



Università del Piemonte Orientale

Dipartimento di Scienze e Innovazione Tecnologica

Corso di Laurea in Informatica

Relazione per la prova finale

Blockchain per l'autoconsumo energetico nel caso del fotovoltaico condominiale

Tutore interno:

Prof.ssa Lavinia Egidi

Candidato:

Tommaso Pessina

Anno Accademico 2018/2019

The main advantage of blockchain technology is supposed to be that it's more secure, but new technologies are generally hard for people to trust, and this paradox can't really be avoided.

- Vitalik Buterin

Indice

1 Abstract.....	6
2 Introduzione.....	7
3 Scopo del progetto proposto.....	8
3.1 Contesto del progetto.....	8
4 Specifiche del progetto.....	12
4.1 Specifiche fornite dall'azienda.....	12
4.2 Specifiche in aggiunta.....	12
4.3 Funzionalità del sistema.....	13
4.3.1 Calcolo del coefficiente di ripartizione degli autoconsumi.....	13
4.3.2 Indicizzazione e mantenimento dati.....	14
4.3.3 Gestione basata su ruoli.....	15
4.3.4 Comunicazione con i contratti in Blockchain.....	15
5 Strumenti utilizzati.....	16
5.1 Introduzione alla Blockchain.....	17
5.1.1 Tipologie di Blockchain.....	18
5.1.2 Blockchain “Ethereum” nel dettaglio.....	18
6 Lavoro svolto.....	22
6.1 Analisi della richiesta.....	23
6.2 Realizzazione del progetto.....	25
6.2.1 Sviluppo degli Smart Contract.....	25
6.2.2 Funzionalità dei principali Smart Contract.....	26
6.2.3 Funzionalità degli script Python.....	28
6.2.4 Calcolo in Blockchain del coefficiente di ripartizione.....	29
7 Database e Blockchain a confronto.....	30
8 Conclusione e sviluppi futuri.....	34
9 Bibliografia.....	36
10 Ringraziamenti.....	38
11 Appendice 1: Documento tecnico.....	39

Indice delle figure

Figura 1: Configurazione di autoconsumo commerciale.....	9
Figura 2: Configurazione condominiale del caso di studio di interesse per lo sviluppo del software.....	10
Figura 3: Schema di coordinamento tra gli attori della comunità.....	11
Figura 4: (A) Centralizzata (B) Decentralizzata.....	17
Figura 5: Rilevamento corruzione blocco.....	20
Figura 6: Funzionamento funzioni di hash.....	20
Figura 7: Porzione del sistema responsabile della produzione dei dati. .	22
Figura 8: Architettura del sistema con comunicazione illusoria.....	24
Figura 9: Procedimento di inserimento di un dato in Blockchain (BC)...	29
Figura 10: Crescita della dimensione della Blockchain di Ethereum.....	32

Indice delle tabelle

Tabella 1: Confronto Blockchain e Database.....	30
Tabella 2: Confronto svantaggi di Blockchain e Database.....	31
Tabella 3: Migliori utilizzi di Blockchain e Database a confronto.....	34

1 Abstract

Lo scopo di questo studio guidato era analizzare il caso del “fotovoltaico in condominio”, applicato nel contesto dell’autoconsumo energetico all’interno di una *energy community*.

Per valutare l’andamento dell’autoconsumo è stato definito un coefficiente di ripartizione per quest’ultimo, che dovrà essere calcolato nuovamente ad ogni specifico istante di tempo.

Tutti i dati prodotti da questo sistema, non vengono salvati in una base di dati, ma in Blockchain. Questa tecnologia, utilizzata in concomitanza agli Smart Contract, agisce come un vero e proprio database.

In questa relazione verrà esaminato come è stato sviluppato l’intero sistema, fornendo i dovuti accenni riguardo le tecnologie ed i software utilizzati, come: Geth, Ethereum, Solidity ed altro ancora.

Infine, è stato scelto di effettuare una ricerca basata sul confronto di database e Blockchain, in modo da valutare se la seconda porti reali benefici.

2 Introduzione

Le tecnologie Blockchain e Smart Contract stanno ottenendo molte attenzioni in questi ultimi anni.

L'interesse verso queste tecnologie è dovuto principalmente alla loro versatilità di utilizzo. Essi infatti spaziano dal mantenimento dati alla gestione automatica di pagamenti.

Numerose tra le più grandi aziende hanno deciso di utilizzare una Blockchain per le più svariate motivazioni e i più vari impieghi. Ad esempio, in Ottobre 2019 il colosso svedese Ikea ha segnato un primato mondiale. Ikea Islanda ha infatti emesso una fattura elettronica basata su Blockchain, che è stata saldata in automatico grazie ad uno Smart Contract [1].

Occorre specificare che sono gli Smart Contract che rendono possibili le operazioni sulla Blockchain, quindi per lo svolgimento di questo progetto sono sempre stati considerati in coppia.

In questa relazione verranno esaminati i dettagli relativi al caso di studio, le funzionalità del sistema sviluppato e si cercherà di fornire le dovute motivazioni relative all'utilizzo di Blockchain rispetto ad una base di dati. Si è infatti deciso di effettuare una ricerca per confrontare i sistemi basati su Blockchain rispetto a quelli basati su Database, in modo da comprendere meglio i vantaggi che un sistema con Blockchain può fornire.

3 Scopo del progetto proposto

Il progetto svolto durante la mia attività di studio guidato è stato estratto dalla collaborazione scientifica in atto tra l'ateneo e l'azienda "Ricerca sul Sistema Energetico S.p.A." (RSE), con sede a Milano. Si tratta di un'azienda controllata dal Gestore dei Servizi Energetici, incaricata dello sviluppo di attività di ricerca nel settore elettro-energetico, con particolare riferimento ai progetti in territorio nazionale.

Lo scopo del progetto era la realizzazione di un sistema, basato su Blockchain e Smart Contract, volto alla ricezione e successivo mantenimento di dati.

In questo capitolo verrà esaminato il contesto generale del progetto.

3.1 Contesto del progetto

L'ambito di applicazione del sistema è quello delle "Energy Community", ovvero gruppi di soggetti attivi nella generazione rinnovabile, stoccaggio, vendita e consumo dell'energia prodotta attraverso pannelli fotovoltaici, generando così uno sfruttamento dell'autoconsumo collettivo [2].

Il caso di particolare interesse, su cui l'azienda ha scelto di basare lo svolgimento del progetto, è stato quello del "fotovoltaico in condominio" che, con le regole vigenti in Italia, rappresenta uno schema al momento inapplicabile. Attualmente, per quanto riguarda l'impianto fotovoltaico, è consentito solamente che possa alimentare i carichi generali di una struttura (ad esempio ascensore e luci delle scale).

L'azienda ha effettuato molteplici ricerche volte alla valutazione di differenti configurazioni di autoconsumo collettivo. La soluzione scelta da RSE come riferimento per lo sviluppo di questo progetto è la *configurazione di scambio virtuale* o “*commerciale*”, mostrata in Figura 1.

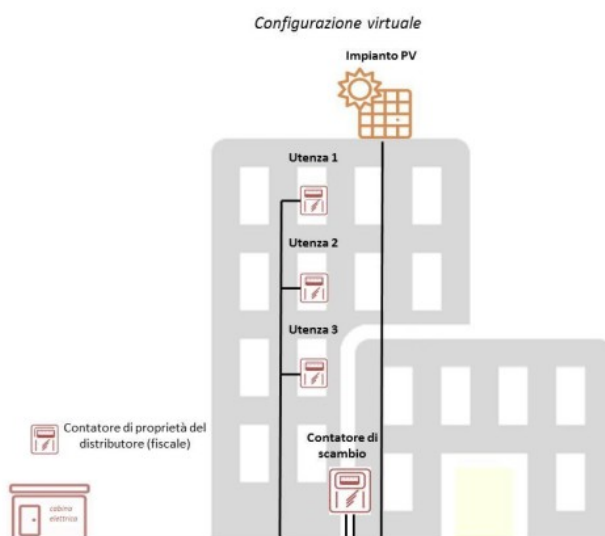


Figura 1: Configurazione di autoconsumo commerciale

Questa configurazione presenta vari benefici, tra cui:

- la possibilità di libera scelta del proprio fornitore per ciascun condominio;
- la possibilità di aderire o meno alla comunità;
- le condizioni tecniche di accesso alla rete pubblica rimangono invariate sia che il cliente accenda o meno alla comunità;
- la gestione dei clienti morosi rimanere in capo al fornitore dell'energia.

Per quanto riguarda la configurazione condominiale, oggetto dell'analisi svolta dall'azienda, essa può essere riassunta come in Figura 2.

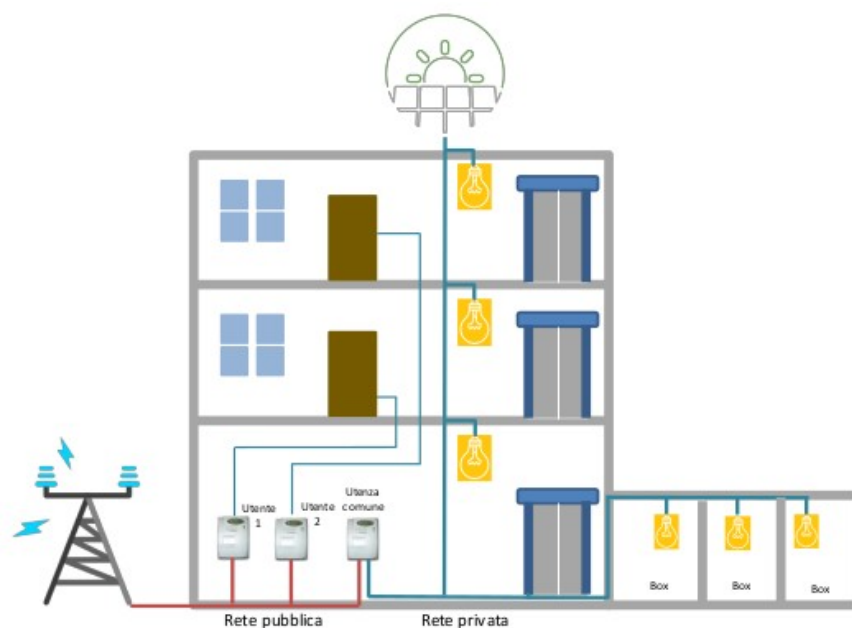


Figura 2: Configurazione condominiale del caso di studio di interesse per lo sviluppo del software

La ricerca ha inoltre tenuto conto del fatto che da parte della comunità condominiale non sia previsto alcun tipo di investimento, dato che si fa uso dei presenti contatori elettronici 2G. In particolare, come definito all'interno del documento fornito dall'azienda [2]:

“contatori elettronici 2G, come definiti dalla delibera ARERA 87/2016/R/eel, che tramite la “chain 1”, consentono di ottenere misure quartodinarie validate entro 24 ore, e tramite la “chain 2” rendono disponibili dati non validati near real time.”

Lo schema di riferimento adottato per la ripartizione dell'energia fotovoltaica, fornito da RSE, è mostrato in Figura3.

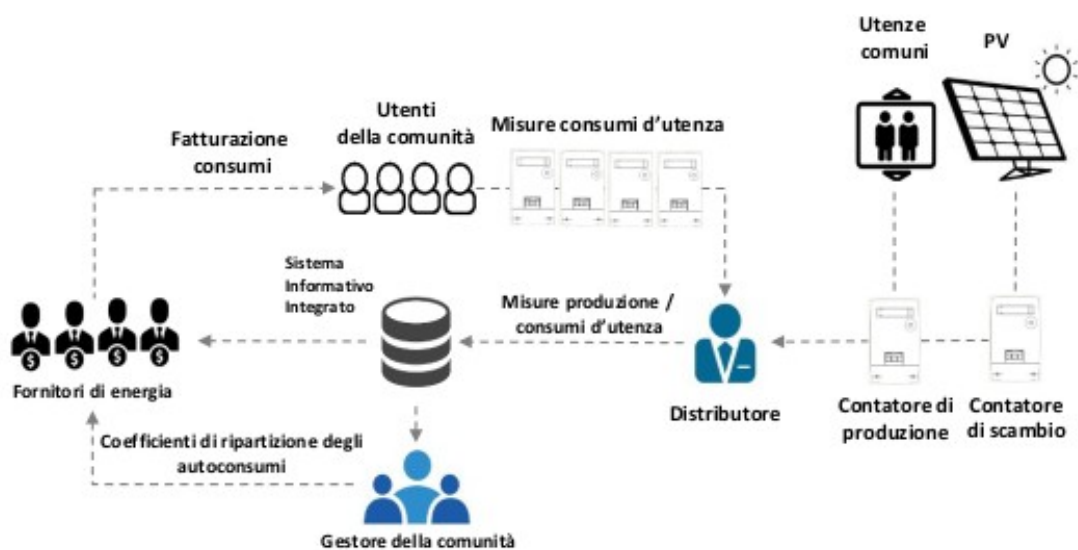


Figura 3: Schema di coordinamento tra gli attori della comunità.

Facendo riferimento alla Figura 3, sopra mostrata, il sistema sviluppato consiste nel “Sistema Informativo Integrato”.

4 Specifiche del progetto

Le specifiche necessarie allo svolgimento di questo progetto sono state estratte da un documento tecnico fornito dall'azienda.

4.1 Specifiche fornite dall'azienda

Il punto chiave del progetto era fare in modo che tutte le operazioni venissero eseguite all'interno della Blockchain.

L'azienda ha fornito un file di esempio contenente vari dati tipici che il sistema doveva essere in grado di interpretare correttamente, su cui si è basato l'intero sviluppo.

La Blockchain doveva essere in grado di mantenere i dati relativi all'energia consumata da ogni singola unità immobiliare, l'energia consumata dalle utenze comuni e l'energia prodotta dal pannello fotovoltaico.

Attraverso il mantenimento di questi dati è possibile calcolare il coefficiente di ripartizione degli autoconsumi, la cui formula è stata fornita direttamente da RSE.

Infine, l'azienda ha specificato come i dati venissero comunicati dai contatori, ovvero tramite il protocollo MQTT.

4.2 Specifiche in aggiunta

In aggiunta alle specifiche fornite dall'azienda, si è scelto di inserire ulteriori funzionalità volte alla semplificazione dell'utilizzo del sistema. L'interazione con uno Smart Contract, per sua natura, deve avvenire tramite riga di comando. Perciò si è scelto di sviluppare alcuni

programmi Python con la funzionalità di interfaccia, rendendo possibile un'interazione tra utente e Smart Contract presente in Blockchain. Questa scelta rende inoltre più semplice l'esecuzione automatica di determinate funzioni.

Dall'azienda non è stata fornita alcuna preferenza riguardo il metodo di mantenimento dei dati in Blockchain, per la loro visualizzazione o per la loro indicizzazione. Si è quindi scelto di procedere nel modo che si dimostrasse più comodo per un utente e più efficiente a lato computazionale.

4.3 Funzionalità del sistema

In questo capitolo verranno esaminate tutte le funzionalità richieste dal sistema sviluppato.

4.3.1 Calcolo del coefficiente di ripartizione degli autoconsumi

La funzionalità di maggior interesse, richiesta dal sistema, è il calcolo del coefficiente di ripartizione degli autoconsumi ed il suo mantenimento in Blockchain ad ogni dato istante di tempo.

Questo coefficiente consiste nel rapporto tra l'energia "virtualmente" prodotta tramite il pannello fotovoltaico e la somma dell'energia complessivamente utilizzata dalle varie utenze private, con particolare attenzione al caso in cui l'energia prodotta sia superiore a quella utilizzata in cui il coefficiente viene fissato ad 1.

La formula per il relativo calcolo è stata fornita dall'azienda ed è la seguente:

$$Cr_i = \frac{E_{pv_i}}{\sum_{j=1}^N Cut_{ij}} \text{ con } Cr_i = 1 \text{ se } E_{pv_i} > \sum_{j=1}^N Cut_{ij}$$

dove:

- i indica ciascun periodo temporale minimo di misura (della durata, ad esempio, di 15 minuti);
- N è il numero delle utenze private condominiali che partecipano allo schema di autoconsumo collettivo;
- Cr_i è il coefficiente di ripartizione, che specifica quale quota dei consumi complessivi delle utenze private è considerata soddisfatta dalla “produzione virtuale” dell’impianto fotovoltaico nel periodo temporale i ;
- E_{pv_i} è l’energia misurata sul contatore di scambio con la rete del condominio nel periodo temporale i , corrispondente alla “produzione virtuale” dell’impianto fotovoltaico;
- Cut_{ij} è il consumo dell’utenza privata j , rilevato dal rispettivo contatore nel periodo temporale i .

4.3.2 Indicizzazione e mantenimento dati

Per rendere possibile il calcolo del coefficiente di ripartizione è necessario permettere al sistema di mantenere i dati ed in seguito di visualizzarli.

Gli strumenti volti al mantenimento dei dati sono la Blockchain e gli Smart Contract. Questi ultimi permettono di essere scritti come dei veri e propri programmi e di esportare verso un’interfaccia le funzioni definite al loro interno. In questo caso si tratta di esportare le funzioni per l’inserimento e la visualizzazione dei dati.

La Blockchain, d’altro canto, è la tecnologia in cui gli Smart Contract operano. Per rendere quindi possibile la funzionalità sopra descritta,

Smart Contract e Blockchain sono da intendere come veri e propri oggetti sinergici.

4.3.3 Gestione basata su ruoli

Essendo uno Smart Contract un programma, i controlli che un singolo può effettuare sono limitati. Infatti, chiunque abbia accesso ad uno di essi, può eseguire le funzioni definite al suo interno.

L'azienda aveva pensato di diversificare la tipologia dei dati differenziando il loro inserimento (o visualizzazione) in base a ruoli. Il caso di studio ha però dimostrato più efficiente differenziare i ruoli in base al tipo di funzione che si desidera eseguire.

L'idea è infatti quella di verificare se chi vuole effettuare una data funzione, possieda correttamente un determinato ruolo. L'esecuzione della funzione richiesta sarà quindi consentita solamente se si possiedono i giusti permessi.

4.3.4 Comunicazione con i contratti in Blockchain

Per rendere questa funzionalità più facile per un utente, si è scelto di sviluppare alcuni script Python volti alla comunicazione con gli Smart Contract presenti in Blockchain. In questo modo non è necessaria, per un utente, la conoscenza approfondita di determinati software.

Ogni script Python richiede in input un determinato numero di parametri. Ognuno di essi richiede l'indirizzo del contratto schierato (o *deployed*) in Blockchain a cui fa riferimento. In aggiunta a questo, i parametri variano in base alla tipologia di funzione che si richiederà allo Smart Contract.

5 Strumenti utilizzati

I principali strumenti necessari per lo svolgimento di questo progetto sono stati Blockchain e Smart Contract. Per quanto riguarda la Blockchain, l'azienda ha proposto quella di Ethereum poiché è stata sviluppata appositamente per interagire con gli Smart Contract. Per quest'ultimi invece RSE ha proposto il linguaggio di programmazione Solidity, anch'esso creato appositamente per la loro scrittura.

Altri software che si sono dimostrati necessari durante lo sviluppo del progetto, sono stati Truffle e Geth. Il primo consiste in una software suite open-source che permette la compilazione, lo schieramento e l'esecuzione dei contratti in Blockchain. Esso si è dimostrato particolarmente utile per effettuare debugging e per schierare i contratti in Blockchain. Geth invece sta per Go ETHereum ed è una delle tre originali implementazioni del protocollo Ethereum, scritta in Go ed è open-source. Esso si è dimostrato di particolare importanza per l'interfacciamento con la Blockchain, fornendo tutte le funzioni necessarie alla comunicazione con essa.

Infine, la scelta verso il linguaggio di programmazione "Python" è dovuta alla sua leggerezza, versatilità e facilità di comprensione. Per il corretto funzionamento degli script sviluppati, sono necessarie due librerie: "web3.py" per poter interfacciarsi con la Blockchain e "paho-mqtt.py" per poter comunicare con un Broker.

5.1 Introduzione alla Blockchain

Il termine Blockchain letteralmente significa catena di blocchi, ed è esattamente questo. I primi lavori su una Blockchain sono stati descritti nel 1991 da Stuart Haber e W. Scott Stornetta [3].

Spesso con il termine Blockchain si pensa a Bitcoin, ma si tratta solamente di una tipologia specifica di questa.

Uno dei maggiori punti d'interesse delle Blockchain è la loro natura decentralizzata. Infatti, non è presente un'unica entità a cui devono essere demandate tutte le richieste.

La Blockchain è fortemente decentralizzata poiché ogni nodo (o *peer*) facente parte di questa, risulta coinvolto attivamente.

In Figura 4 è possibile comprendere in modo chiaro come sia differente l'architettura tra le due tipologie di configurazione.

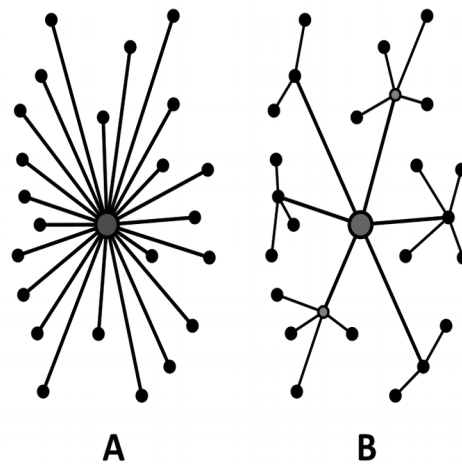


Figura 4: (A) Centralizzata
(B) Decentralizzata

Sempre nel contesto delle Blockchain, la decentralizzazione è presente in ogni operazione. Per esempio: nel caso venga effettuata una transazione

su uno Smart Contract, essa dovrà essere *mined*, ovvero dovrà essere controllata, confermata e resa definitiva da tutti i *peer* presenti nella catena.

5.1.1 Tipologie di Blockchain

Nonostante la Blockchain sia di natura decentralizzata, esistono tipi differenti di configurazione [4].

Le tre principali tipologie sono:

- pubblica: si tratta di una blockchain senza nessun controllo su chi può entrare a farne parte;
- privata (o *permissioned*): si tratta di una Blockchain a cui può accedere solamente chi è autorizzato. Risulta però necessario avere una sorta di meccanismo di controllo degli accessi. Da notare però è come questo meccanismo renda necessaria la presenza di un'entità volta ad eseguire questa funzione. Ciò rende inevitabilmente la Blockchain centralizzata;
- a consorzio: si tratta di una Blockchain a metà tra una pubblica ed una privata. Tendenzialmente si tratta di Blockchain proprietarie.

La scelta su quale di queste sia la migliore varia in base all'applicazione e in base a correnti di pensiero differenti.

Per lo sviluppo di questo progetto è stata scelta la prima.

5.1.2 Blockchain “Ethereum” nel dettaglio

La Blockchain “Ethereum” consiste in una piattaforma globale ed open-source per applicazioni decentralizzate e come definito sul sito ufficiale:

“Su Ethereum puoi scrivere codice che controlla valori digitali, funziona esattamente come programmato ed è accessibile in qualsiasi parte del mondo.” [5]

I primi riferimenti a Ethereum si ebbero nel Novembre del 2013 quando Vitalik Buterin, Mihai Alisie, Anthony Di Iorio e Charles Hoskinson [6] pubblicarono un documento ufficiale che descriveva per la prima volta questa tecnologia.

Ethereum fornisce diverse reti con cui comunicare, tra cui alcune private. Per lo sviluppo di questo progetto è stata creata un Blockchain pubblica.

I blocchi di questa catena sono costituiti principalmente da:

- header;
- nonce;
- hash del blocco stesso;
- hash del blocco precedente;
- timestamp;
- dati;
- merkle root & merkle tree.

Tra essi risultano di particolare importanza l'hash del blocco precedente e il timestamp. Il primo si tratta del punto forte della Blockchain. Infatti, se un malintenzionato volesse in qualche modo alterare un determinato blocco, la modifica sarebbe evidente poiché questa catena di hash verrebbe corrotta, come mostrato in Figura 5.



Figura 5: Rilevamento corruzione blocco

Un attaccante esperto potrebbe però scegliere di modificare tutti gli altri nodi della catena (nello specifico l'hash). Il problema si pone quando arriverà al primo blocco. Esso, comunemente chiamato blocco genesi, viene salvato su un differente mezzo (rispetto alla Blockchain) in modo che, se venisse alterata la copia in Blockchain, la modifica venga riconosciuta e si possa procedere alla sua rimozione.

Un'assunzione che viene fatta, è come le funzioni hash operano. Una loro caratteristica molto interessante in questo contesto, è come una minima modifica su un determinato input abbia una ripercussione notevole sull'output (come mostrato in Figura 6).

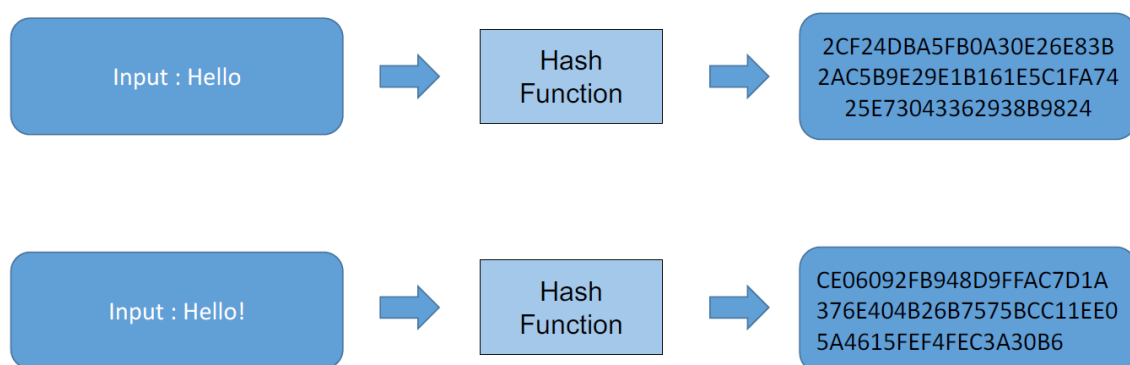


Figura 6: Funzionamento funzioni di hash

L'hash del blocco stesso si mostra utile invece, come si evidenzia nella Figura 5, per il susseguirsi di questa catena.

Il campo timestamp si è invece dimostrato utile nell'inserimento in Blockchain dell' i -esimo coefficiente di ripartizione. Infatti, dopo essere stato calcolato, viene estratto il timestamp del blocco (in formato UNIX) per poter riconoscere l'istante di tempo a cui appartiene il coefficiente appena calcolato.

6 Lavoro svolto

In questo capitolo verrà esaminato quello che è stato il mio lavoro durante l'attività di tirocinio.

Alle spalle di questo progetto è presente un sistema volto alla generazione dei dati energetici che vengono comunicati tramite il protocollo MQTT. Essa può essere riassunta come in Figura 7.

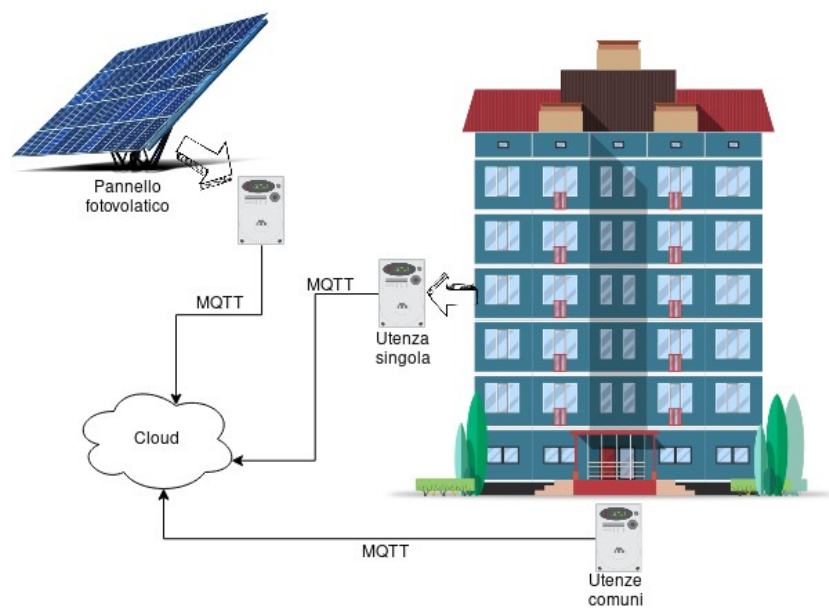


Figura 7: Porzione del sistema responsabile della produzione dei dati

Lo sviluppo di questo progetto è avvenuto in autonomia sul computer personale.

L'azienda aveva messo a disposizione una macchina remota, a cui era possibile accedere tramite SSH fornendo una chiave privata, il cui scopo era installare il sistema e testarlo. Purtroppo, un susseguirsi di eventi in

azienda ha reso questa operazione di complessa realizzazione. Si è infatti dimostrato necessario ricorrere all'utilizzo di Docker.

6.1 Analisi della richiesta

Il passo fondamentale è stato scegliere come organizzarne l'architettura, come ricevere, mantenere ed indicizzare i dati.

Per quanto riguarda l'organizzazione del sistema, sono state definite tre differenti entità. Nel dettaglio:

- Client: si tratta di colui che può effettuare le operazioni di inserimento o visualizzazione dei dati;
- Data Storage System(DSS): si tratta dell'entità responsabile del mantenimento dei dati;
- Role-Based Access Controll (RBAC): è il responsabile della gestione dei ruoli.

Per ognuna di queste entità è stato sviluppato uno Smart Contract.

L'obiettivo era ottenere una comunicazione illusoria tra queste entità, in modo da rendere le operazioni di verifica del tutto trasparenti.

L'architettura è rappresentata in Figura 8.

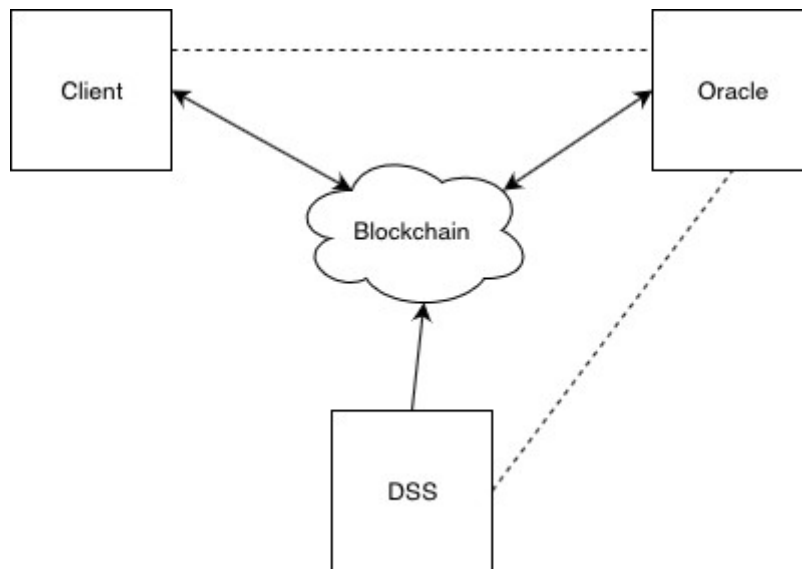


Figura 8: Architettura del sistema con comunicazione illusoria

Per quanto riguarda il recupero e il mantenimento dei dati, la richiesta prevedeva che fossero ricevuti tramite il protocollo MQTT e personalmente ho scelto il Broker “Mosquitto” per la sua leggerezza.

Come salvare, mantenere ed indicizzare i dati è stata una scelta basata su ricerche svolte da me personalmente poiché l’azienda non ha fornito nessuna preferenza. Si è scelto di organizzare i dati per mezzo di una struttura, essendo essa un tipo definito dal linguaggio Solidity. Essi vengono indicizzati tramite un identificatore ottenuto effettuando l’hash dei singoli campi del dato, poiché in questo modo non è possibile risalire ad alcuna informazione.

6.2 Realizzazione del progetto

Il progetto è stato svolto per la maggior parte in ambiente Linux, nello specifico Ubuntu. La realizzazione di questo software ha reso necessario alternare fasi di ricerca e sviluppo.

In questo capitolo verrà esaminato nel dettaglio quello che è stato il vero e proprio lavoro di sviluppo del software.

6.2.1 Sviluppo degli Smart Contract

Per il corretto funzionamento del sistema sono stati sviluppati sette Smart Contract. Nello specifico si tratta di: Client, DSS, RBAC, RBACinterface, DSSinterface, Roles ed Ownable. Nel dettaglio:

- Client, DSS, RBAC: si tratta degli Smart Contract corrispondenti alle entità principali;
- DSSinterface e RBACinterface: si tratta degli Smart Contract responsabili dell'interfacciamento tra le entità principali;
- Roles ed Ownable: si tratta di due Smart Contract con la finalità di esportare funzioni utili verso le entità principali, agendo come librerie.

Lo Smart Contract “Ownable” è stato creato per inserire il concetto di proprietà per ognuno di essi. È stato infatti definito un modo per identificare il proprietario del contratto, sfruttando una funzionalità del linguaggio Solidity, ovvero il *modifier*. Grazie ad esso è infatti possibile garantire che una funzione possa essere eseguita solamente dal legittimo proprietario del contratto.

Lo Smart Contract “Roles” invece esporta, principalmente verso l’ “RBAC”, tutte le funzioni necessarie alla corretta gestione dei ruoli. Esso

contiene infatti le funzioni per aggiungere, rimuovere e controllare se un dato indirizzo possiede uno specifico ruolo.

Gli Smart Contract “DSSinterface” e “RBACinterface”, funzionando da interfacce, contengono solamente i prototipi delle funzioni definite nel contratto a cui si riferiscono.

Infine, gli Smart Contract delle entità, sono i principali, ovvero quelli con cui si andrà ad interfacciarsi.

Per il loro sviluppo si è scelto di utilizzare un IDE online ampiamente usato in questo ambito: “REMIX”. Tramite esso è stato anche possibile testare il comportamento degli Smart Contract schierandoli in un ambiente virtuale (EVM, Ethereum Virtual Machine).

6.2.2 Funzionalità dei principali Smart Contract

Ogni principale Smart Contract ha una funzione ben definita, nello specifico:

- Client: è il contratto a cui occorre demandare l’inserimento e la visualizzazione di ogni tipo di dato;
- DSS: è il contratto responsabile del mantenimento dei dati. L’interfacciamento diretto ad esso non è permesso, salvo per il legittimo proprietario. Questo Smart Contract esporta le funzioni per inserire e visualizzare i dati verso il Client;
- RBAC: è il contratto responsabile del settaggio, mantenimento e controllo dei ruoli. L’interfacciamento ad esso è permesso solamente ad una persona fidata che corrisponderà al legittimo proprietario di esso. Questo contratto esporta le funzioni di controllo del possesso dei ruoli verso il DSS.

Per garantire la corretta comunicazione tra questi contratti è necessario inserire l'indirizzo di colui che eseguirà la richiesta all'interno (quindi nel codice) di quello che effettua la richiesta. Per rendere possibile questa operazione sono state definite apposite funzioni per l'inserimento dell'indirizzo all'interno dello Smart Contract.

Nel caso in cui si cercasse di effettuare una richiesta, senza aver settato correttamente questo indirizzo, si incorrerebbe in un errore. Esso però è stato pensato in modo da non fornire alcuna informazione circa la sua motivazione.

Lo Smart Contract "Client", invece, è stato sviluppato in modo da agire indipendentemente dalla funzione richiesta, in modo tale da non avere differenti tipi di esso. Nello specifico, la possibilità di eseguire o meno una determinata funzione è gestita tramite i ruoli.

Quando un "Client" decide di effettuare un'operazione, la demanderà al "DSS" che si occuperà di richiedere al "RBAC" se colui che ha effettuato la richiesta ha i corretti permessi.

Nel dettaglio sono stati definiti i seguenti ruoli:

- permesso di leggere i dati relativi all'energia presenti in Blockchain;
- permesso di inserire dati in Blockchain che si distinguono tra:
 - dati relativi all'energia utilizzata dalle utenze comuni;
 - dati relativi all'energia prodotta;
 - dati relativi all'energia utilizzata dalle singole utenze;
- permesso di admin che consente di:
 - visualizzare e inserire in Blockchain il coefficiente di ripartizione degli autoconsumi;

- resettare i contatori presenti all'interno dello Smart Contract.

La loro assegnazione avviene mantenendo una coppia chiave-valore all'interno dello Smart Contract "RBAC", assegnando ad un indirizzo un determinato numero che coincide con un dato ruolo.

6.2.3 Funzionalità degli script Python

Per poter inserire e visualizzare semplicemente i dati in Blockchain sono stati sviluppati vari script Python. Nello specifico sono state definite le funzioni per:

- assegnare un ruolo ad un dato indirizzo di account;
- rimuovere un ruolo ad un dato indirizzo di account;
- inserire in modo automatico i dati in Blockchain;
- visualizzare tutti i dati presenti in Blockchain;
- visualizzare ed inserire in Blockchain il coefficiente di ripartizione;
- effettuare il reset dei contatori necessari al calcolo del coefficiente di ripartizione.

Questi script comunicano con la Blockchain attraverso un *end-point* (indirizzo IP e porta) tramite HTTP. In base al tipo di Smart Contract con cui desidera comunicare lo script, dovrà contenere al suo interno la corretta ABI (Application Binary Interface) che consiste in un JSON contenente le funzioni che il contratto ha definito al suo interno. Questa informazione è reperibile dopo aver compilato il suddetto contratto.

Nello specifico dell'inserimento dei dati in Blockchain, si è dimostrato necessario sviluppare uno script che si occupasse della ricezione dei dati tramite MQTT. All'arrivo di un dato questo script si preoccupa di eseguire

un ulteriore programma il cui scopo è l'inserimento di esso in Blockchain (rappresentata in Figura 9 come BC), come evidenziato in Figura 9.

Messaggio  Ricevuto dal programma  Lanciato programma per inserirlo in BC

Figura 9: Procedimento di inserimento di un dato in Blockchain (BC)

6.2.4 Calcolo in Blockchain del coefficiente di ripartizione

Per il calcolo del coefficiente di ripartizione degli autoconsumi è necessario avere i dati relativi all'energia totale prodotta e all'energia totale utilizzata. Per ottenere questi dati dalla Blockchain, occorre visualizzare tutti quelli presenti e selezionare solamente coloro che si dimostrano di interesse. Questa operazione richiederebbe parecchio tempo nel caso in cui i dati siano molti. La soluzione scelta è stata quella di inserire all'interno dello Smart Contract "DSS" alcune variabili con la funzione di accumulatore, con il seguente funzionamento: quando un dato viene inserito in Blockchain, se ne identifica la tipologia e si aggiunge il suo valore alla variabile associata al tipo di dato.

In questo modo, per il calcolo del coefficiente di ripartizione degli autoconsumi, basta estrarre i valori necessari da queste variabili.

È altresì possibile azzerare questi accumulatori tramite uno specifico script Python, l'unica richiesta è il possesso dei permessi di admin.

Per quanto riguarda il mantenimento del coefficiente di ripartizione, esso viene automaticamente inserito in Blockchain quando si esegue lo script Python responsabile del relativo calcolo.

7 Database e Blockchain a confronto

In aggiunta allo sviluppo del sistema, si è scelto di effettuare un confronto tra la Blockchain e le basi di dati.

In primis, per effettuare un rapido confronto, si può notare quanto riassunto in Tabella 1 [7][8].

Tabella 1: Confronto Blockchain e Database

<i>Blockchain</i>	<i>Database</i>
Non richiede un amministratore	Richiede un amministratore
Decentralizzata	Centralizzata
Nessun meccanismo per la gestione degli accessi	Presenta un meccanismo per la gestione degli accessi
Copia completa su ogni <i>peer</i>	Approccio client-server
Supporta: Read e Write	Supporta: Create, Read, Update e Delete
Più complessa da implementare	Facile da implementare
La performance è influenzata dai metodi di verifica e validazione	Può essere davvero rapido e offre scalabilità

Già dalla Tabella 1 è quindi possibile notare sostanziali differenze tra le due tecnologie.

D'altro canto però entrambe le soluzioni presentano degli svantaggi, come è possibile notare in Tabella 2 [8].

Tabella 2: Confronto svantaggi di Blockchain e Database

<i>Blockchain</i>	<i>Database</i>
Alto consumo energetico	Punto unico di caduta (<i>single point of failure</i>)
Con una grande mole di transazioni non scala molto bene	Amministrazione degli account
Può assumere una grande dimensione	Occorre applicare le corrette patch di sicurezza al sistema.
Le tasse per le transazioni possono essere elevate	-
Tipi differenti di Blockchain non sono interoperabili	-

Come si può notare, Blockchain presenta più svantaggi rispetto ad un comune Database. Per meglio comprendere gli svantaggi di Blockchain, verranno ora esaminati punto-punto i singoli casi descritti in Tabella 2.

- Alto consumo energetico: la potenza computazionale richiesta per la creazione di un blocco o per la verifica e validazione di una transazione, deve essere demandata a livello hardware. Esso necessita di energia elettrica e più le operazioni che deve effettuare sono complesse, più ne sarà richiesta;
- Scalabilità: ogni transazione deve essere gestita a sé stante, quindi nel caso esse siano molte, i ritardi sono inevitabili. Una soluzione a questo può essere accorpare, quando possibile, più operazioni in un'unica transazione.
- Dimensione: un problema in questo senso può essere quello dell'entrata in Blockchain di un nuovo nodo. Esso dovrà infatti

scaricare la sua copia locale della Blockchain e se essa è di grosse dimensioni si tratterà di una lunga operazione. In Figura 10 è possibile notare come la dimensione dell'intera Blockchain Ethereum sia cresciuta nel corso degli anni [9].

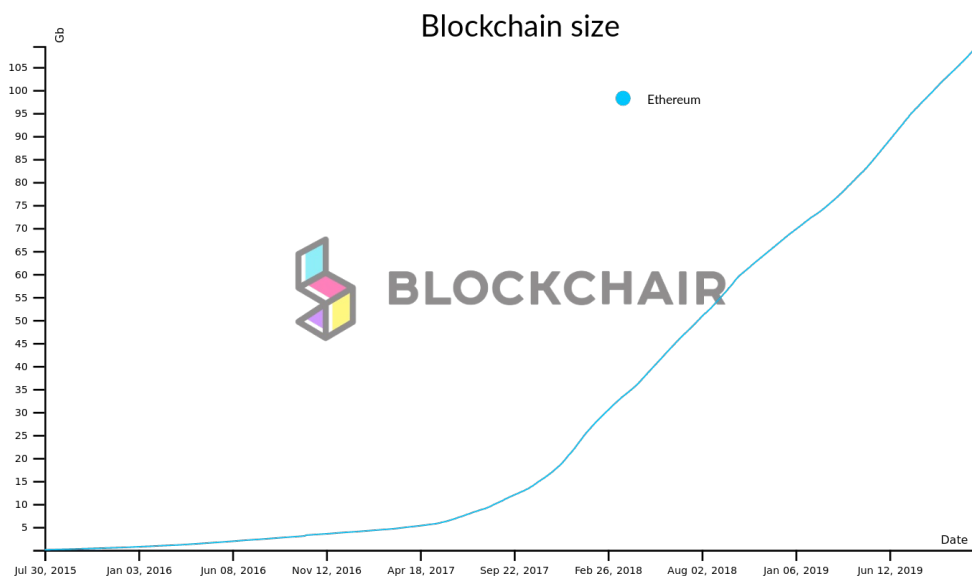


Figura 10: Crescita della dimensione della Blockchain di Ethereum

- Tasse: analogamente al concetto di domanda-offerta del mercato economico, quando la domanda di transazioni cresce, aumenta il numero di blocchi richiesti. Per questo motivo le tasse aumentano, per poter ricompensare maggiormente i *miner*.
- Interoperabilità: esistono vari protocolli volti alla comunicazione ed interfacciamento di due Blockchain differenti. Nonostante questo, la Blockchain è per sua natura intesa come un sistema a sé stante.















Dopo aver esaminato questi dettagli, è possibile affermare che il vantaggio di utilizzare una tecnologia o l'altra non è sempre presente, occorre infatti tenere conto del caso d'uso.

8 Conclusione e sviluppi futuri

In conclusione, rispetto all'analisi effettuata, si può affermare che il vantaggio di utilizzare Blockchain rispetto ad un database non è sempre presente. Nel momento in cui si deve prendere la decisione su quale tecnologia utilizzare, è utile porsi la domanda: "che tipologia di dati occorre mantenere?".

In Tabella 3 [8] è possibile notare i casi d'uso in cui si presenta migliore l'utilizzo di una data tecnologia rispetto alla sua controparte.

Tabella 3: Migliori utilizzi di Blockchain e Database a confronto

	<i>Blockchain</i>	<i>Database</i>
Dati che richiedono molti aggiornamenti		
Processo delle transazioni online rapido		
Informazioni non trasparenti rispetto al pubblico		
Dati finanziari che richiedono un'elaborazione rapida		
Dati che non richiedono verifiche o validazioni		
Applicazioni autonome per lo storage dei dati		
Mantenimento di dati relazionali		

Transazioni monetarie		
Trasferimento di valute		
Verifica di dati fidati		
Verifica a chiave pubblica		
Applicazioni decentralizzate (DApps)		
Sistemi di votazione		

Per quanto riguarda gli sviluppi futuri, ce ne potranno essere sicuramente per gli script Python. Questi infatti potrebbero essere migliorati sotto vari aspetti, quali:

- l'inserimento automatico e dinamico di:
 - ABI del contratto;
 - indirizzo dell'account;
 - password legata all'account;
 - indirizzo del contratto schierato in Blockchain;
- l'interazione tramite pagine web;
- fornire un file di configurazione per l'inserimento automatico, all'interno dello script, l'indirizzo del Broker.

Un ulteriore sviluppo futuro potrebbe essere l'implementazione di un metodo per la generazione automatica delle fatture elettroniche.

9 Bibliografia

- [1] : [ONLINE], www.wired.it, <https://www.wired.it/economia/finanza/2019/10/03/ikea-blockchain/>, “Ikea vende i primi mobili pagati con uno smart contract su blockchain”, consultato il 14 Novembre 2019.
- [2] : RSE (Ricerca sul Sistema Energetico), “AUTOCONSUMATORI COLLETTIVI DI ENERGIA”, 2019, consultato il 5 Novembre 2019
- [3] : Narayanan Arvind, Bonneau Joseph, Felten Edward, Miller Andrew, Goldfeder Steven, “Bitcoin and cryptocurrency technologies: a comprehensive introduction”, 2016, consultato il 18 Ottobre 2019
- [4] : [ONLINE], www.hedgetrade.com, <https://hedgetrade.com/3-types-of-blockchain-explained/>, consultato il 01 Novembre 2019
- [5] : [ONLINE], Solidity website, <https://www.ethereum.org/>, consultato il 18 Ottobre 2019
- [6] : [ONLINE], “history of ethereum”, 2019, <https://www.coinmama.com/guide/history-of-ethereum>, consultato il 30 Ottobre 2019
- [7] : [ONLINE], www.101blockchains.com, <https://101blockchains.com/wp-content/uploads/2019/04/Blockchain-vs-Database.jpg>, consultato il 6 Novembre 2019

- [8] : [ONLINE], www.hackernoon.com,
<https://hackernoon.com/databases-and-blockchains-the-difference-is-in-their-purpose-and-design-56ba6335778b>,
consultato il 6 Novembre 2019
- [9] : [ONLINE], www.blockchair.com,
<https://blockchair.com/ethereum/charts/blockchain-size>,
consultato il 07 Novembre 2019

10 Ringraziamenti

A conclusione di questa relazione, è doveroso ringraziare tutte le persone che mi sono state vicine e che ho avuto modo di conoscere durante questo percorso.

In primis vorrei ringraziare la Prof.ssa Lavinia Egidi, tutore interno per lo svolgimento di questo tirocinio, per avermi sapientemente orientato e seguito con estrema disponibilità durante lo svolgimento del progetto e nella stesura della relazione finale.

Inoltre, vorrei ringraziare l'azienda RSE per avermi offerto la possibilità di svolgere l'attività di studio guidato. In particolar modo vorrei ringraziare la Dott.ssa Roberta Terruggia, e colleghi, per avermi seguito durante lo svolgimento del progetto con grande disponibilità.

Un ringraziamento anche alla mia famiglia, in particolare a mia madre e mio padre che, grazie al loro sostegno morale ed economico, hanno reso possibile il raggiungimento di questo traguardo.

Un ringraziamento particolare anche ad Elisa che, grazie al suo affetto (e pazienza) ha saputo supportarmi (e sopportarmi) per l'intero percorso universitario, rendendo ancora più importante questo traguardo. Inoltre, ha saputo incoraggiarmi, nonostante le mie paure, più di chiunque altro.

Un enorme grazie a tutti!

Tommaso Pessina