, is that to bind the parties to each other by embedded code in a distributed ledger. More importantly, DLT ensures both parties to have secure transactions and the contract will execute automatically. This is the exact description of smart contracts as 'self-executed' and 'self-enforced'. With the advent of distributed ledger, needing an efficient query of diverse data and indexing entries become more important. Within the blockchain, a smart contract is actually coded inside the block and it invokes by receiving a transaction or message and send the transaction in the response of transaction that has received then it can read, write or create the contract.

A smart contract is an independent factor a behalf of the user to perform some operations 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 [35].

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* uses the smart contract that defines three functions: *'deposit'*: store unit of X into a contract, *'trade'* send back one unit if X instead of five units of Y and *'withdraw'* to roll back all asset into the contract. *Bob* starts activating smart contract by calling the *'deposit'* function and moving 3 unit of X asset into a 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

signature and *'withdraw'* is called by owner, then transfer all deposits 1 unit of x and 10 unit of Y to Bob[17].

There are some features of smart contact out of this example:

- Contract has its states and control over them. These states can be updated by executing the contract. The blockchain keeps a record of all previous states of a contract because overwriting the previous state would involve overwriting records earlier in the blockchain.
- Contract allows us to do business logic in code.
- Contract is deterministic means for the same input produce the same output.
- Smart contracts can be used to implement dynamic data storage.
- Correct smart contract describes all possible results.
- Data explains Relationship between participants.
- Smart contract is triggered by receiving a transaction and sent the transaction afterward.
- Since contract stores on the blockchain, that can be visible by every participant on the network.
- Each participant can trace contract operation due to *sign* message using cryptographically verification.

1.6.1 Security in Smart contract

A survey on the smart contract shows that the Bitcoin and Ethereum focused mostly on financial contracts. That's why is essential that the smart contract execute and perform correctly. A security issue in such a structure is to be used for a malicious purpose such as DAO hack or some incidents in Bitcoin. Such incidents caused a hard fork of blockchain to mitigate the malicious transaction. Atzei et al. list 12 vulnerabilities that are assigned by Solidity, EVM and blockchain. Such vulnerabilities can be addressed by secure smart contracts[37].

1.6.2 Vulnerabilities in Ethereum Smart Contracts

Atzei et al. in this section categorized the vulnerabilities in smart contracts into three classes(solidity, Ethereum virtual machine, blockchain:

Level	Vulnerabilities
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.

- Exception disorder: this has occurred in a situation such as execution run out of gas, call stack limit and *throw* command.

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* fallback function. However the upper bound of unit of gas is 2300 that is available to *callee* and is executable.

Type casts compiler in solidity may faced to some errors such as assigning integer to type string

Reentrancy the nature of transaction cause the programmer to believe that invoking non-recursive function can not *re-enter* before termination. But it is not always true, because callback function may allow the attacker to *re-enter* caller function which cause the unexpected behavior or invocation loops and consume all gases.

Keeping secrets Fields of contract can be public and private. However, using private field can not guarantee its secrecy. Because executing contract will publish transactions on blockchain. using suitable cryptographic techniques can remove this problem.

Immutable bugs AS contract publish on blockchain, that can not be changed any more, if there is error on contract, it can not be patch it. there is no way to predict the error of contract and terminate implementation.

Ether lost in transfer when the ether associated to any user or address. ether will lose forever. programmer must be aure to send ether to valid user address.

Stack size limit when ever contract call another contract *call stack* increases one frame. *call catack* is bounded 1204 frames: when the stack exceed this limit size. Further invocation will give an exception.

Unpredictable state when user send transaction on blockchain to call smart contract. user can not be sure if the contract transaction time and transactions received by user are at the same state. Because other transaction have changed the contract state.

Generating randomness many contract generate random number where all miners chose unique initialization seed.

Time constraints or timestamp is used to determine which actions are mandatory in the state.

Chapter 2

Blockchain as the infrastructure of semantic web

2.1 Distributed ledgers and indexing

A distributed ledger based on a blockchain does 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 the possibility of to control automatically what happens with cryptocurrency on the blockchain without involving the untrusted external sources. Ethereum smart contracts have an account that can normally store, update or make a function with the input and output. The concept*accounts* are widely used in this study refer to an agent such as human and the concept *balance* refers to cryptocurrency 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 the 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 the next step. It indexes basic entities such as account, blocks related to distributed level and data can be stored or retrieved here. In the functional level, smart contracts contain a lot of functional interfaces that depict the other functionality of platforms such as Ethereum [3].

2.1.1 Why do we use ontology for Blockchain?

Generally, Blockchain is the distributed database that replicated over all nodes as a cloud computing architecture. These databases are distributed across multiple organizations.

That's why standard interpretation is needed that the data would be understandable for organizations. Interpretations are applicable via formal specification that enables verification and inference within software and applications executed on the network.
This is where ontology comes to play to ensure a common interpretation of data of shared database among different enterprises. blockchain as a modeling form used a different type of ontology:
informal/semi ontology to facilitate search and enhance better understanding of the business process for developing and applying on the blockchain.
Formal ontology helps the formal specification to automate inference and verify the operation of the blockchain. On the other word, blockchain modeling based on formal ontology can help the development of smart contract to execute on the blockchain.
Also, we can use ontology to capture data within blockchain: On one hand, It facilitates a better understanding of blockchain concepts for human. On the other hand, enables interlinking with other link data to convey deductions and formal reasoning[**?**].
Vocabulary used within ontology increase the transparency of transaction in a way that by describing the transaction in the context of linked data comfort the graphical representation of the location of such transaction. Thus, it increases also the capability of analysis by users.

2.1.2 Linked Data

Linked data is structure using vocabularies like schema.org to interlinked between different data and resources and it helps the semantic query of data. When information represents in linked data, querying about related information can be easily discovered. It is built up to standard web technologies such as:
- 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[Ugarte].

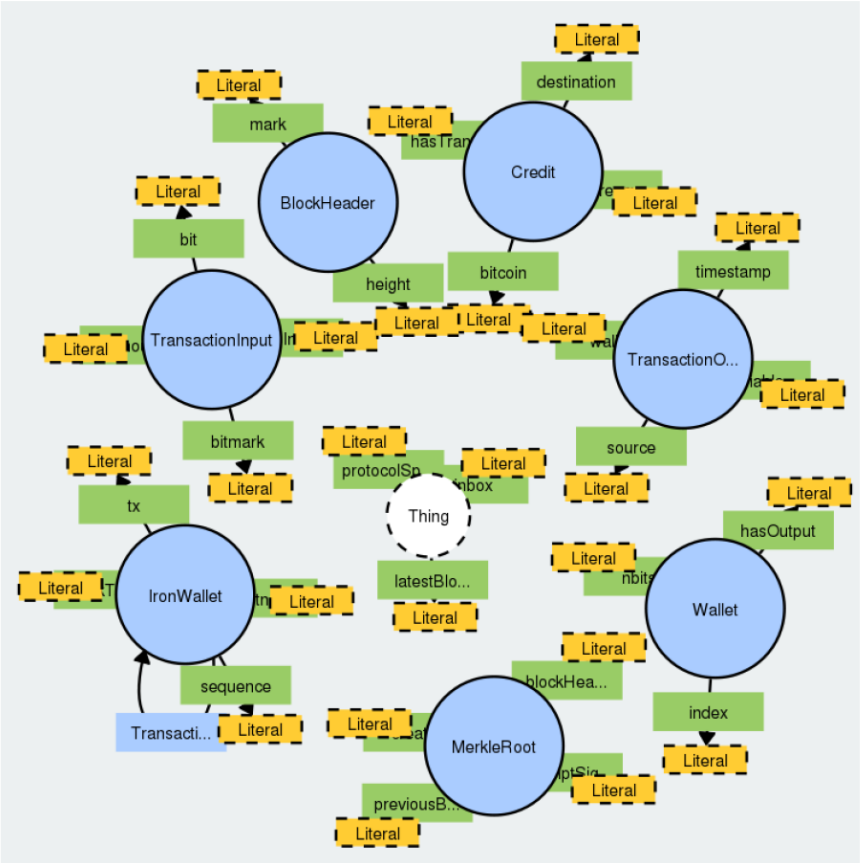

Figure 2.1: Illustration of Ontology diagram[**?**]

2.1.3 RDF

The Resource Description Framework (RDF) is a family of W3C specifications. RDF is used to describe and model information. It describes as a subject predict object is called a triple. i) Subjects that RDF expressions describe them. ii) predict is specific properties, attribute or relation to describe a resource. iii) The object is the name of property or value. We can build a graph based on these three objects.

2.1.4 SPARQL

According to Wikipedia definition: It is semantic query language for a dataset that makes us able to retrieve and modify data stored in RDF format.

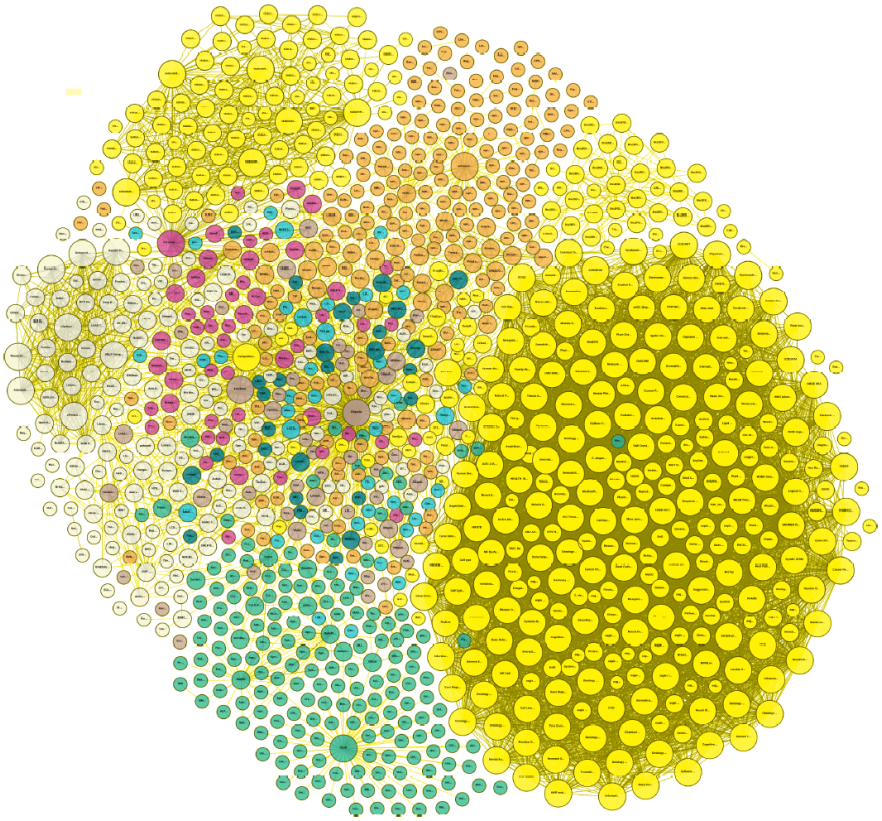

Figure 2.2: Linked data Diagram[Ugarte]

2.1.5 Ontology Web Language

Ontology Web Language is made to represent knowledge about things and the relations among them. OWL is a computational logic-based language, which means the language modeled in OWL can operate in a computer program like negation, intersection and so on.

2.2 Vocabularies

2.2.1 Vocabulary in Distributed Ledger

In order to generate linked 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 an early stage. Some systems define such vocabulary such as

FlexLedger, EthOn, BLONDiE[3].

FlexLedger: describes HTTP interfaces to blockchains, with a standard vocabulary and responses of these interfaces. FLexLedger is a protocol for decentralized ledger and graph data model which represents ledger creation, querying and data model using JSON-LD. However, FlexLedger does not have explicit vocabulary about ontology nor having concrete ontology for itself.
It is striking to say that the FlexLedger is not suitable to implement in some graph model like graphchain because in FlexLedger meta and the content data are stored together in the same graph whereas the GraphChain blocksâĂŹ content is stored outside the blockchain a separate graph [26].

EthOn is an OWl ontology that describes blockchain classes such as *"blocks, accounts, message" ,"state"*and relations such *"has parent block"*[33].

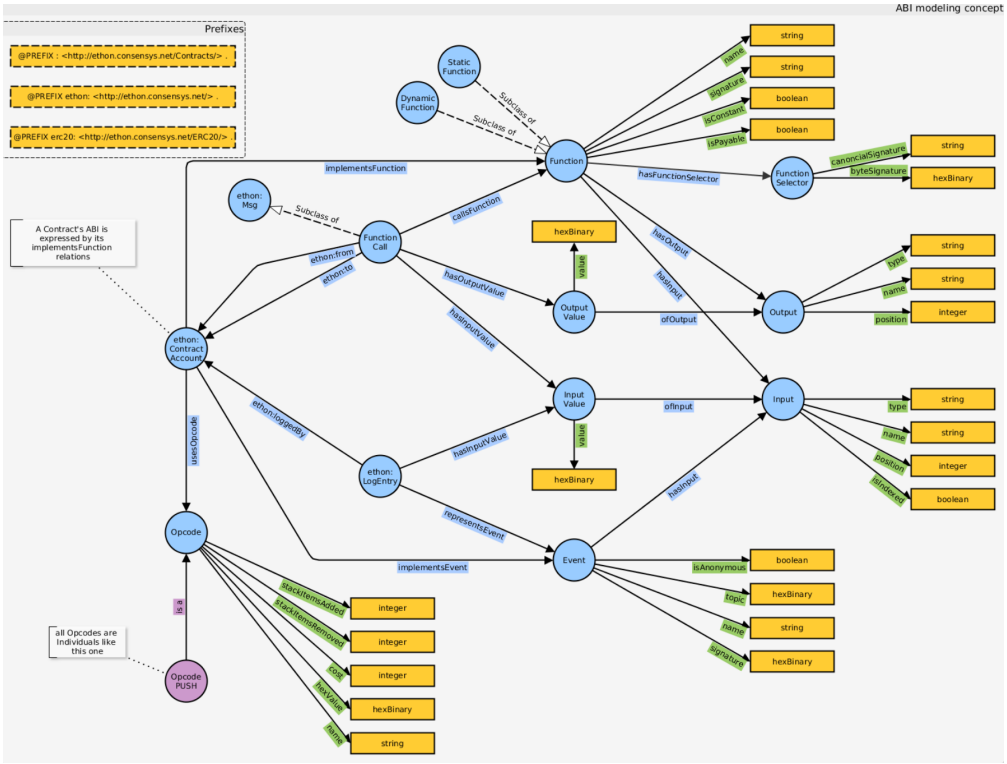

Figure 2.3: EthOn Contract Model(blue arrow is object properties, green arrow is data properties, purple circle is instance and blue one is class)[33]

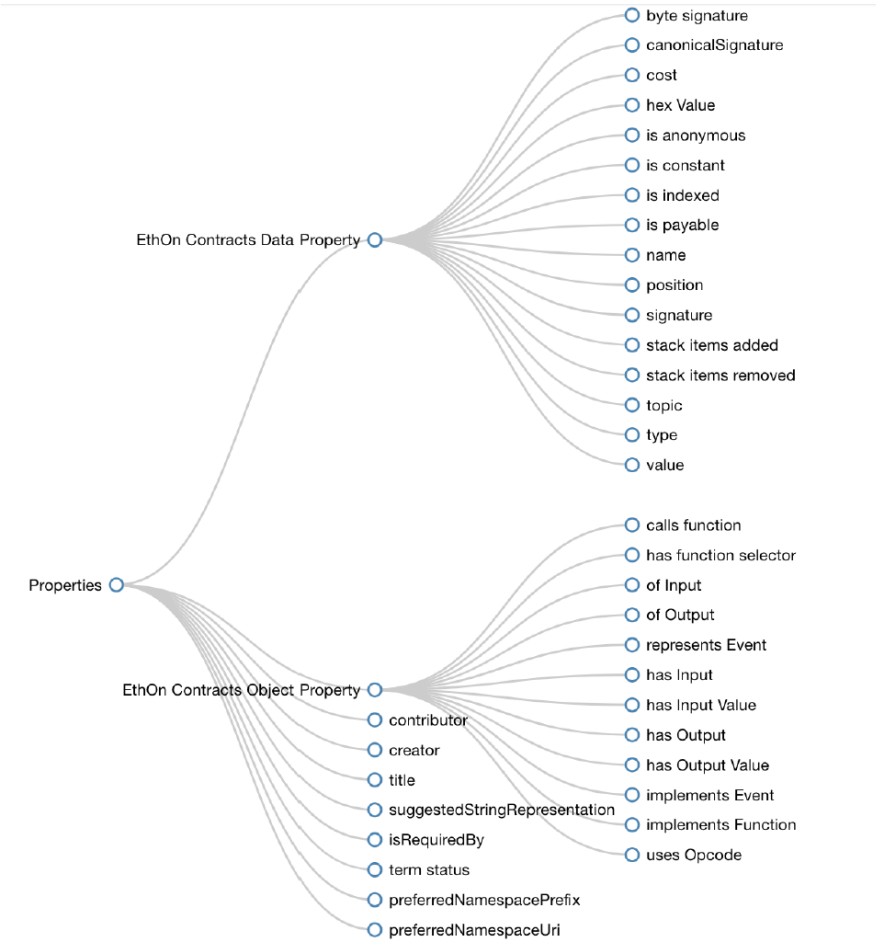

Figure 2.4: EthOn classes[33]

Blockchain Ontology with Dynamic Extensibility BLONDiE is another OWL ontology for describing the Blockchain structure like EthOn. But, it is more generic then EthOn. For example, EthOn and BLONDiE both defined some terms such as 'account', 'block', 'transaction' and some attributes such as 'transaction payload' or 'miner address'. BLONDiE defines some other concept for different Blockchain such as 'BitcoinBlockHeader' and 'EthereumBlockHeader' as subclasses of 'BlockHeader'. At the moment, BLONDiE supports two cryptocurrencies like bitcoin and Ethereum where all links and relationships between objects and attributes represent in RDF(Resource Description Framework)[3].

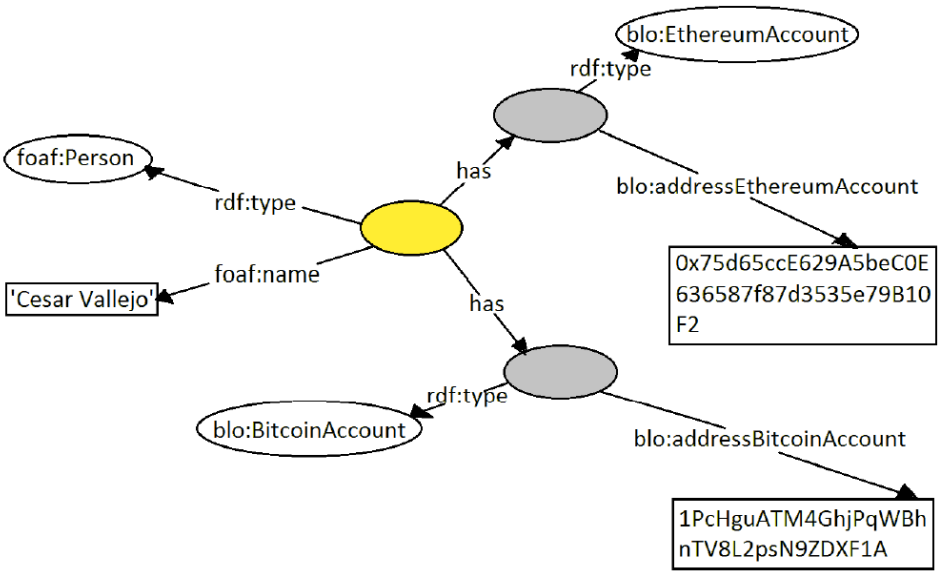

Figure 2.5: BLONDiE usage example[Ugarte]

2.2.2 Vocabulary in Smart Contracts

As mentioned earlier, EthOn and BLONDiE are both similar concepts that can be used for smart contracts. As the smart 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 the web and HTTP APIs which may enable us to annotate smart contracts as well. However, the contracts are not Web API and implementation may differ but the main concept does not differ. In other words, the vocabularies used to annotate web services, are used to annotate smart contracts too. It seems that the Combination of a distributed ledger with smart contract and web service due to profitability becomes common[3].

2.2.3 Ontology-based smart contract

Kim and Laskowski[15] propose ontology can help in development of smart contact which an be executed on blockhchain. This approach can come up with proof of concept using Etheruem or other traceability ontology[Ugarte].

2.2.4 Decentralized Storage and RDF

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

JSON-RPC is a remote procedure call protocol in JSON. This protocol allows sending data to the server without needing to the response. Ethereum blockchain which has properties such as pay fee with Gas. there are different ways of storing data in an RDF triple The most wallet in Ethereum in JSON format that is easily convertible to RDF and front-end is made up of HTML technology that we can use it to embed data on the website interface as Microdata, JSON-LD. The main focus of research is the storage of data on blockchain itself. There exist limitations of storage in Ethereum blockchain and fees. that's why, it is suitable to use compact RDF serialization. There are some methods to store data on Ethereum blockchain that are summarized in the table below[Ugarte]:

```
{
"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

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	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[Ugarte]

2.2.5 Semantic Blockchain

By increasing the usage of blockchain technology recently, the need for semantic reasoning on the distributed ledger is on the crease as well. The blockchain is the best platform to utilize semantic web principles in this technology and add a new trusted property to a dataset. It makes a new dataset so trustworthy. Using semantic web technology on blockchain in a novel idea and the way of how to apply this technology in blockchain and smart contract is also an as controversial issue.
There are some **definition of semantic blockchain:**
Semantic blockchain is the representation of stored data on distributed ledger using linked data.
Semtcic blockchain is the applying semantic web standard on the blockchain that these standards are based on RDF.

2.2.6 semantify Blockchain process

Semantic blockchain or semantic distributed ledger affects the industrial world and subsequently, the result leads to start developing new application and framework to combine two worlds: thee are some way to sanctify blockchain: -Mapping the basic blockchain to RDF making usage of vocabulary, ontology and so on. - Storage of data in a blockchain is expensive, The only way is to store the hashing point to data set in blockchain and

then share RDF on the blockchain. - Create semantic blockchain that internal data exchange protocol is based on RDF[Ugarte].

2.2.7 Semantic Ontology Mapping

To generate RDF, it needs to map the basic blockchain entities to relevant semantic web terms, concepts and ontology. To make the query more efficient, BLONDiE used some ways such as:
Firstly, records relating 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 is strengthened with link some enteties like: blocks or smart contract and input to originating contract.
Blockchain stores just a 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 a contract. This specification is created when the smart contract is compiled and stored in the blockchain and also smart contracts will index on the blockchain in binary form. To interact with contract Application Binary(ABI) Interface specification is needed. This specification is in the form of JSON and created when a smart contract is compiled and stored in the blockchain. The ABI determines all functions of contracts and descriptions about input, output parameter for each contract[3].

By the presence of address and ABI, it is obvious to generates RDF to index smart contract. Allan Third at el.[3]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.

2.2.8 Minimal Service Model

The MSM defines a service that has Operations. Operations have input, output and default MessageContent that may include mandatory or optional MessageParts. Mes-

```
{ "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

sagePart provides support for finer-grained input/output discovery, as available in OWL-S and WSMO[33].

We present one example that Rashid et al. used to clarify more the meaning of MSM in the blockchain: In this example, educational data is used to store in the blockchain using Ethereum *web3* library.
it is considered that every block adds to blockchain and retrieve transaction within the block. If the transaction contains a smart contract. Then it retrieves the contract address using Ethereum API. Then, it saves the smart contract address, Application Binary Interface(ABI) as triple in RDF. ABI describes the name of smart contracts and way of calling them. Auther, then saved each method in ABI as an RDF triple based on MSM ontology as follow[33]:

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')

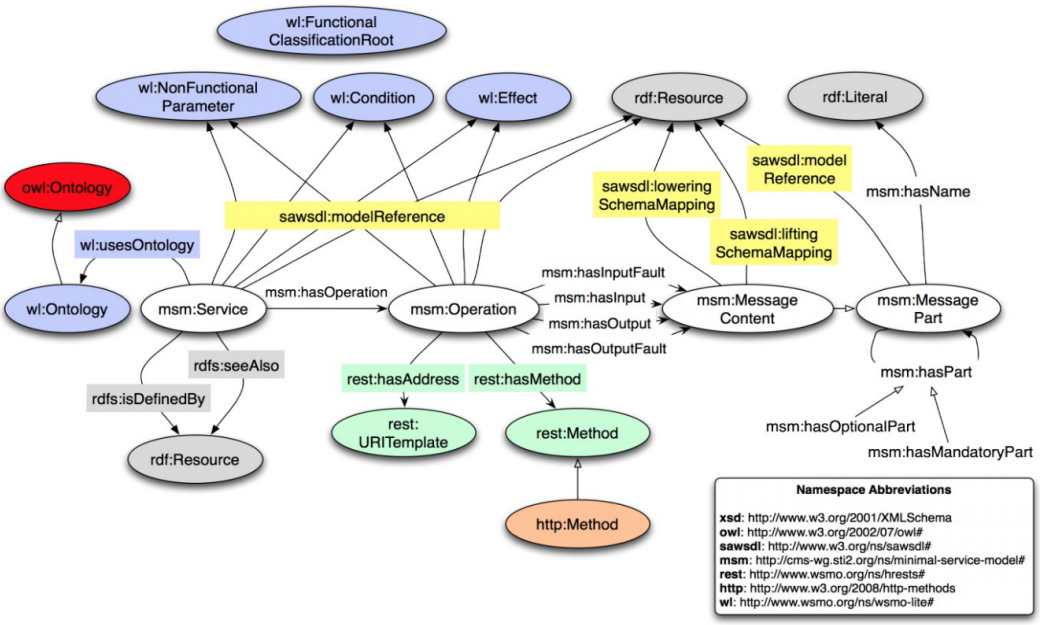

Figure 2.9: Illustration of Minimal Service Model Ontology[33]

2.2.9 Web 3.0

[Ugarte] The evolution and interaction of people on the Internet emerged on a classification of this revolutionary technology. Early websites formed what is now known as Web 1.0 the âĂIJweb of documentsâĂİ, documents serving as information portals with basic ability to link websites. Later the Web 2.0 emerged as the âĂIJweb of peopleâĂİ, it introduced users to collaborate with content creation and alteration. For the coming Web 3.0 there is a debate about what is the proper definition of its characteristics. For some group, the Web 3.0 is powered by the Semantic Web, where people can access to linked information fast and easy. But now, with the emergence of a decentralized web powered by Blockchain technology and since it enables unmediated transactions, there is a new focus on the Web 3.0 based on through the trustful nature of the blockchain. It is the âĂIJread-write-own webâĂİ. Here, the user owns and participate in owning the protocol. It is both peer to peer and machine to machine. And it is applicable to people, companies and autonomous entities [14]. For instance, the term Web 3.0 is used by Ethereum in a different context than the suggested by Berners-Lee. It is proposed

as the separation of content from the presentation by removing the need to have servers at all [15]. Stephen Tual, EthereumâĂŹs CCOâĂŹs, defines that what makes Ethereum different than Web 2.0 is that âĂIJthere are no web-servers, and therefore no middleman to take commissions, steal your data or offer it to the NSA, and of course nothing to DDoSâĂİ.

2.3 Retrieve Information form semantic blockchain

Reasoning and knowledge representation in the semantic web is based on Description Logic(DL). Description logic has some elements such: classes and properties, the link between a different object, etc. Ontology is the study about the objects and relations between these objects or entities. Each DL has set of constructors to combine the concept and roles. Concepts used in inclusion and definition axiom that model knowledge domain. These axioms called terminological box(TBox) with respect to ontology individual axioms from Assertion Box (ABox). ABox and TBox create Knowledge Base(KB).
This study extends in the keeping with the retrieving the most relevant resources for a given query by the requester, where both query and resources are ontology-based and matched called semantic matchmaking. This process leads to two approach *full match*, *no match*. For request R and resource check whether all features in request exist in S. The classic inference service is capable to Boolean match approach. But, It is not enough because fully matched is rare. Non-standard inference determines the semantic ranking of resource with respect to request and logic based results. If request and resource are not compatible:
- contradiction concept which part R is not compatible by S. Consider G is the part that has conflict and K as part where S and R are compatible.
- Concept Abduction: R and S are compatible but S not match fully with R. consider H as Hypothesis that represent what is request in R and not exist in S. If R and S are not matched. Use contraction to represent K part(compatible part) and H hypethesis to suggest part to reach full match.
Moreover, used number of element G and H in penalty function computation to define semantic distance matrix. This matrix can be used to rank resource with regard to request[Michele Ruta and Sciascio].

2.3.1 Semantic Blockchain Architecture

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

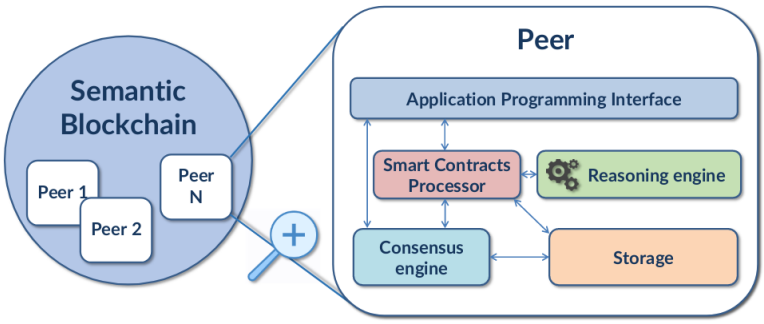

Figure 2.10: Framework Architecture [Michele Ruta and Sciascio]

Peer agent is identified as by public key and its accounts. Each agent can enforce semantic discovery to transfer assets between annotations that are stored in the blockchain. And each peer is 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[Michele Ruta and Sciascio].

2.3.2 Making smarter Contract: Putting Semantic in Consensus Protocol

In the 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 these concepts, it is possible to compare requests with different recourse concerning semantic annotation of shared domain ontology.
Smart contract semantic is represented by a service layer that gives a correct description of discovery outcome. Thus, it raises the trust of the discovery process for users. In the

semantic blockchain, Baqa et al.[12] used domain ontology to maps to EthOn contract. These concepts used for OWL-S ontology, indexing and invoking smart contracts on Blockchain via URIs[12]. Semantic blockchain also performs operations such as registration, discovery, selection, and execution that are implemented as a smart contract.
Ruta at el.[Michele Ruta and Sciascio] focused on semantic mismatching as a significant feature of semantic blockchain concerning basic blockchain. This allows computing the semantic distance between resources and queries with the same ontology. The logic-base matrix enforces the semantic ranking of the element of the query[Michele Ruta and Sciascio].
A: Resource Registration: As already mentioned, a Smart contract designed using OWL specification. It used concepts and entities defined as a class in OWl language and relationship between entities defined as object properties concerning ontology. To construct smart contract semantic requires a framework to design such contracts.
Multiple resources stored on the blockchain and Each domain is related to a different ontology that provides 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 describes resources, resource price.
To register resources on the blockchain, a user is required to register an account with public address and related private key to call the smart contract as a parameter. In ontology, a user describes as class corresponding to an account along with other attributes such as an address, private key, and other details.
To achieve this, A new ontology called *EthereumContractConcepts* is implemented containing *EthereumContract* data property. A *ContratcAccount* consist of license number, date, smart contract owner and state. After all, a smart contract is developed with vocabulary as domain-based-ontology[12].

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

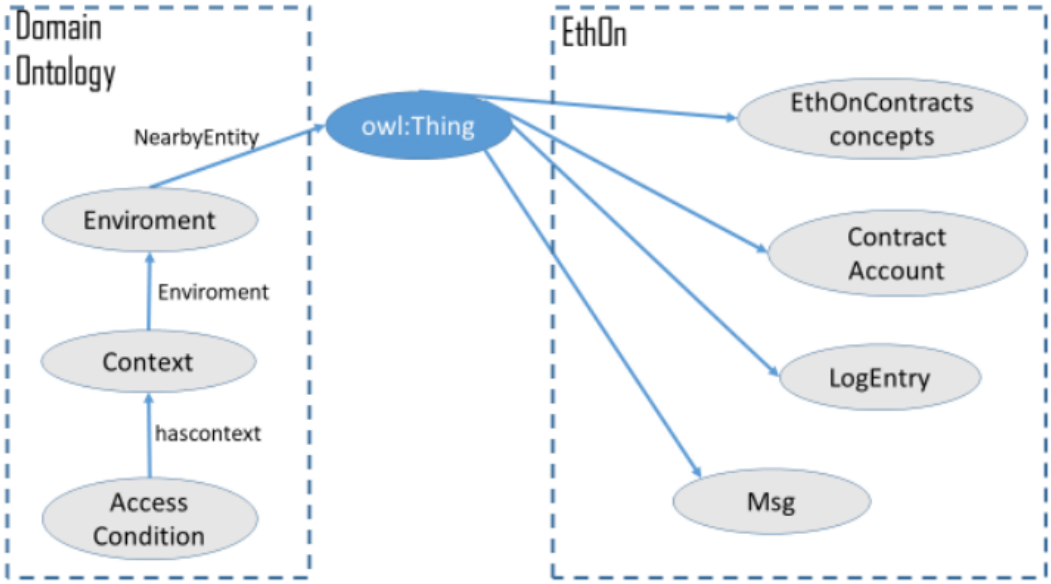

Figure 2.11: Domain ontology [12]

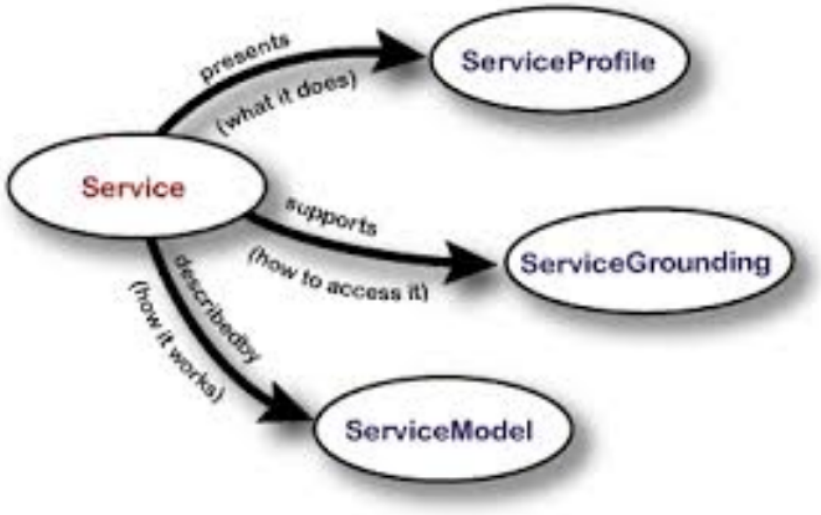

Figure 2.12: Service-based smart contract

By the use of EthOn concepts, we can monitor added block to the blockchain. When a transaction has a smart contract, it is retrieved from the store contract address via API, then the address of smart contract and Application Binary Interface(ABI) as a triple in the RDF store. ABI consist of the smart contract method and way of calling it. Methods are stored in ABI as an RDF triple concerning OWL-S ontology. The table below shows the OWL-S vocabulary to describe smart contract[12].

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

B:Semantic Discovery To search for a smart contract or an item, The requester sends 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 contracts 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.

Baqa et al. here used the *gossip* approach to propagate the request. The nodes which received the request perform 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 relevance score s_min and cost $p_i <= p_{max}$, is returned.
Second, send the request to other n nodes randomly, the other nodes perform in 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 perform matchmaking locally.
Afterward, the nods send back the result to the main requester at the specified address[12].

C: Explanation In this process, the requester sends the request containing: Semantic is the annotation of request and URI od related resource. The receiver node response matchmaking result that encompasses the semantic dependency score in [0,1]interval and the concept expression of G (conflicting part between R and S), K(the compatible part between R and S) and H(the hypothesis concept to reach the matchmaking). This process is optional and needed when the requester needs the explanation of matchmaking results.

C: Resource Selection: After receiving all results, the requester selects the best resource and smart contract, sending a message to the resource owner and contextually payment. the receiver responses with the proper resource representation relied on the meaning of uri[4]. The resource discovery and retrieval interaction are shown bellow and Each associated transaction is recorded on the blockchain[Michele Ruta and Sciascio].

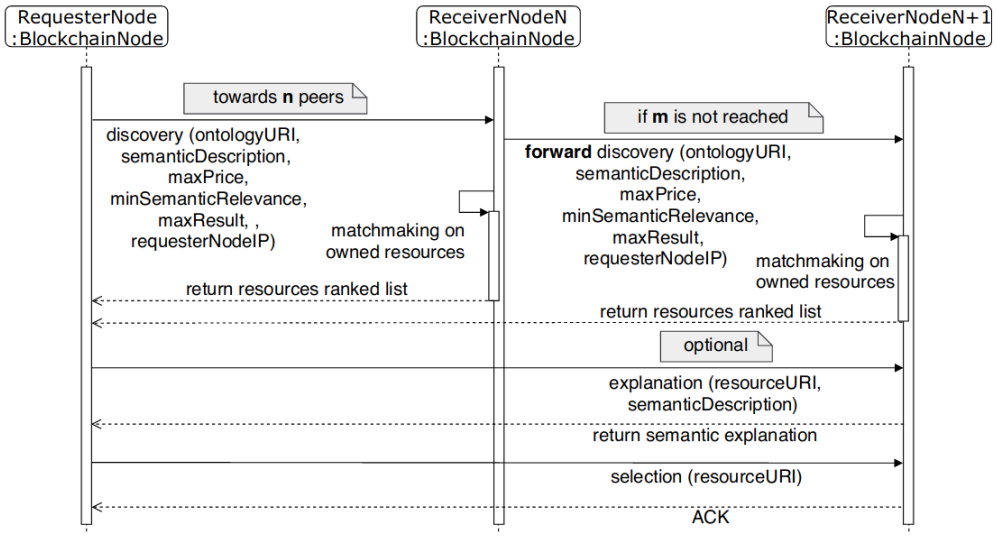

Figure 2.14: Resource Discovery[Michele Ruta and Sciascio]

2.3.3 Indexng of Smart Contract

As already mentioned, the distributed ledger does not have a central registry due to its structure. The contract in this distributed ledger is not directly queried. and blockchain is the time-ordered structure where data exist on multiple blocks. In such a distributed structure, developing the indexing process for the smart contract is necessary. The indexing system provides the ability to analysis and search service on blockchain and displays the outside. There is a different level of indexing such as low level to index basic entities such as account, transaction, block and the top level for more functional operations is indexed.

2.3.4 EthOn

Is an ontology that describes blockchain concepts such as transaction, account, block using w3RDF schema and ontology web language OWL to describe the relation of these objects on the blockchain. 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 concepts, functions, events, input, and output, etc. EthOn accepts consider just some relation between smart contract and Ethereum framework. Some multiple tools and languages can annotate a smart contract. In this study, It is focused on OWL-S ontology due to some significant features such as flexibility, widely description of service and no- restriction... The main goal is to develop to support the smart contract too for semantic web service context. For reaching this aim, the web graph chain is PI and a smart contract can be represented as an executable function in a distributed environment. The semantic web describes message, service, and entities in a machine-readable format that can support logical reasoning based on semantic web service description. It helps to map between different ontologies that facilitate the transformation of the concepts among ontologies[12].

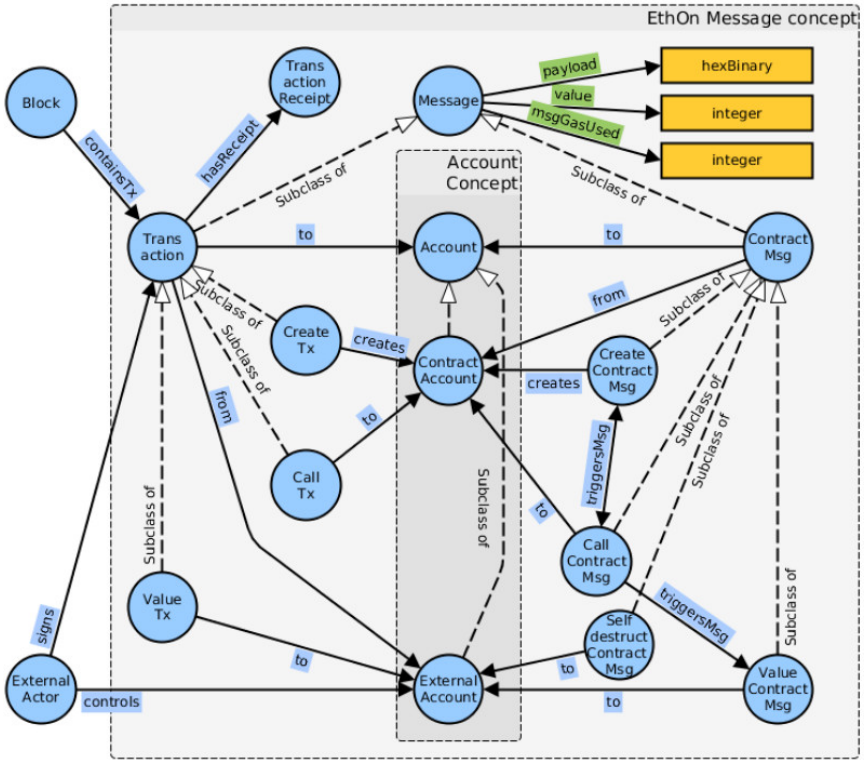

Figure 2.15: EthOn message concept[12]

Chapter 3

Use Case: Supply Chain Management

3.1 Supply chain management

Supply chain is a amount of activities that information flow product form suppliers to customers, manage all activities related to sourcing, conversions, delivery and etc. Supply chain management, not only is beneficial in various Fields such as financial, social and world. It enhances resources, exploitation and cyclic time from material to end users. Supply chain processes started within the companyâĂŹs processes or distribution. We believe that supply chain management is essential for the delivery of business results[13].

Supply chain management is critical factor in global business and improvement effort become increasingly important. Supply chain management is an great strategy that make alignment flow of raw material, manufacturing, and distribution to customer according to customers demand.
Supply chain management is an approach that integrate the network of operating entities to delivery system fulfilling the satisfaction of customer and protecting the competitiveness of supply chain. In this regards, Supply chain is the chain of activities and demand marketing service[34]. Christopher (TODO) claimed against uni-dimensional supply chain as follow:
"Supply chain management should be termed demand chain management to reflect the fact that the chain should be driven by the market, not by suppliers. Equally the 'chain'

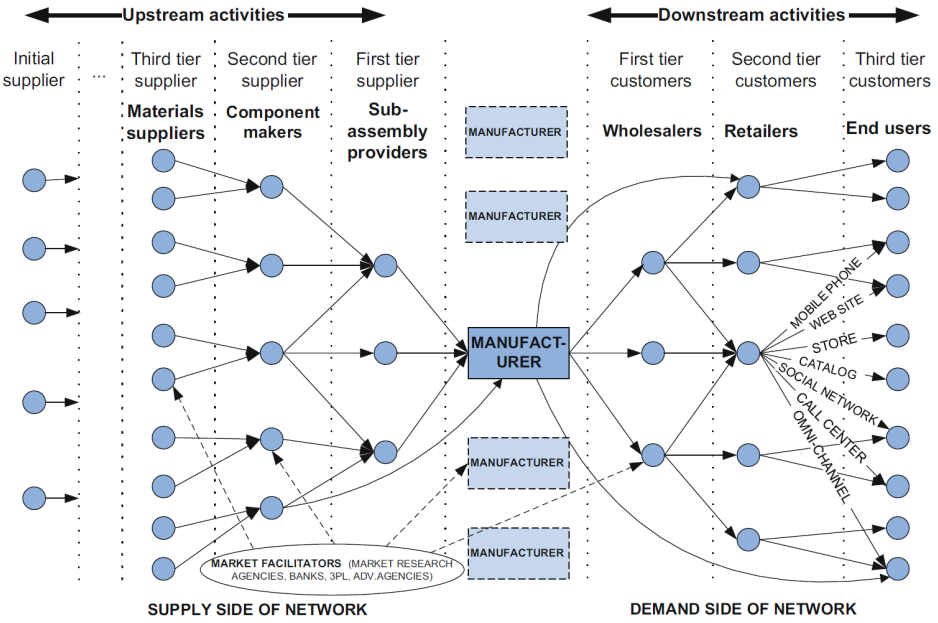

Figure 3.1: Network supply chain management[8]

should be replaced by 'network' since there will normally be multiple suppliers and, indeed, suppliers to suppliers as well as multiple customers and customers to be included in the total system."

3.2 How to improve SCM applying blockchain?

The decentralized nature of blockchain couple with data immunity made it appropriate for supply chain management. This feature provides opportunity to fulfill requirements of supply chain. Using blockchain, from one hand store data on supply chain, thus data can not tampered easily. On the other hand, data from multiple stack holders can be integrated into blockchain rather then stored in individual system. One of the critical factor in supply chain management, manufacturing management, delivery management, is data management. The type of data in supply chain is not bounded to inventory information. Data steaming is as core process in supply chain management.

Data for SCM fed into blockchain via information flow. In this process information flow previous input and stack holders store data in off-line way. Information can ex-

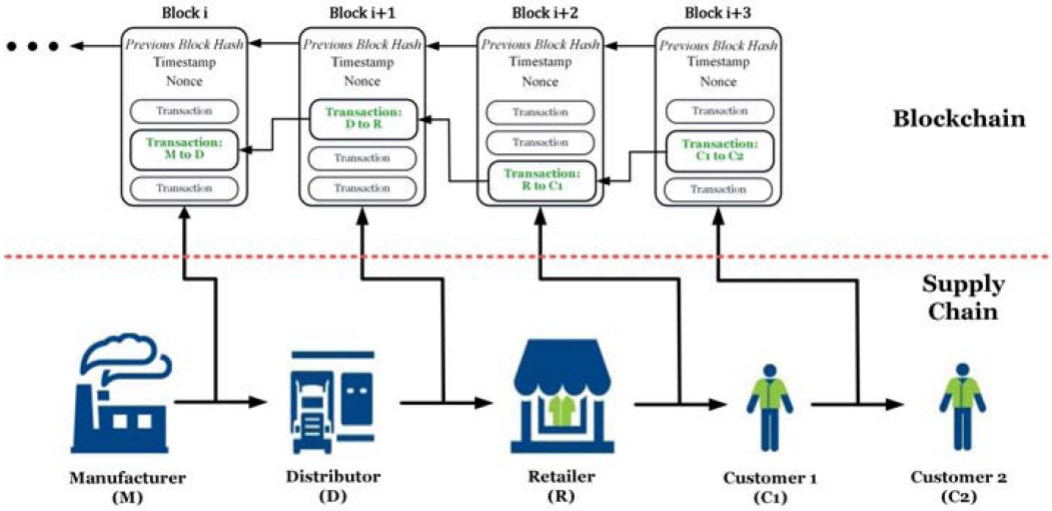

Figure 3.2: Supply chdain management with blockchain[23]

change through postal system. One possible way is to automate information flow, but it brings some concerns such as manipulation of data veracity and violate security and time consuming of data processing. Therefore, we can improve SCM system by sharing, information within supply chain, protect data veracity and accelerate the data recovery. But there are some issues to do this, one of them is to gather data from external environment convert to digital information via IoT devices, smart sensors and etc. The other is to identify how implement blockchain on SCM through various approaches. So that, they can obtain more requirement of SCM from industry.

Generally, critical property of blockchain which is decentralization cause to deal with three concerns such as Trust-less, security, and authentication. We deal with these issues in SCM by traceability, efficiency and transparency. Blockchain application is on increase in payment infrastructure, cryptocurrency and verification. In supply chain management, blockchain not only offers transparency regarding who is participating and which action take place, but also provide solutions for real-time supplier management, shipment management, identity management and etc. Blockchain is beneficial to all parties and supply chain network.

3.2.1 Blockchain operation in supply chain management

The main benefits of blockchain technology application have been concerned on how it can provide valid information to supply chain network participants. The need for collective activities will disapear, Through such transparent information sharing mechanisms. Participants can rely on transaction history to transfer assets like ether, even without needing to third party[16]. Blockchain emphasized on four attributes: 1. Information synchronization: all participants has access to same data containing history of transaction.

2. p2p network and consensus mechanism: there is no need to third authority to control transactions

3. Smart contact: all obligations are formalized in programming code which should be met.

4. Data stability: Modification must be verified just with consensus of all participants, Not altering without collective verification[16].

Based on Four common attributes of blockchain summarized in table follow:

Context	**Logistic**[9]	**HealthCare**	**Banking**[11]	**Engineering**[6]	**Green SCM**[18]
Information synchronization	digital version of a ledger to decentralized system participants to follow data	Allow copies of patient's record to update with participants records	Remove link of third party financial institution in distribution of goods	participants interact with blockchain via private and public key	all participants have same copy of ledger which is updatable by new modification in blockchain
P2P network	participants share transaction with blockchain and copy of ledger	Participants have copy of whole blockchain to ensure that date is authenticated	Helping system participants control big data and constructing ownership	transaction broadcast into blockchain via user's node	Participants can follow record of transaction, thus can track performance of transportation
Smart contract	legal provision are formalized into code and verified through a network of peers	records stored on blockchain	ensure that payment is done automatically after reaching result and reduce manual risk	script storing in blockchain is similar to stored procedure in relational database management system	Automatic payment is done, when regulations are met or value added to products
Data stability	transaction are approved by consensus of all participants, it stores in block	Create indecomposable databases for medical records and tracing pharmaceuticals through manufacturing and distribution	after entering information into system, can not be altered, thus preventing from subsequent fraud	transaction will be verified through hash of previous block in blockchain	Tracing the waste, distribution responsibility to system participants for cleanup cost

Table 3.1: Summary of blockchain characteristics in various contexts supply chain management.

3.2.2 Analysis of supply chain ontology models

This section reports on the analysis using the comparison framework. The analysis presented here provides insights about the gaps in existing supply chain ontology research. Table lists these ontologies shows the values for the criteria application, scope, KR paradigm, language and methodology approach.
Scope concerns with the field of industry which used in ontology.
Language is the type of language that is used in ontology.
Application is concerned with the type of problem-solving task for which the ontology is to be used. These tasks span a wide range of applications in supply chain.
Key concept is about main concept used in ontology.
Purpose concern with the goal of using ontology using in supply chain[31].

Supply chain Ontology model	KR paradigm	Language	Scope	Purpose	Key concept	Methodically approach	Application
Model by Sousa at al.[4]	Algebra of sets	UML	Business network	Improve manufacturing performance of virtual enterprise by bridging gap between planing level and control level , integrating incoherent manufacturing systems	plan, unit, product, organization, order, resource, customer, activity.	Combination of generic techno-organizational requirement and synthesis of ontology	Production and operation planning virtual enterprise of and control semiconductor industry
TOVE ontology[15]	DL	XML	internal supply chain	Develop enterprise model which will have ability to deduce answer queries in industrial environment	agents, roles, positions, activity, resource, commitment, communication, authority	Ontology developed and comply to pose competence questions at the beginning	Using TOVE model to analyze enterprise ontology
IDEON ontology[21]	Algebra of sets	UML	inter-business network	provides a common foundation for designing, managing, and controlling collaborative, distributed enterprises	process, organization, resource, enterprise, product	there is no formal evaluation, however the ontology was assessed based on its actual uses against the purposes.	support multiple enterprise applications: Crisis Action Planning and Integrated Product-Process Development which enabled systems engineering.
Model by Ye at al.[38]	DL	OWL, SWRL	inter-business network	serve as an interlingua to enable information integration across interacting applications and semantic integration of heterogeneous system in supply chain	structure, activity, performance, purpose, role, party, transfer	SCOR provides a basis for describing the knowledge of supply chain performance; The same methodology for the development of EO was used	application integration framework based on developed ontology is used to solve problem of information integration
EAGLET ontology[10]		UML	supply chain of thing	facilitate the visibility and inter-operability of things along the supply chain.	Event, agent, location, equipment, Thing	Hybrid:inspiration, synthesis and compliance of this ontology with real-world practices is much higher.	1. extend formalization beyond current knowledge 2. address more supply chain practice; 3. used as an instantiation pattern; 4. focus on fixed asset and agent visibility
MSE ontology[20]	DL	RDF, OWL	inter-business network	provide semantic and syntactic interoprability service to enable understanding of basic manufacturing terms between different manufacturing system engineering enterprise applications	product, enterprise, project, process, resource, strategy, flow	built on the experiences and knowledge of published manufacturing system information models.	There are some application area thereby companies enter into tempoSEMrary inter-enterprise collaboration for exchanging information
Enterprise ontology[25]			not applicable	facilitate enterprise design and analysis by supporting communications among human	person, activity, plan, resource, marketing, organization, purpose, strategy	inspiration and synthesis; ontology was assessed based on its uses against the original purposes.	improve method with a framework for integrating method and tools which are appropriate to enterprise modeling and management of changes

Table 3.2: Different supply chain ontology models

3.3 How smart contract improve supply chain

Before era of smart manufacturing, supply chain management is and log process called as black box, in terms of input-output mechanism. This black box consist of multiple mini process related to product, services, customer and etc. Many possible by emerging new technology like smart sensor and IoT devices, This long process are made more cost-effective, transparent. Meanwhile collaboration between supply chain structures bring strategic benefit to involve all parties.

Using some new technology in supply chain management such as smart contract, IOT devices and consensus mechanism in blockchain make this system more cost-efficient, efficient and secure at the same time. Smart contract can be implemented through out life-cycle of supply chain process of raw-material to end users. For instances, smart contract can be used in terms of demands of product personalization, utilization and services, subsequently automated payment machines that are connected to IOT can co-operate using smart contract in terms of requirement that must be met before going to next machine.

In additions, Based on blockchain with logic rules, smart contract can be used to control quality and cont management between company and Original Equipment Manufacture(OEM). Also manufacture can use smart contract with authorized provider in terms of how product should be repaired or updated and son on.

(blockchain based trust mechanisimm IOT) Furthermore, Smart contract can simplicity the performance of supply chain and improve transparency across supply chain. Using IOT devices to track the locations of goods. Smart contract can be helpful in this case determining the provenance of goods. This is important part in all industry special in food industries. recently, some big companies start using smart contract to trace the device racing the goods, including meet, foods, containers, and even diamond[Ada].

3.4 How smart contract works?

To understand how smart contract works?, the first we should know how Etheruem operates and how smart contract comes into play? could any The main term of Etheruem is account, any time that there is exchange information between accounts, it captures in blockchain.

There are two accounts in Ethereum : One is externally owned account that controlled

by private key, the other is contract account which is activated by EOA and controlled by internal code.
In order to create smart contract, participant should identify an opportunity to agree to other parties. This could any values such as good and service. Then they must set term and conditions that have to b met. This could be triggered by participants or external events. All agreement will be written in code as smart contract which can be deployed in blockchain where executed by first initiating contract.
To initiate smart contract, participant fulfill transaction with contract account which encrypted by private key and transmitted to other nodes in blockchain. The other participant verify transaction if it is triggered by valid participant. After all, the transaction is added to blockchain after obtaining consensus by majority.
Then, smart contract will be executed and recorded, finally, all nodes are updated in network. But already mentioned, the outcome not altered and just states of blockchain will change. Note that, every initiator has to pay for transaction fee to create changes in the state of Ethereum platform. The smart contract used in Ethereum to store information on blockchain securely[19].

3.5 Addressing supply chain using smart contract:

Using blockchain guide to solve cooperation challenges in SCM. Blockchain improves the transparency and verification in supply chain and smart contract handle the exchanges between many participants and nodes in blockchain.
Smart contract can handle improve this issue three factors: Tractability, efficiency and transparency.

- *Transparency:* Supply chain improves traceability in supply chain by storing provenance of goods on blockchain. this will ensure the manufacture where goods, raw material are coming from valid sources and customer have more confident that purchase legal products.
  Another usage of smart contract in supply chain. In addition, transparent authentication across supply chain. Participant can verify easily each other using certification stored in blockcahin in digital form. All together, allow supply chain

manager to have correct decision when selecting supplier to work with.

- *Traceability:* Smart contract helps supply chain in terms of traceability in a way that it can easily trace the inventory at every long way From raw-material to end user delivery. The product can be traced using serial number, tags or smart sensors.
  With the growing of IOT, device can connected to each other. smart sensor provides some information such as location, environment, and quality of product. Another problem in supply chain process is that the goods will be depreciated when arriving to users due to delay. Smart contract attibutes ca mitigate delay , keeps the supply chain agile to be fully- equipped to cope with the product delivery distribution such as natural catastrophe and etc.
  As already mentioned, one of the main usage of smart contract is tracing the inventory in supply chain and facilitated the supply chain system to be track the products.

- *Efficiency:* Smart contract can also improve the efficency of supply chain in a way that using self executed attibute of smart contract help supply chain to eliminate the paper work method and just executed smart contract onlyany time that obligations are met. Anther thins that using smart contract as code enhance the cost efficiency. because it eliminate numerous document and paper work required. it helps to reduce costs and amount of effort undertaken[19].

3.6 Concept of supply chain

To understand supply chain process, author analyzed the stack holders. In figure 3.1 indicate different exchanges between stack holders that could be information, goods or products. This is necessary to understand the requirements of stack holders to mange the supply chain process.

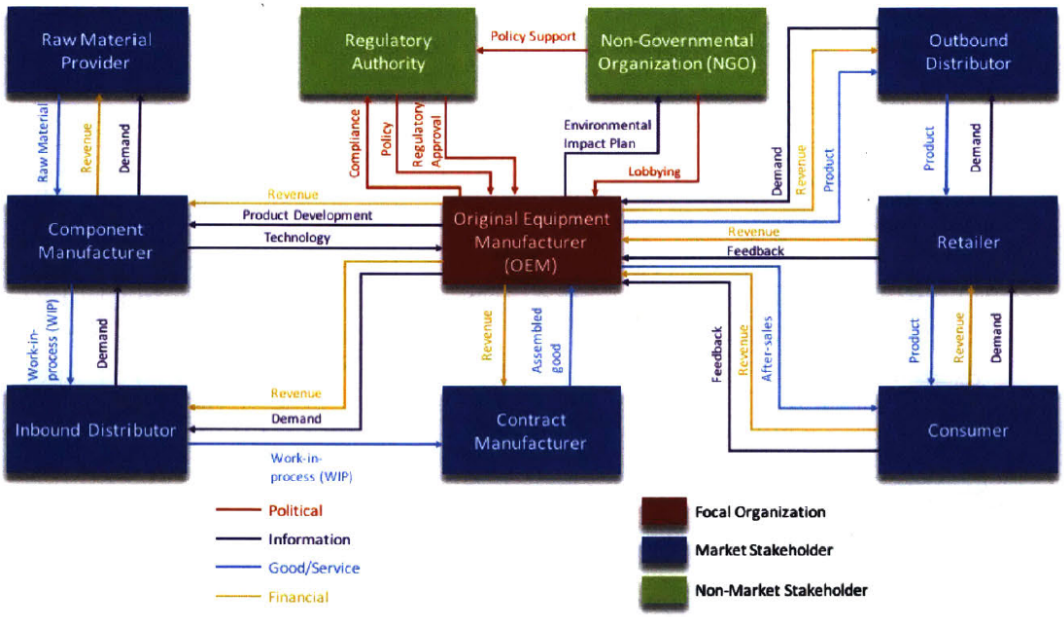

Figure 3.3: Stack holders network for a generic supply chain

- **OEM:** Original Equipment Manufacture has key role among all stack holders. OEM has multiple tasks such as forecasting demands of customers, inventory management, control flow and product development and marketing. In fact, the main goal of OEM is to develop product, distribute them, observe satisfaction of customer by fulfilling their demands.

- Raw-material: it is the first phase in supply chain where it's responsible for providing the raw material for manufacture providing high quality of raw material.

- Component Manufacture: raw material is used by component manufacture to produce component with high quality and earn revenue and trust of OEM in return.

- Inbound Distributor: Obtain component from manufacture to distribute it to contract manufacture assembly.

- Contract Manufacture: the OEM can choose to outsource final assembly to a contract manufacture to build product.

- Outband Distributor: it receives final product from OEM and distribute it to retailers for sale.
- Retailer: It is responsible to sale the product to customer. Maximize sale of product and enhance the quality of products.
- Customer: It is the end of supply chain process and is main source of demand.
- Authority: Is non market stackholders and has main role in this process to ensure that products meet requirements and minimize danger.
- Non-Governmental Organization(NGO): It is a non-market stackholder that have impact on product. Their goal is to public awarness of issue and increase their popularity[19].

3.7 Challenges encountered in Supply chain

As supply chain is long and complex process. There are some challenges which segregated into two types:

- **Planning stage:**
  It deals with forecasting the customer demands from one hand and inventory management on the other hand. *Demand prediction* is the main challenges in supply chain. Supply chain develops some models to deal with this issue. As the demand of customers change overtime. Supply chain build up some models and analysis methods to predict demand using data and other characteristic.
  Inventory management, supply chain tries to keep balancing between supplier and customer. If supply chain is too low, client and customer relationship will be affected negatively. If supply is too high, that will have much inventory and it costs lower price. Apart from all, the most important thing in this process is time managing to minimize delay and delivery time to customer. Supply chain build some models to fill the inventory to maximize profit and minimize costs.
- **Coordination stage**: Due to multiple stockholders in supply chain, it exist information inconsistency. Lack of information poses some challenges in supply chain that we referred to some of them. It affected the product traceability negatively.Lack

of real time information to track the product, it is difficult to update demand prediction and consequently destroys the trust between stack holders. One suspects information is being held by others and so on.
In addition, The product quality and information sharing hold the business structure together,that allow supply chain to improve coordination and product quality, manage the risk or distribution due to natural catastrophe[19].

3.8 How to improve challenges using smart contract?

Supply chain used smart contract to deal with three concerns:
1- Provenance of goods
2- Good traceability
3- Using open database to build trust between stack holders
To determine **provenance**, Supplier A(main supply chain) records the detail of raw material by the usage of smart sensor the location of materials and time of shipping can be stored into blockchain which will be available to all parties.

Using the provenance the source of product will be verification for all parties. In order to **tracking** the product, all parties should record all detail of shipment in blockchain. Whenever a good received on the specific location, contract will be triggered to shipping party. The task of OEM is to determine time and location of shipping .

Supplier A, send shipment to Supplier B and record on blockchain. Supplier B also records on blockchain by receiving shipment. Smart contract then check whether all predetermined obligations are met. Afterward, payment will carry out otherwise, an alert is triggered to parties to eliminate problems. Automate tracking helps to simply the supply chain process and update states. That's why the supply chain will be more agile to un-predicted situations.
As already said, supply chain is**open database** where all parties can store the detail of information in blockchain. By every shipment, the reputation or credit of parties will increase which would be included smart contract for tracking goods. When smart

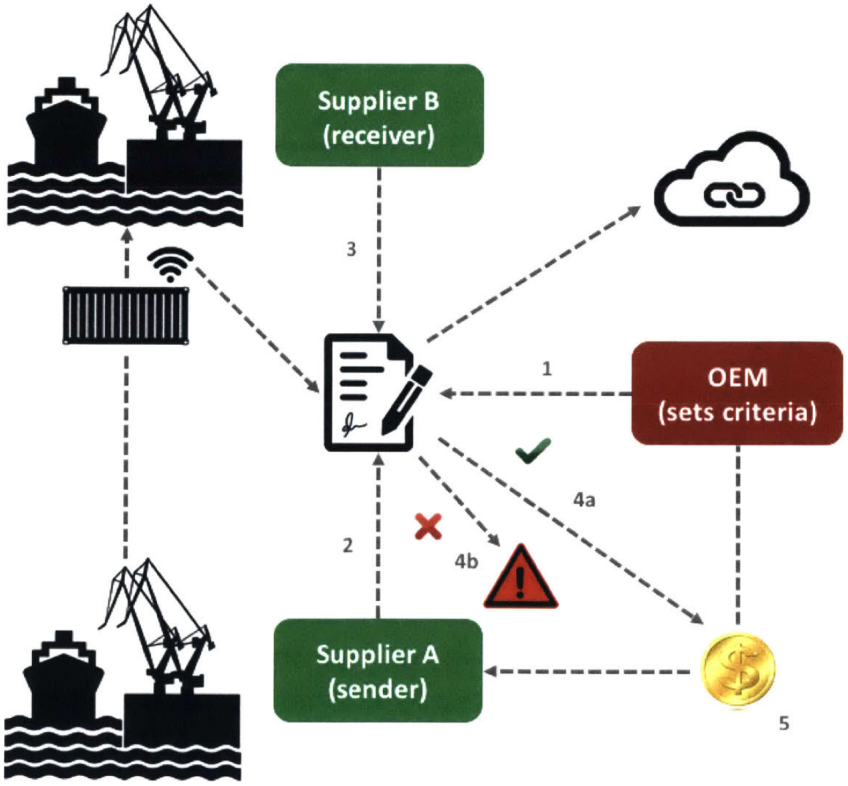

Figure 3.4: Using smart contract to track the goods in supply chain management

contract call a contract for an transaction, score (credit) also be accessible. The open database help smart contract for more transparency[19].

3.8.1 proof of concept development

Tracking contract let the parties to trace the shipment of goods and execute the payment automatically once shipment is completed and predetermined conditions are met. In smart contract event used to display message when the transaction is executed. Smart contract here consist of some functions as follows:

getBalance it allows to parties to check the balance of an account by inserting public address.

arrived: is an alert function which show the mount of goods and address, whenever good arrive to destination. **arriveGood** it allows receiver records all detail of goods on blockchain once goods are received. Receiver check whether the items are receive are match with those are shipped. if they match, an event is triggered that the item has

been received successfully otherwise it fails.

sendGood sender records all information about goods as soon as goods departure. A mapping is used to map the tracking the goods like address quantity and item. Also goods is traced with real time location using sensor and other information such as time and address recorded into blockchain. An event will be triggered as item has been shipped.

checkSuccess This allows to any of parties to check the number of successfully shipment.

getDetails: Gets all the details for the package contract in a call and returns all of the public members of the contract.

3.8.2 Smart contract Validation

Remix IDE is browser-based compiler that enables users to compile contract with solidity code. It simulates EVM to execute smart contract and debug it. As executing smart contract on Ethereum costs some ether, Remix provides an environment to compile and debug it without any transaction fee. *Figure 3.5* Figure 3.5 represents a screenshot of the Remix IDE, where cost of executing a contract (goodTracking.sol), transaction (sendGood)and it's output are represented:

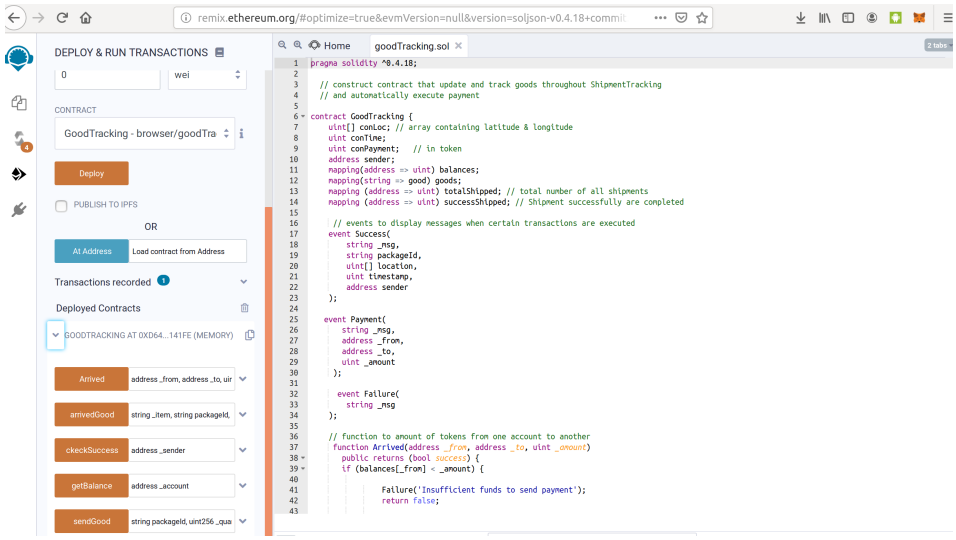

Figure 3.5: Remix IDE to deploy shipment tracking smart contract

To validate smart contract, the contract deployed with some tokens to administrator. Admin sets contract to ship good from supplier A to supplier B. When supplier A send good and record detail on blockchain. When supplier B receives good, records detail also on blockchain. When the shipping quantity is not matched, an alert is triggered and fail smart contract to deploy.

Figure 3.6: Recorded transactions after deploying smart contract

3.9 Semantic Blockchain development Implementation

There are limited work integrating semantic in blockchian specially smart contract.There exist only some work on Ethereum ontology which describe blockchain concepts using RDF schema and OWL. In this analysis used Ethereum ontology which describes basic semantic concepts related to specific schema such transaction or blocks. I used specific concepts to transaction that i used in this study.

EthOn represent different models such as block, account, smart contract concepts, instances and relation. Ethereum ontology covers concepts and relations between objects. As my goal is to extend ontology to support my favorite schema based on dataset by mapping to EthOn blocks or transactions model.

EthOn describe concepts, block, transaction in machine readable format that can facilitate reasoning which can be supported our semantic web service description Some possible ways to semantify blockchain:

Step 1. Create schema of blockchain or transaction , mapping of these data to RDF make using of vocabularies and remote procedure calls.

Step 2. Using blockchain data storage as input that will be mapped based on these data.

Step 3. Create properties of all blocks or transactions..

Step 4. Mapping all block properties to block dataset and generates output of all inputs.

Step 5. Copy output in turtle file where related prefixes are defined.

Step 6. Generate output based on SPARQL query and data produced on previous step.

3.9.1 Proof of concept

Dataset: The main part of our analysis is the dataset which can be exported by some block explorer such etherscan that is used in this research. Dataset will feed into our Ethereum ontology mapping which each attributes map to responding values of dataset. *Etherscan* is block explorer which allows us to search Ethereum blockchain for transaction, contract, blocks, addresses, price and other activities takes place on Ethereum. *Template* is in the form of RDF triple based on subject predict object that contains prefix of concepts, Ethereum ontology concepts, our schema with the type of concepts in ethOn and instances such as integer, hex binary, boolean and string.

```
ibb:{{tx_block_number}} ethon:containsTx tx:{{tx_hash}} .
tx:{{tx_hash}}  ethon:number
"{{tx_block_number}}"^^xsd:hexBinary .
tx:{{tx_hash}} ethon:blockCreationTime "{{tx_block_timestamp}}"^^xsd:dateTime
tx:{{tx_hash}}  ethon:from "{{tx_from}}
"^^xsd:hexBinary .
tx:{{tx_hash}}  ethon:to "{{tx_to}}
"^^xsd:hexBinary .
tx:{{tx_hash}}  ethon:address "{{tx_address}}"^^xsd:hexBinary  .
tx:{{tx_hash}}  ethon:txGasPrice  "{{tx_gas_price}}"^^xsd:integer .
tx:{{tx_hash}}  ethon:txGasUsed  "{{tx_gas_used}}"^^xsd:integer .
tx:{{tx_hash}}  ethon:ValueTx  "{{tx_value}}"^^xsd:integer .
```

Mapping function is used to tripleize and map all the properties of dataset, our schema and Ethereum ontology concept to each other.

```
function tripleize (data, template){
let properties = {
tx_hash: (data[0][0]),
tx_block_number: data[0][1],
tx_block_header: data[0][2],
tx_block_timestamp: data[0][3],
tx_from: data[0][4],
```

```
tx_to: data[0][5],
tx_address: data[0][6],
tx_gas_used: data[0][8],
tx_value: data[0][9],
        };
```

Tripleize Application used to feed relative dataset to Ethereum ontology concepts and our schema and integrated data.(See appendix for more detail) *Header* of output formed the prefix of out concepts.

```
@prefix ethon: <http://ethon.consensys.net/> .
@prefix ibb: <http://ethon.consensys.net/Block#> .
@prefix ibu: <http://ethon.consensys.net/Uncle#> .
@prefix ibs: <http://ethon.consensys.net/State#> .
@prefix tx: <http://ethon.consensys.net/Tx#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
```

Sparql query is an RDF query languages that is semantic query language for dataset-enable to present and manipulate data stored in Resource Description framework.(see appendix for full source code).

```
PREFIX : <http://ethon.consensys.net/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX ns: <http://www.w3.org/2003/06/sw-vocab-status/ns#>
PREFIX v0: <http://ethon.consensys.net/v0/>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX xml: <http://www.w3.org/XML/1998/namespace>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT   ?class ?p ?o
WHERE
{
?class ?p  ?o

}
```

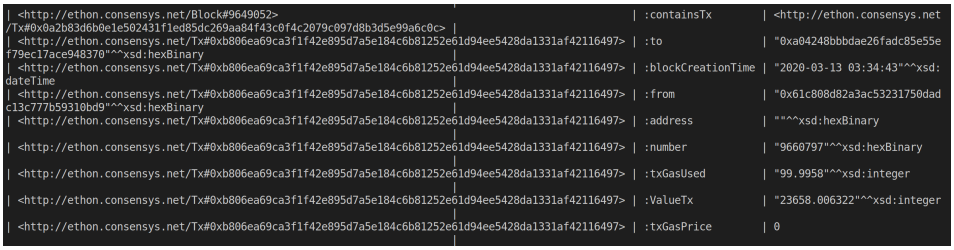

Figure 3.7: Final output of a transaction based on SPQRL query

Chapter 4

Conclusion

In this work, we attempted to analyze smart contract and their integration with semantic web data licesing. To achieve this goal, first we tried to analyze blockchain where smart contract reside on it as program code. Afterwards, we showed that how blockchain could be applied as computational paradigm for semantic web and linked data. Then we focused on bitcoin, Etheruem as main usage of blockahin and related vulnerabilities in Ethereum. Furthermore, we have presented the first initial effort to use this construct in useful ways by extending blockchain with semantic web in supply chain management. As the main goal of this work is to show ssemantic web principle can be use in making smarter smart contract and blockchain network. First, we started presenting our work done in ontology, Ethereum ontology (EthOn) describes blockchain structure and related information such as classes, object properties and etc. Later, we used this concept as RDF triple format which will feed by dataset and will lead into representing our data based on query.

Appendix A

Smart Contract code in solidity

```
pragma solidity ^0.4.18;

// construct contract that update and track goods throughout ShipmentTracking
// and automatically execute payment

contract GoodTracking {
uint[] conLoc; // array containing latitude & longitude
uint conTime;
uint conPayment;   // in token
address sender;
mapping(address => uint) balances;
mapping(string => good) goods;
mapping (address => uint) totalShipped; // total number of all shipments
mapping (address => uint) successShipped; // Shipment successfully are completed

// events to display messages when certain transactions are executed
event Success(
string _msg,
string packageId,
uint[] location,
uint timestamp,
address sender
);

event Payment(
string _msg,
address _from,
address _to,
uint _amount
);

event Failure(
```

```
string _msg
);

// function to amount of tokens from one account to another
function arrivedToken(address _from, address _to, uint _amount)
public returns (bool success) {
if (balances[_from] < _amount) {

Failure('Insufficient funds to send payment');
return false;

}
balances[_from] -= _amount;
balances[_to] += _amount;
Payment('Payment sent', _from, _to, _amount);
return true;
}

// function to show token balance
function getBalance(address _account) public returns (uint _balance) {

return balances[_account];
}

struct good{
string item;
uint  quantity;
uint[] location;
uint  timestamp;
address sender;
}

//function for displaying details of sent shipment

function sendGood(
string  packageId,
uint  _quanitity,
uint[]  _location,
string  _item,
uint _departure
)public returns(bool success){

goods[packageId].item = _item;
goods[packageId].quantity = _quanitity;
goods[packageId].location = _location;
Success('Item shipped', packageId,  _location,
_departure = now, msg.sender);
return true;
}
```

```
// function for displaying details of received shipment

function arrivedGood(
string _item,
string packageId,
uint _quantity,
uint[] _location,
uint _arrival
) public returns (bool success)  {

if (sha3(goods[packageId].item) == sha3(_item) && goods[packageId].quantity == _quant
Success('Item shipped', packageId, _location, _arrival = now, msg.sender);
}

if( _arrival <= _arrival + conTime && _location[0] == conLoc[0] &&
_location[1] == conLoc[1]){

arrivedToken(sender, goods[packageId].sender, conPayment);
} else {
Failure('Payment not triggered as criteria not met');
}

return true;
}
// Gets all the details for each shipment

function getDetails(string packageId)
public constant returns( string, uint, uint[],uint,  address)

{
return (
goods[packageId].item,
goods[packageId].quantity,
goods[packageId].location,
goods[packageId].timestamp,
goods[packageId].sender
);
}

// function to display number of successfully shipments and total shipments
function ckeckSuccess(address _sender) public returns (uint, uint)

    {
        return (successShipped[_sender], totalShipped[_sender]);
      }

}
```

Appendix B

Tripleize application

```
const fs = require('fs');
const parse = require('csv-parse');
const readline = require('readline');
const Handlebars = require('handlebars');
const Stream = require('stream');

fs.readFile('../templates/transactions.tmp', 'utf8', function (err,data) {
if (err) {
throw new Error(err);
} else {
rendering(data);
}

});

function rendering(source) {
const template = Handlebars.compile(source);
const readStream = fs.createReadStream('../data/input/transactions.csv');
//const writeStream = fs.createWriteStream( "../data/output/blocks.ttl", { encoding:
let rl = readline.createInterface({
input: readStream,
output: new Stream,
terminal: false,
historySize: 0
});

rl.on('line', function(line) {
// process line here
parse(line, (err, data) => {
if(err) {
console.log("Error: " + err);
throw new Error(err);
```

```
} else {
tripleize(data, template);
// writeStream.write(line);
}
});

}).on ('close', function() {
console.log('Good done!');
process.exit(0);
});

}

function tripleize (data, template){

let properties ={
tx_hash: (data[0][0]),
tx_block_number: data[0][1],
tx_block_header: data[0][2],
tx_block_timestamp: data[0][3],
tx_from: data[0][4],
tx_to: data[0][5],
tx_address: data[0][6],
tx_gas_used: data[0][8],
tx_value: data[0][9],
};

let result = template(properties);
var header = fs.readFileSync('../data/input/headers.ttl', 'utf8');
//console.log(header +result);
fs.writeFileSync('../data/output/output.ttl', header + result, { encoding: "utf8"});
}
```

Bibliography

[Ada] Bc approach scm in bim environment. In Adam Fitria wijaya, T. H.-h. and Taysheng, J., editors, *Proceedings of the 24th International Conference of the Association for Computer-Aided Architectural Design Research in Asia*. ACM.

[2] Alfonso Panarello, Nachiket Tapas, G. M. F. L. and Puliafito, A. (2018). Blockchain and iot integration: A systematic survey.

[3] Allan Third, J. D. (2017). Linked data indexing of distributed ledgers. In *WWW 17 Companion Proceedings of the 26th International Conference on World Wide Web Companion*, volume 8.

[4] Antono Lucas Soares, A. L. A. and de Sousa, J. P. (1999). Distributed planning and control systems for thevirtual enterprise: organizational requirementsand development life-cycle. page 18.

[Buterin] Buterin, V. A next generation smart contract and decentralized application platform.

[6] Christidis, K. and Sevetsikioti, M. (2016). Blockchains and smart contracts forthe internet of things. page 12.

[7] CribÃďck, K. (2018). Micro payment, viable technical platforms and models for a bank to provide payments on micro amounts.

[8] Dujak, D. and Sajter, D. (2018). *Blockchain Applications in Supply Chain*.

[9] Edvard Tijan, Sasha Aksentijevic, K. I. and Jardas, M. (2019). Blockchain technology implementation in logistics. page 13.

[10] Guido Geertsa, D. L. (2014). The eaglet ontology for highly visible supply chains. page 53.

[11] Guo, Y. and Liang, C. (2016). Blockchain application and outlook in the banking industry. page 12.

[12] Hamza Baqa, Nguyen B. Truongy, N. C. G. M. L. F. L. G. (2019). Semantic smart contracts for blockchain-based services in the internet of things. 5.

[13] Hanqing Wu, Jiannong Cao, Y. Y. C. L. T. S. J. B. T. Y. L. X. W. Y. D. (2019). Data management in supply chain using blockchain: Challenges and a case study. page 9.

[14] Ilya Grishchenko, M. M. and Schneidewind, C. (2018). *A Semantic Framework for the Security Analysis of Ethereum smart contracts*.

[15] Kim, H. M. and Laskowski, M. (2018). Towards an ontology-driven blockchain design for supply chain provenance.

[16] Kim, J.-S. and Shin, N. (2019). The impact of blockchain technology application on supply chain partnership and performance. page 17.

[17] Konstantinos Christidis, M. D. (2016). Blockchains and smart contracts for the internet of things.

[18] Kouhizadeh, M. and Sarkis, J. (2018). Blockchain practices, potentials, and perspectives in greening supply chains. page 16.

[19] Law, A. (2018). Smart contracts and their application in supply chain management. page 89.

[20] Lin and Harding (2007). A manufacturing system engineering ontology model on the semantic web for inter-enterprise collaboration. page 32.

[21] Lin, A. M. W. and Madni, C. (2000). Ideon: An extensible ontology for designing, integrating, and managing collaborative distributed enterprises. page 14.

[22] Markos, K. (2012). Indexes for blockchain data.

[23] Md Nazmul Islam, Vinay Patil, S. K. (2018). On ic traceability via blockchain. page 4.

[Michele Ruta and Sciascio] Michele Ruta, Floriano Scioscia, S. I. G. C. A. P. and Sciascio, E. D. A blockchain infrastructure for the semantic web of things.

[25] Mike Uschold, Martin King, S. M. Y. Z. (1996). Enterprise ontology. page 57.

[26] Mirek Sopek, Przemyslaw Gradzki, W. K. D. K. R. T. R. T. (2018). Www 18 companion proceedings of the the web conference 2018. In *GraphChain âĂŞ A Distributed Database with Explicit Semantics and Chained RDF Graphs*, volume 8.

[27] Ozsu, T. and Valduriez, P. (1999). Principles of distributed database systems.

[28] Pablo Lamela Seijas, Simon Thompson, D. M. (2016). Scripting smart contracts for distributed ledger technology. 30.

[29] Sharples, M. and Domingue, J. (2016). *Adaptive and Adaptable Learning*.

[30] Sukrit Kalra, Seep Goel, M. D. S. S. (2018). Zeus: Analyzing safety of smart contracts. 15.

[31] Tonci Grubic, I.-S. F. (2010). Supply chain ontology: Review, analysis and synthesis. page 11.

[Ugarte] Ugarte, H. A more pragmatic web 3.0: Linked blockchain data.

[33] Umar Rashid, Allan Third, J. D. (2018). Web service for semantic negotiation of smart contracts. 6.

[34] Ureten1, S. and Ilter, K. (2007). Supply chain management ontology: Toward an ontology-based scm model.

[35] Wesley Egbertsen, Gerdinand Hardeman, M. v. d. H. G. v. d. K. and van Rijsewijk, A. (2016). Replacing paper contracts with ethereum smart contracts. 35.

[36] Wood, D. G. (2012). Ethereum: A secure decentralized generalized transaction ledger. page 32.

[37] WÃűhrer, M. and Zdun, U. (2018). Smart contracts: Security patterns in the ethereum ecosystem and solidity. 7.

[38] Yan Ye, Dong Yang, Z. J. L. T. (2008). An ontology-based architecture for implementing semantic integration of supply chain management. page 19.