Politecnico di Torino

Politecnico di Torino

Corso di Laurea Magistrale in Ingegneria Informatica
A.a. 2021/2022
Sessione di Laurea Marzo/Aprile 2022

Applicabilità della tecnologia Blockchain nella Supply chain

Relatori:
Prof. Guido Perboli
Dott. Vittorio Capocasale

Candidato:
Roberto Ferrio

INDICE

INDICE DELLE FIGURE

Introduzione

Il panorama aziendale attuale, caratterizzato da rapide innovazioni e dalla crescente necessità di adattarsi ad esse, ha portato alla nascita di nuovi modelli di produzione e distribuzione sempre più complessi.

Nella fattispecie, è emersa la consapevolezza che per mantenere la propria competitività sul mercato non è sostenibile concentrarsi unicamente sulle attività interne, ma è necessario rapportarsi con altre aziende per l'approvvigionamento di materiali e servizi.

Nasce così il concetto di *supply chain*, definibile come un insieme di entità autonome e indipendenti che operano insieme allo scopo di rendere disponibili prodotti e servizi all'utente finale.

Non è quindi da ridurre alla tradizionale nozione di logistica, che si riferisce alla sola gestione dei flussi di materiale, stoccaggio delle materie prime e distribuzione dei prodotti al cliente finale. Infatti, la gestione della supply chain include la logistica, ma comprende anche produzione, verifica della qualità, servizi al cliente e le altre attività inerenti alla filiera del prodotto.

Data la mole di processi che compongono la supply chain, ne deriva che la sua gestione sia estremamente complicata, e perciò l'adozione di una adeguata infrastruttura informatica risulta vantaggiosa se non mandatoria. Il discorso relativo alla creazione di un sistema informatico presenta però delle evidenti complessità a causa dei problemi intrinsechi introdotti dalla supply chain.

In primis, una supply chain è composta da molteplici aziende; ciò significa che una parte dei dati viene gestita esternamente all'impresa, e sono necessarie operazioni di standardizzazione e condivisione dei dati tra le diverse compagnie, particolarmente ostiche soprattutto se le aziende hanno già sistemi IT proprietari precedentemente sviluppati.

Vi sono anche problemi relativi alla trasparenza: specialmente nel rapportarsi con partner in *outsourcing*, vi è la necessità per ogni attore di controllare ogni passaggio del prodotto nella filiera produttiva, ma ciò può avvenire solo con la collaborazione tra le parti e con la inter-operatività tra i loro sistemi informativi (che come già detto, risulta difficile da ottenere).

Questa complicanza è aggravata ulteriormente dalle differenti regolamentazioni tra i settori coinvolti, così come dalle giurisdizioni relative a diversi paesi nel caso di collaborazioni internazionali, in particolar modo in materia di privacy e trattamento dei dati.

A tutto ciò va aggiunta anche la possibilità di azioni disoneste da parte di qualche partecipante, sia dal punto di vista della qualità del prodotto, sia riguardo i pagamenti tra le aziende, così come la necessità di far fronte tempestivamente a ritardi o interruzioni sulla catena di produzione.

Non è difficile comprendere come molte tecnologie tradizionali siano inadeguate a far fronte a tali sfide. Negli ultimi anni è emersa però una possibile soluzione, rappresentata dalla tecnologia Blockchain.

Questo lavoro di tesi mira a valutarne l'applicazione nel campo della supply chain e della logistica, fornendo un'introduzione tecnica sull'argomento e specificando quali problemi si punta a risolvere nelle catene di approvvigionamento attraverso l'utilizzo di questa tecnologia, così come le complicanze da affrontare nella sua introduzione e le limitazioni intrinseche derivanti dalla sua filosofia di progettazione.

Nella sezione conclusiva verrà inoltre presentato un caso studio semplificato di una supply chain di portata ridotta, della quale si analizzeranno le attività cardine, le criticità e le funzioni che una eventuale implementazione della Blockchain possa svolgere per migliorare il processo aziendale e la cooperazione tra le entità.

Verranno inoltre comparati gli attuali framework Blockchain di livello industriale utili a questo scopo, per poi concludere con l'esposizione dei test svolti con l'obiettivo di verificare la fattibilità tecnica dell'implementazione e mostrare le pratiche più consone in termini di codifica dei dati e scrittura del codice tali da massimizzare l'efficienza del sistema, fornendo perciò una base teorica per eventuali sviluppi successivi del progetto.

1. Blockchain

Il termine Blockchain nasce nel 2008 da un'idea di "Satoshi Nakamoto", pseudonimo dietro il quale si cela probabilmente più di un individuo, con lo scopo di fungere da tecnologia abilitante per la valuta digitale Bitcoin[1].

Tale tecnologia appartiene alla famiglia dei *Distributed Ledger,* ovvero database condivisi e accessibili da molteplici partecipanti. Ciò che la distingue rispetto ad altre soluzioni per il salvataggio dei dati è il rivoluzionamento del concetto di fiducia, che invece di essere garantita da un intermediario come nei sistemi classici, viene resa parte integrante del protocollo attraverso meccanismi matematici e crittografici.

Ogni transazione, prima di entrare a far parte del ledger, deve essere verificata da tutti i partecipanti tramite un algoritmo di consenso e, grazie alla crittografia, l'informazione diventa pressoché immutabile.

Le caratteristiche chiave che ne derivano sono le seguenti:

- **Decentralizzazione:** le informazioni sono replicate su più nodi. Non esiste quindi alcuna autorità centrale che mantenga i dati o verifichi le transazioni, garantendo quindi maggiore sicurezza e resilienza del sistema data l'assenza di un *single point of failure*
- **Autonomia:** il protocollo utilizzato consente di gestire le informazioni senza l'impiego di intermediari
- **Immutabilità:** i dati possono solo venire aggiunti (un ledger è un registro *append-only*), e le informazioni scritte non possono essere modificate
- **Trasparenza e verificabilità:** il contenuto del registro è visibile a ogni entità, ed è inoltre possibile risalire alle informazioni presenti in un qualsiasi momento della sua storia
- **Autenticità e non ripudio:** grazie alle firme digitali è possibile confermare la provenienza di una transazione

La Blockchain rappresenta perciò una fondamentale rivoluzione, che rende possibile l'implementazione di specifici modelli di sistema distribuito nei quali sicurezza e consistenza del dato sono garantiti per design, con pesanti ramificazioni non solo in campo finanziario, ma anche, come verrà spiegato in seguito, nella gestione della supply chain.

1.1 Stato e transazioni

A livello teorico, è possibile formalizzare la Blockchain come una macchina a stati: sotto tale definizione, si può interpretare lo stato di una Blockchain come "l'insieme di informazioni rilevanti per decidere il consenso ", quali ad esempio il saldo dei conti di tutti gli account della rete Bitcoin[2].

È possibile modificare lo stato di accettazione attuale da S_i a S_{i+1} inviando una transazione, che rappresenta quindi una funzione di transizione dell'automa.

L'insieme di transazioni valide deve essere definito e salvato sulla Blockchain sotto forma di *smart contract*. Gli smart contract sono programmi considerabili a prova di manomissione, che permettono l'esecuzione automatica, il controllo e la certificazione di determinate funzioni ed eventi.

Nello specifico, sono script che risiedono nella Blockchain, e perciò posseggono un indirizzo verso cui è possibile inviare transazioni allo scopo di eseguirne le funzioni. Queste vengono eseguite in modo indipendente e automatico su tutti i nodi della rete, sfruttando come argomenti i dati inclusi nella transazione[3].

Nonostante il nome possa trarre in inganno, gli smart contract non hanno di per sé alcun valore legale, ma possono però ottenerlo se accompagnati da un contratto vero e proprio che ne dia legittimità.

Al momento, la piattaforma più famosa per la creazione di smart contract è *Ethereum*, dalla quale si può sfruttare la *Ethereum Virtual Machine* (EVM) utilizzando il linguaggio ad oggetti *Solidity,* specifico per questo tipo di programmi.

Per assicurare la congruenza dello stato, la Blockchain deve garantire la corretta sequenzialità delle transazioni, così come la loro provenienza e immutabilità. Queste proprietà vengono ottenute sfruttando, come fondamenta, dei particolari tipi di funzioni chiamate *funzioni di hash crittografiche*, e la tecnica crittografica della *firma digitale* che si appoggia ad esse.

1.2 Funzioni di hash crittografiche

Una funzione di hash è una funzione deterministica e unidirezionale che mappa dati di dimensione arbitraria in una stringa di lunghezza fissa. Nello specifico, definiti X e Y rispettivamente il dominio e il codominio della funzione h, $\forall x \in X, y = h(x)$, con $y \in Y$ e $|y| = l$, dove l è un intero fisso dipendente dalla funzione utilizzata. L'output così ottenuto prende il nome di *hash*, o *digest*.

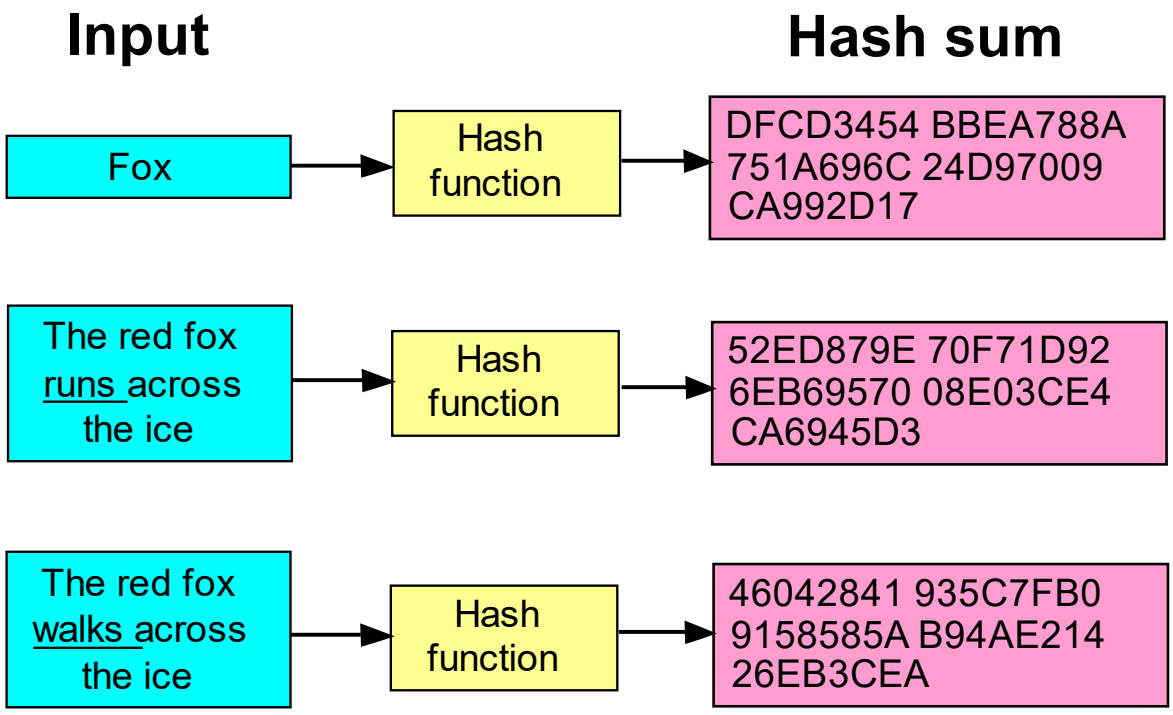

Figura 1 - Funzione di hash[4]

Una funzione di hash gode delle seguenti proprietà:

- **Determinismo:** mantenendo costante il valore in input, il risultato della funzione sarà sempre lo stesso
- **Efficienza:** calcolare un valore hash deve essere semplice e veloce a prescindere dal valore in input
- **Uniformità:** ogni valore di hash appartenente a Y deve essere generato a partire dai valori in X con circa la stessa probabilità

Una funzione di hash, per essere definita crittografica, deve possedere inoltre delle proprietà aggiuntive:

- **Resistenza alla preimmagine:** dato $y \in Y$, deve essere difficile trovare un x tale che $h(x) = y$
- **Resistenza alla seconda preimmagine:** dato $x_1 \in X$, deve essere difficile trovare un $x_2 \neq x_1$ tale che $h(x_1) = h(x_2)$
- **Resistenza alla collisione:** deve essere difficile trovare x_1 e $x_2 \in X$, con $x_1 \neq x_2$, tali che $h(x_1) = h(x_2)$. La resistenza alla collisione, indicata anche come forte resistenza alla collisione, implica una resistenza alla seconda preimmagine, ma non implica la resistenza alla preimmagine

Questo genere di funzioni è utile per verificare l'integrità dei dati trasmessi, dato che è possibile inviare il digest insieme al messaggio spedito, così da poterlo comparare al digest del messaggio in ricezione, che sarà diverso in caso di corruzione dei dati.

1.3 Firma digitale

La firma digitale è un meccanismo crittografico volto a dimostrare l'autenticità di un messaggio a prescindere dalla sicurezza del canale di comunicazione utilizzato.

L'uso garantisce al destinatario le seguenti proprietà:

- **Autenticazione:** l'identità del mittente è quella corretta
- **Integrità:** il messaggio non è stato alterato durante l'invio
- **Non ripudio:** il mittente non può negare di aver inviato il messaggio

Come fase preliminare, è necessario che il mittente e il destinatario si accordino sull'uso di una specifica funzione di hash crittografica, così come una coppia di funzioni per la cifratura e la decifratura. Il mittente deve inoltre creare una coppia di chiavi asimmetriche, ovvero una pubblica da condividere con il destinatario e una privata da mantenere segreta.

Fatto ciò, il mittente applica la funzione di hash al messaggio, per poi utilizzare la funzione di cifratura in congiunzione con la sua chiave privata sul digest così ottenuto, criptandolo e ricavando la firma digitale. A questo punto, può spedire il messaggio originale in chiaro, detto anche *plaintext*, insieme alla firma.

Il destinatario, ricevuto il messaggio composto da plaintext e firma, decripta quest'ultima utilizzando la funzione di decifratura e la chiave pubblica del mittente, così da ottenere il digest calcolato in precedenza, e lo confronta con quello ottenuto dal messaggio in arrivo. La firma risulta valida se i due hash coincidono.

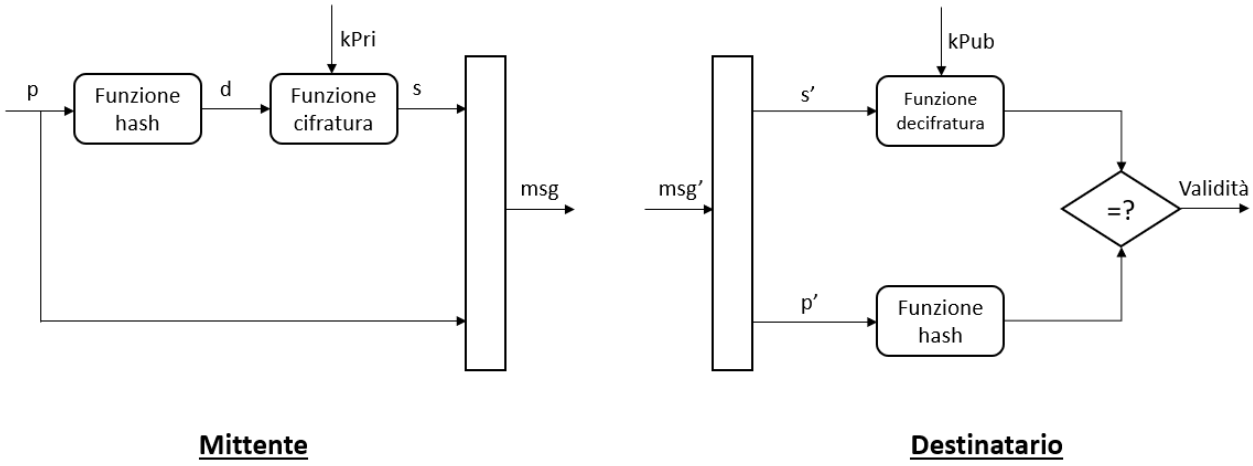

Figura 2 - Firma digitale

La firma digitale viene sfruttata ampiamente nell'ecosistema Blockchain, in quanto un utente è tenuto a firmare ogni transazione inviata. Ne deriva come ogni nodo della rete possegga una coppia di chiavi, una privata e una pubblica. Su quest'ultima viene solitamente eseguito un algoritmo di hash in modo da ottenere una stringa di lunghezza inferiore che funga da indirizzo del nodo, che verrà quindi impiegato per specificare il mittente o il destinatario di una transazione.

L’uso della firma garantisce la provenienza della transazione e che questa non sia stata alterata durante l’invio. La Blockchain, però, è un sistema distribuito composto da molteplici attori, e di per sé la firma digitale non consente di assicurare il corretto ordinamento delle transazioni, né la validità del suo contenuto o perfino l’immutabilità del ledger, ed è per questo che firme ed hash rappresentano solo un pezzo del puzzle su cui si fonda il suddetto sistema.

1.4 Struttura a blocchi

Con le sole proprietà finora delineate, il sistema sarebbe vulnerabile ad attacchi di tipo *Double spending*. Infatti, a differenza degli asset fisici, quelli digitali sono facilmente duplicabili, ed è possibile per un utente malevolo eseguire molteplici transazioni inviando una somma già spesa in precedenza.

Il primo passo nella soluzione del problema è quello di certificare la sequenzialità delle transazioni, e la soluzione proposta da Bitcoin è simile a quella sfruttata nei server di timestamp: le transazioni vengono racchiuse in blocchi. Alla creazione di un nuovo blocco, questo viene sottoposto a hashing includendo anche l’hash del blocco precedente, formando quindi una catena.

Ciò lega la transazione all’interno di un blocco a una specifica posizione temporale, dato che l’inserzione avviene sempre in testa alla catena, e perciò la sua presenza nel blocco B_i implica un’esecuzione precedente ad una nel blocco B_{i+1}.

Non è inoltre possibile l’inserzione di una nuova transazione in un blocco passato, men che meno la sua manomissione, siccome qualsiasi modifica invaliderebbe gli hash di tutti i blocchi successivi.

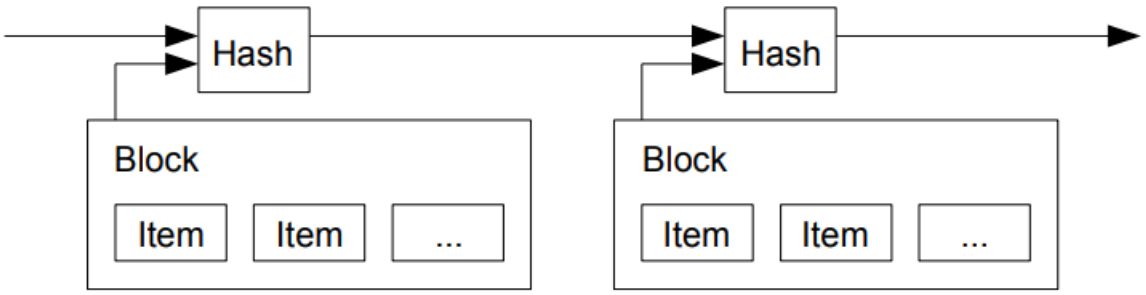

Figura 3 - Hashing sequenziale[1]

Data la grande mole di transazioni presenti in un blocco[5], è necessario un metodo efficiente per la verifica dei contenuti a partire dall'hash.

Per questo motivo, la Blockchain sfrutta una struttura dati detta *albero di Merkle* o *albero hash*, un albero in cui ogni nodo foglia è etichettato con l'hash crittografico di un blocco dati, nel nostro caso rappresentato da una transazione, mentre ogni nodo interno è etichettato con l'hash delle etichette dei suoi nodi figli.

La radice nell'albero rappresenta l'impronta crittografica di tutti i nodi dell'albero, e una modifica a uno qualsiasi di essi invaliderebbe l'intera struttura.

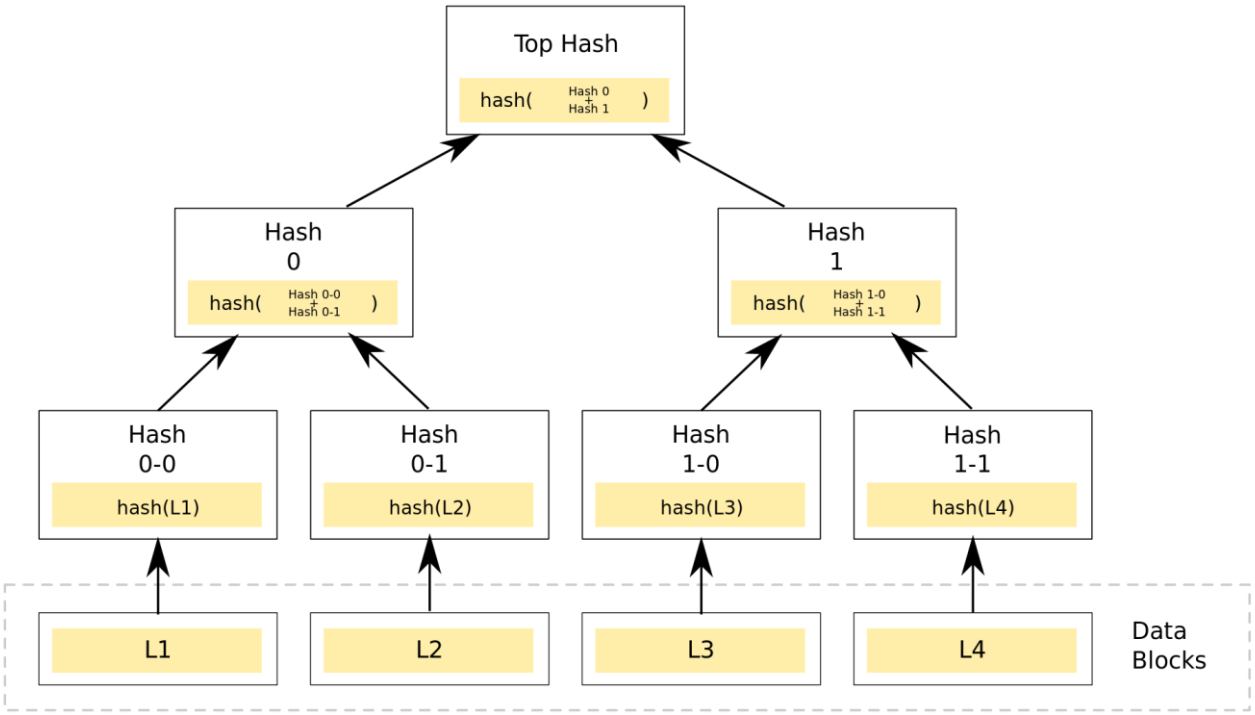

Figura 4 - Struttura albero di Merkle[6]

Verificare la presenza di una transazione in un blocco comporta la necessità di ricostruire l'albero di Merkle e confrontare la radice ottenuta con quella data.

Per fare ciò è necessaria la risalita del ramo di appartenenza del nodo, e quindi l'esecuzione di un numero di funzioni di hash pari alla profondità dell'albero. Normalmente, la complessità della computazione sarebbe lineare con il numero di transazioni, quindi $O(N_{tx})$, ma utilizzando la struttura ad albero di Merkle, nello specifico la struttura ad albero binario, il numero di operazioni necessarie si riduce a $O(log_2(N_{tx}))$.

Nell'esempio sottostante di albero di Merkle binario perfettamente bilanciato ad otto foglie, si può osservare come, per il controllo della transazione T_4, verrà eseguita una query alla rete riguardo H_4, la quale dovrà restituire i blocchi H_3, H_{12} e H_{5678} necessari alla verifica, che consisterà nella risalita del ramo eseguendo tre volte la funzione di hash.

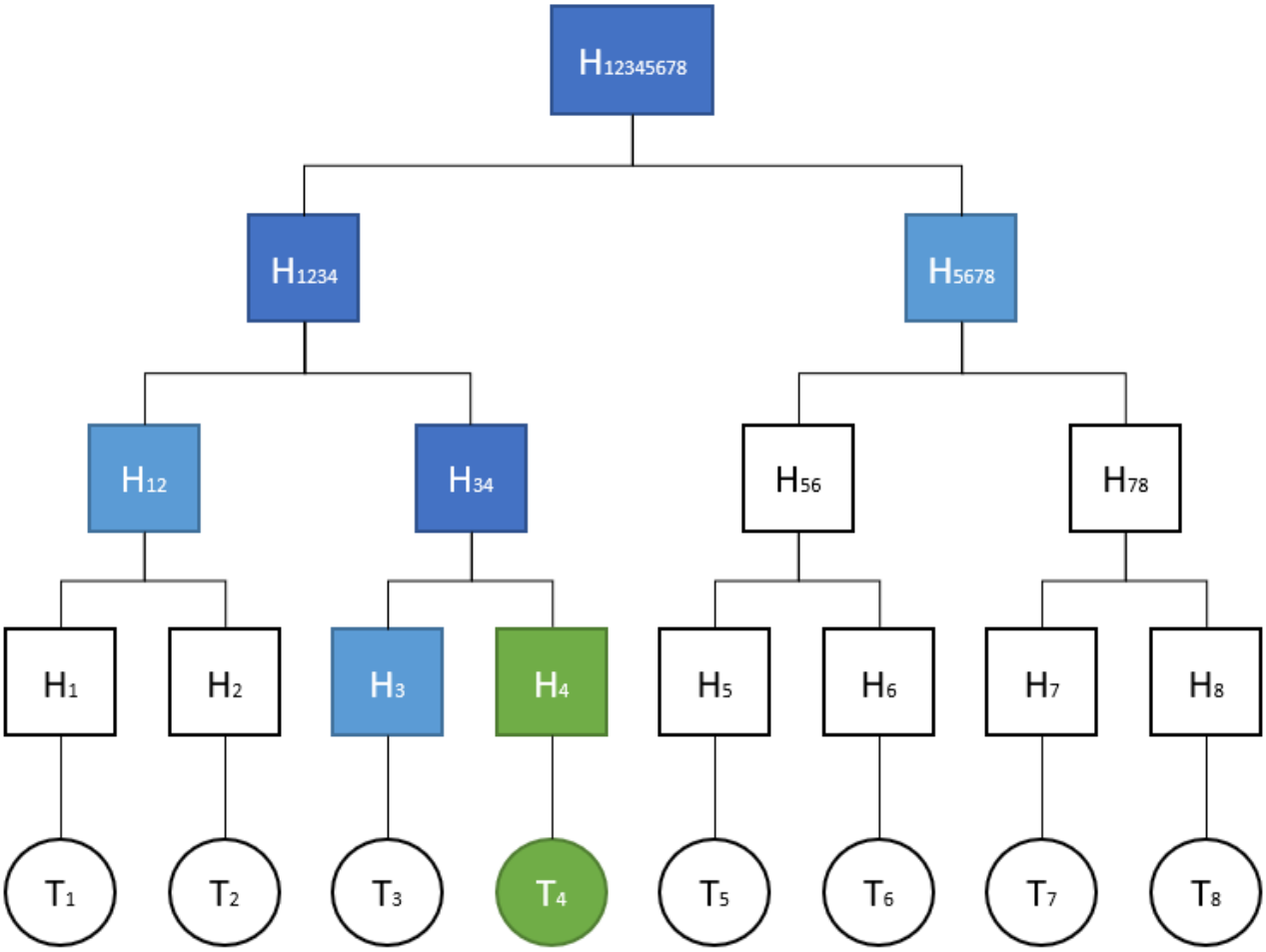

Figura 5 - Ricostruzione di un ramo in un albero di Merkle

1.5 Algoritmi di consenso

L'ultima problematica che la Blockchain mira a risolvere è il *problema dei generali bizantini*. Il nome deriva dalla descrizione informale del problema, che viene esemplificato da una situazione in cui tre o più generali bizantini, impegnati in una campagna d'assedio, devono decidere se procedere con l'attacco o ritirarsi, potendo comunicare tra loro solo attraverso messaggeri.

È probabile, però, che tra i ranghi vi siano uno o più traditori con l'obiettivo di confondere gli altri con messaggi discordanti, e l'unica possibilità di vittoria è che tutti i membri siano d'accordo sulla mossa da fare.

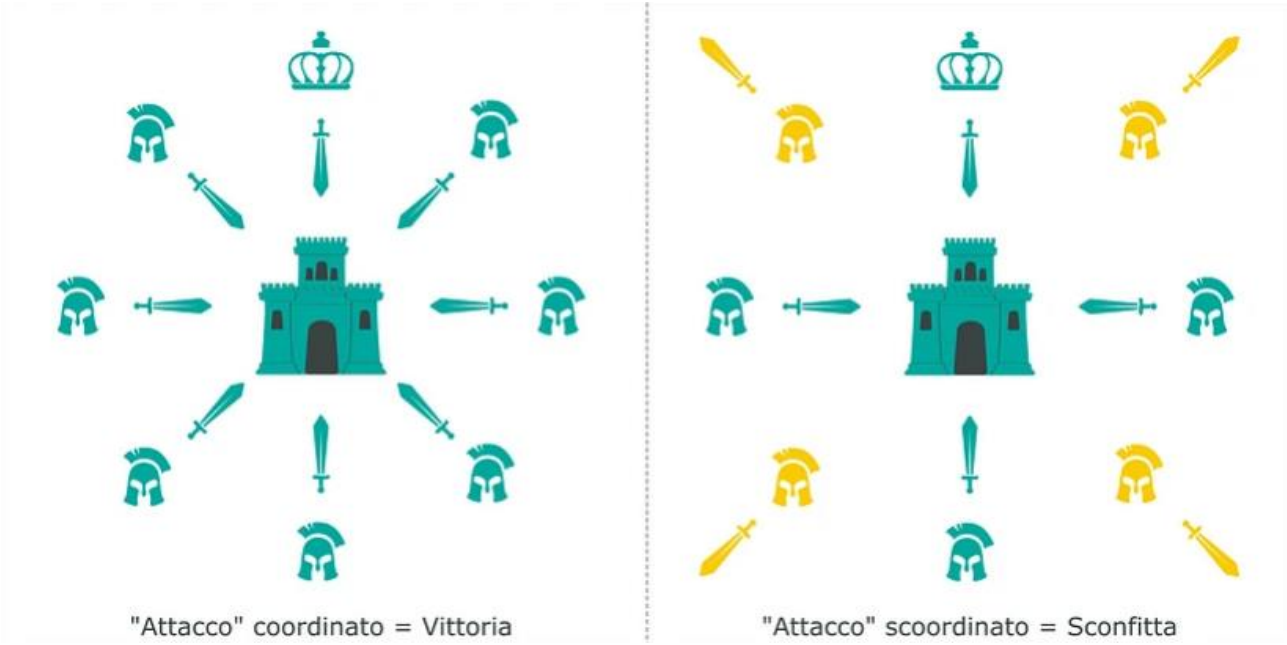

Figura 6 - Problema dei generali bizantini[7]

Traducendo il problema nella sua definizione formale, si richiede che, dato un numero *N* di processi tra i quali ve ne siano alcuni che possono non inviare messaggi o inviarne con contenuto arbitrario, al termine dell'algoritmo tutti i processi corretti impostino la variabile di decisione sullo stesso valore[8].

La Blockchain ottiene ciò grazie all'implementazione di un *consensus algorithm* che, attraverso una sorta di voto di maggioranza, permette ai nodi di accordarsi sullo stato del sistema nonostante la rete in cui risiedono sia per sua natura distribuita e inaffidabile.

L'algoritmo proposto da Nakamoto, attualmente ancora utilizzato nella rete Bitcoin, è il *Proof-of-Work* (*PoW*), ispirato dal sistema anti-DoS *Hashcash*[1].

Questo algoritmo consiste nell'incoraggiare i partecipanti, i cosiddetti *miners*, a competere nel trovare per primi una soluzione ad un complesso quesito matematico sotto promessa di un compenso per il lavoro svolto. Il risultato di tale processo, detto *mining*, è studiato per essere di facile verifica, mentre il problema in sé richiede grossi consumi di potenza computazionale per trovarne la soluzione, dato che l'unico modo per risolverlo comporta l'uso di tecniche di *brute force* e l'assenza di algoritmi efficienti.

Solitamente il problema si basa sull'uso di funzioni di hash: nel caso di Bitcoin consiste nel generare un nuovo blocco tale che il suo digest abbia come prefisso un numero definito di zeri. Tale numero rappresenta la difficoltà del problema, che aumenta esponenzialmente tanto più questo diventa grande, e varia in tempo reale a seconda del numero di blocchi generati all'ora in modo da bilanciare il rateo di crescita della catena senza pregiudicare la resilienza agli attacchi.

Per fare ciò, partendo da un dato input, ovvero le transazioni provenienti dalla rete raccolte in un nuovo blocco, si modifica iterativamente un numero detto *nonce* all'interno del blocco fino all'ottenimento del digest desiderato.

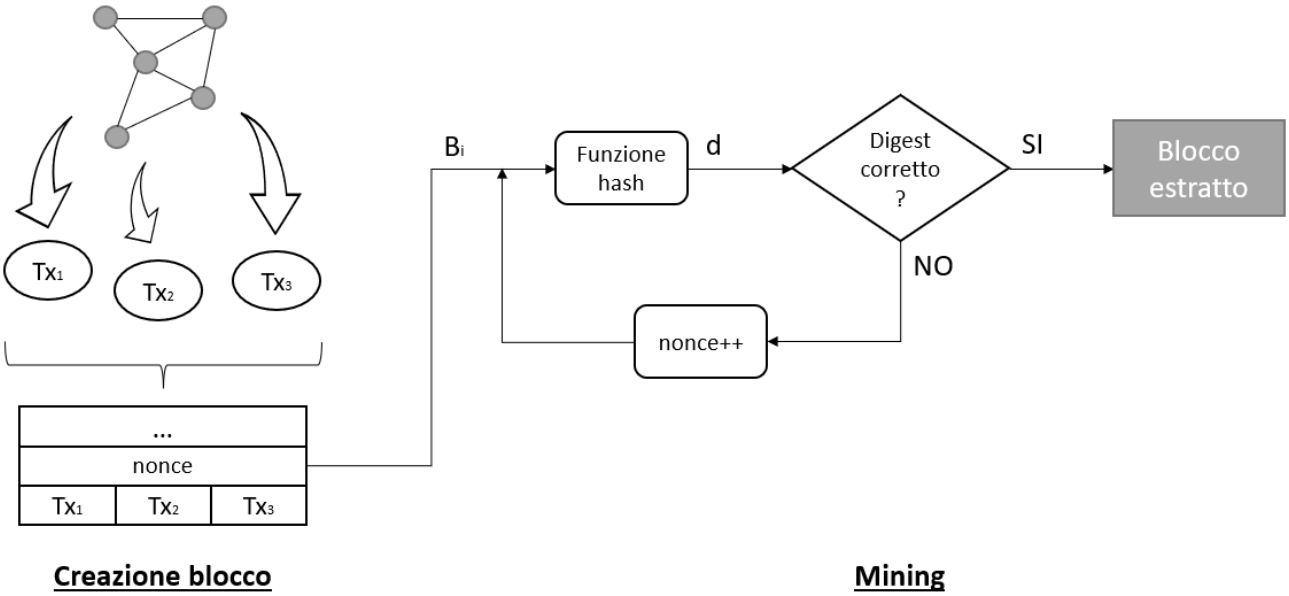

Figura 7 - Algoritmo di consenso PoW

Il presupposto su cui si basa questo algoritmo è che più della metà della potenza computazionale sia detenuta da nodi onesti. In questo modo, la catena da loro generata risulterà quella più lunga, e verrà perciò considerata come quella valida.

Oltre a garantire la legittimità dei dati in ingresso nella Blockchain, PoW fornisce anche una forte resistenza alle modifiche: come già spiegato, la struttura della catena implica che una modifica a un blocco invalidi tutti i successivi, rendendo necessario effettuare nuovamente il processo di mining per ogni blocco invalidato. Ne deriva che, all'allungarsi della catena, la potenza necessaria per sostituire i blocchi e ottenere quindi il controllo della catena più lunga aumenti sempre di più, fino a diventare insostenibile.

Ciò si traduce nel concetto di "*finalità probabilistica*" o anche "*consistenza debole*", che indica come la probabilità che una transazione raggiunga uno stato in cui questa non possa venire revocata o modificata aumenti all'aumentare della sua profondità nella catena. È il motivo per il quale nella rete Bitcoin è consigliabile attendere l'aggiunta di sei blocchi prima di considerare una transazione come finalizzata.

Una conseguenza di questa proprietà è la possibilità che si verifichino dei *fork*: esiste l'eventualità che due o più miner pubblichino contemporaneamente un blocco valido, e che prima che questa divisione sia rilevata i miner continuino a lavorare parallelamente sulle biforcazioni. Le transazioni successive al riconoscimento del fork si collegheranno ai blocchi del ramo più lungo, mentre le altre verranno scartate. Ciò provoca pesanti impatti sulla latenza delle transazioni e in generale sull'efficienza del sistema.

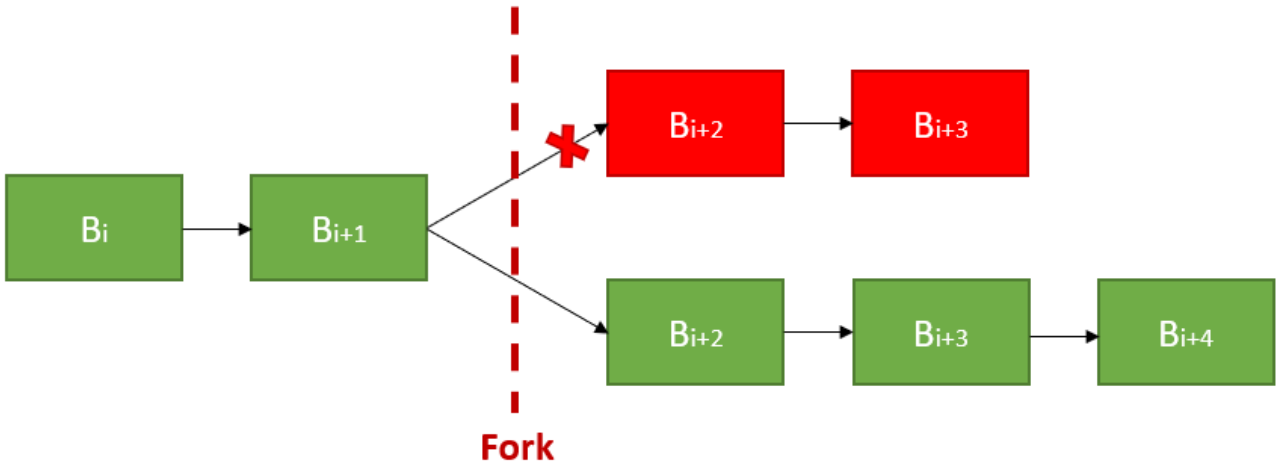

Figura 8 - Fork nella Blockchain

Un altro lato negativo è rappresentato proprio da ciò che rende l'algoritmo sicuro, ovvero il consumo di CPU richiesto dal mining, che porta a uno spreco eccessivo di risorse e quindi grandi consumi energetici e ridotto throughput.

Esistono in ogni caso molteplici algoritmi di consenso. Il *Proof-of-Stake* (*PoS*) è una versione alternativa al PoW meno dispendiosa in termini energetici: invece di mettere in gioco la propria potenza computazionale, i nodi forniscono una prova delle finanze a loro disposizione, e i loro voti sono pesati in modo proporzionale alla loro ricchezza. Il concetto alla base di questo algoritmo è che più un utente possiede risorse nella rete e meno è incentivato a commettere atti illeciti.

Ovviamente questo sistema soffre di problemi di accentramento del potere e iniquità nel sistema di voto, ma esistono soluzioni derivanti da PoS con maggiori livelli di equità, come ad esempio *Delegated Proof-of-Stake* (*DPoS*) nel quale i nodi votano i validatori che contribuiranno al consenso invece di parteciparvi direttamente.

PoW e PoS operano sulla base che la rete contenga valute o token rispettivamente per la ricompensa dei nodi e la verifica nella votazione. Ciò non è sempre così dato che ci sono reti Blockchain nelle quali non vengono gestite criptovalute.

Un esempio di algoritmo che supporta l'assenza di token è il *Proof-of-Authority* (*PoA*), basato sul valore dell'identità dell'operatore del nodo e della sua reputazione. I validatori sono perciò tenuti a divulgare la propria identità, ed è quindi indicata in contesti dove questa è già conosciuta e fidata.

Gli algoritmi qui elencati, compresi alcuni derivanti dal PoA come l'*Istanbul Byzantine Fault Tolerance* (*IBFT*), appartengono alla classe di algoritmi definibili *Byzantine Fault Tolerant* (*BFT*), ovvero resistenti al comportamento malevolo e al crash di un numero limitato di nodi[9].

Esiste un altro gruppo di algoritmi di consenso che proteggono solo da quest'ultimo, chiamati *Crash Fault Tolerant* (*CFT*). Come si può immaginare, questa tipologia di algoritmi andrebbe impiegata solo in contesti nei quali è garantita la fiducia tra i partecipanti, come nelle Blockchain private. Un esempio di algoritmo CFT è *Raft*, utilizzabile sia in *Hyperledger Fabric*[10] che in *Quorum*[11].

Posta la presenza di eccezioni, gli algoritmi di consenso nei quali un leader propone i blocchi da aggiungere, come Raft o quelli basati sul *Practical Byzantine Fault Tolerance* (*PBFT*) come IBFT, garantiscono la finalità delle transazioni non appena queste sono aggiunte alla Blockchain. Si parla in questi casi di "*finalità deterministica*" o "*consistenza forte*".

1.6 Formato di un blocco

Concludiamo questa trattazione tecnica della Blockchain descrivendo la struttura ad alto livello dei blocchi. È possibile organizzare i blocchi in due sezioni, dette *header* e *body*:

- **Header**
  - *Versione del blocco:* indica il protocollo utilizzato nella validazione del blocco
  - *Radice albero di Merkle:* l'etichetta della radice dell'albero hash, ovvero l'hash di tutte le transazioni presenti nel body
  - *Timestamp:* marca temporale della creazione del blocco in formato Unix
  - *nBits:* soglia massima del digest, inversamente proporzionale alla difficoltà. Perché un blocco sia valido occorre che il suo hash sia uguale o inferiore a questo valore
  - *Nonce:* numero usato per risolvere il consenso
  - *Hash blocco padre:* l'hash del blocco precedente
- **Body**
  - *Contatore transazioni:* il numero di transazioni presenti nel blocco
  - *Transazioni:* la lista delle transazioni

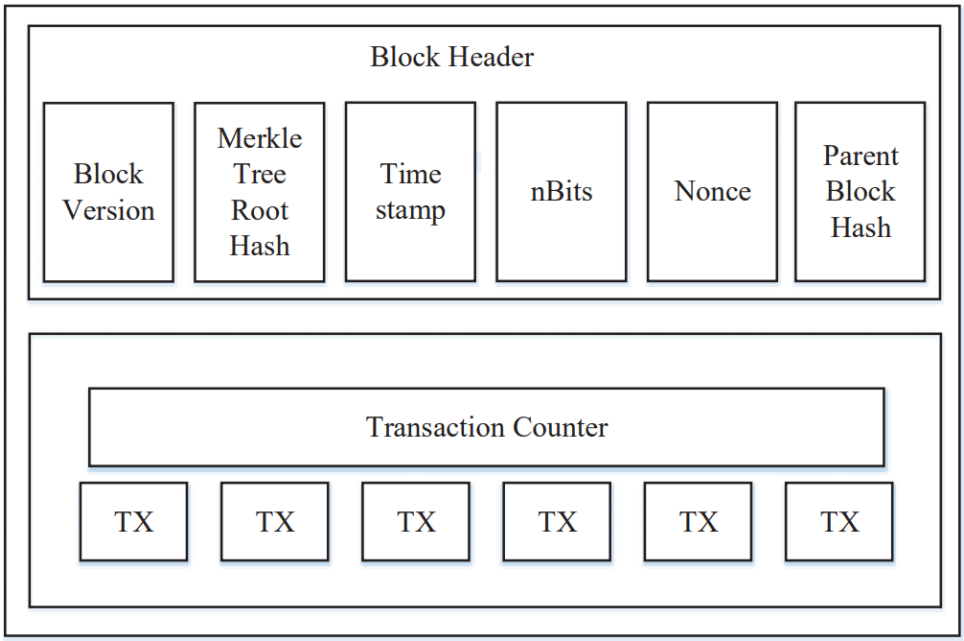

Figura 9 - Formato ad alto livello di un blocco[12]

Va notato che la struttura qui riportata è un modello appartenente alla versione originale della Blockchain, e può differire tra le varie piattaforme. Ad esempio, i campi numero bit e nonce hanno senso di esistere solo in presenza di un algoritmo di consenso di tipo PoW, e saranno eliminati o sostituiti con altre informazioni utili. Inoltre, nella pratica, la struttura è molto più complessa di quella rappresentata: viene riportata a titolo esemplificativo quella di un blocco in Fabric 1.0.

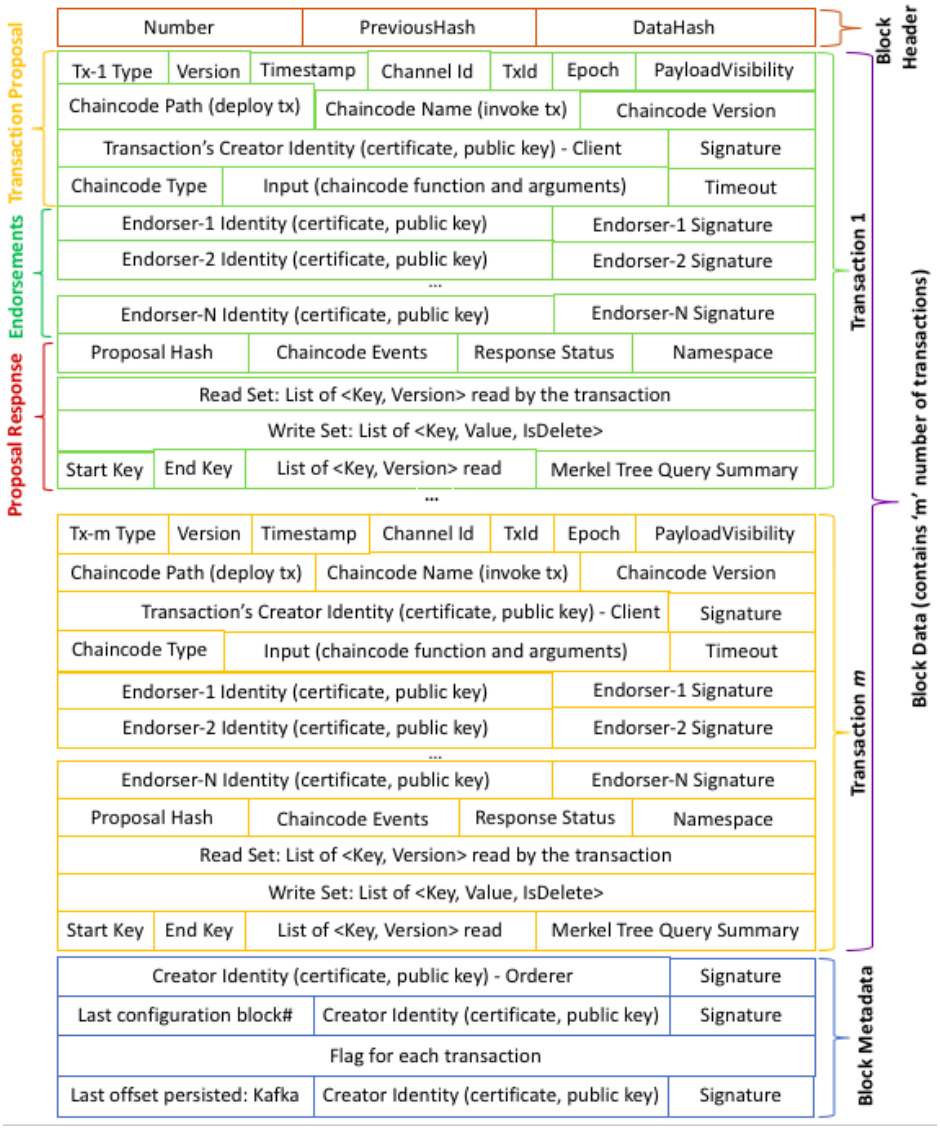

Figura 10 - Struttura di un blocco in Hyperledger Fabric 1.0[13]

A prescindere dalle peculiarità di ogni formato, rimane costante il fatto che l'hash del padre è contenuto nell'header di ogni blocco, in modo da creare la struttura a catena. Nel momento in cui si inizializza la Blockchain, il primo blocco non avrà ovviamente alcun predecessore: tale blocco prende il nome di *blocco genesi*, e fornisce un punto di partenza verificabile dell'intera struttura.

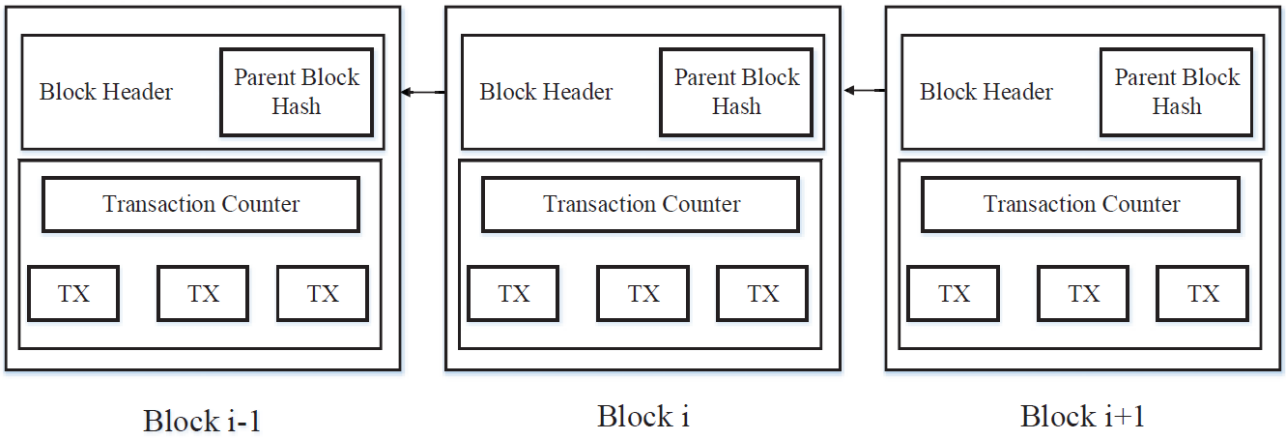

Figura 11 - Struttura ad alto livello della catena di blocchi[12]

1.7 Tipologie di Blockchain

Nell'accezione più pura del termine, una Blockchain è prettamente decentralizzata, consentendo l'accesso a chiunque con il massimo livello di trasparenza. Come si può immaginare, una struttura di questo tipo mal si presta a contesti in cui la privacy rappresenta un punto fondamentale.

Per di più, sicurezza e velocità della rete sono inversamente correlate a causa dei meccanismi di consenso, e in situazioni dove è vitale il mantenimento di un throughput minimo una Blockchain tradizionale non è sostenibile.

Al classico modello di Blockchain pubblica si affiancano quindi nuove distinzioni basate sulla necessità o meno di autorizzazione per l'aggiunta di record (*permissionless* e *permissioned*) e sull'accessibilità della rete (*pubblica* e *privata*). In particolare, si possono distinguere quattro tipologie di Blockchain[14]:

- **Pubblica:** ogni entità può accedere alla rete, visualizzare i dati, effettuare transazioni e partecipare al consenso. Di solito i nodi vengono ricompensati per lo svolgimento delle computazioni necessarie alla creazione di un nuovo blocco, il cosiddetto *"mining"*, così come gli viene richiesto un pagamento per l'esecuzione di una transazione.

  Vantaggi

  - Posta l'osservanza dei protocolli di sicurezza, è una delle più sicure.
  - Indipendente da qualsiasi organizzazione
  - Trasparenza della rete

  Svantaggi

  - Basso throughput
  - Nessuna restrizione sugli accessi
  - Se gli hacker ottengono il controllo di più del 50% della rete, possono alterarla unilateralmente

  Casi d'uso

  - Rete di criptovalute
  - Gestione atti notarili

- **Privata:** il controllo della rete è affidato a un'unica organizzazione. In questo caso non è esatto parlare di Blockchain visto che si può ricondurre, di fatto, a un database centralizzato, e manca quindi la caratteristica di risolvere i problemi di fiducia tra i partecipanti. Tipicamente operano su una rete interna all'azienda, e perciò sono di dimensione nettamente inferiore.

Vantaggi

- Controllo completo sulle autorizzazioni e l'accessibilità fornita ai nodi
- Ottime performance rispetto alle Blockchain pubbliche

Svantaggi

- Non risolve i problemi di fiducia tra i partecipanti
- Vengono persi i punti di forza della Blockchain, e quindi un database distribuito classico rappresenta nella maggioranza dei casi una scelta più performante e meno complessa da implementare

- **Ibrida:** combina elementi della Blockchain pubblica e di quella privata. Nello specifico, l'organizzazione può decidere quali dati mantenere privati e quali dati rendere pubblici. Tipicamente le transazioni non sono pubbliche, ma possono essere verificate da esterni attraverso uno smart contract (utile per la compliance alle normative).

  Vantaggi

  - Maggior privacy rispetto a una Blockchain pubblica
  - Transazioni rapide
  - Scalabilità

  Svantaggi

  - La trasparenza dipende da quanto l'organizzazione sia disposta a mostrare

  Casi d'uso

  - Gestione cartelle cliniche
  - Mercato immobiliare

- **Consortium:** il database è gestito da un numero ristretto di partecipanti. Solo loro ne detengono una copia e partecipano al consenso. Definiscono anche le politiche di invio delle transazioni e di partecipazione alla rete.

È utilizzabile tra entità che non si fidano tra di loro, e che necessitano del controllo su chi può scrivere o leggere i dati (ad esempio se va garantita la *GDPR compliance*), e in più sono indicate nel caso ci siano vincoli minimi sul throughput. Solitamente non vi sono pagamenti per l'esecuzione delle transazioni, ma ogni entità deve gestire almeno un nodo, con costi fissi che prescindono dall'effettivo uso della rete.

I dati contenuti in questo tipo di Blockchain sono considerati affidabili solo per i membri della rete, mentre un'entità esterna dovrebbe valutarli alla stregua dei dati salvati in un database centralizzato.

Vantaggi

- Sicurezza
- Efficienza
- Controllo degli accessi

Svantaggi

- Minor trasparenza rispetto a una Blockchain pubblica

Casi d'uso

- Pagamenti bancari
- Organizzazioni di ricerca
- Supply chain

Va precisato che queste distinzioni sono teoriche, e alcuni framework potrebbero non entrare del tutto in una categoria (ad esempio, Quadrans è classificata come Blockchain pubblica, ma per la partecipazione al consenso il client è obbligato a sottoporsi a una procedura di verifica detta *Know Your Customer*)[15].

2. Supply chain

Come già introdotto, la nascita della Blockchain è legata al contesto finanziario; tuttavia le sue peculiarità hanno attirato le attenzioni di altri ambiti [16-17]. Di questi, uno dei più prominenti è il settore della Supply Chain e della logistica, che vede in questa tecnologia grandi potenzialità nella gestione della filiera produttiva.

Infatti, l'uso garantirebbe non solo la tracciabilità del prodotto in tempo reale, ma ridurrebbe anche eventuali contenziosi tra le società e, in generale, porterebbe a una diminuzione dei costi amministrativi.

Nonostante i grandi vantaggi offerti da questa tecnologia, la sua natura ancora acerba porta difficoltà non trascurabili nella sua adozione in applicazioni logistiche reali.

La sezione seguente mira a fornire una panoramica sul concetto di supply chain, quali problemi la affliggono e come la Blockchain può porsi come soluzione, così come i limiti nell'applicazione di questa tecnologia.

2.1 Definizione di Supply chain

Data la nascita relativamente recente del concetto di supply chain, non vi è un consenso unanime sulla sua definizione. Si può però riassumerne il significato come l'insieme di attori, processi e flussi che portano beni o servizi al cliente finale.

La supply chain racchiude quindi la gran parte delle attività più importanti dell'impresa, tra cui ovviamente la logistica, ovvero la pianificazione, implementazione e controllo dei flussi di materie prime e prodotti lavorati, ma anche produzione, marketing e gestione delle vendite, ricerca e sviluppo e via dicendo.

Il focus dell'impresa si espande, non più concentrato unicamente sull'interno dell'azienda, ma anche verso le organizzazioni facenti parte della filiera, così come al consumatore.

Ne deriva come la gestione della supply chain sia estremamente complessa, e richieda un grande sforzo di integrazione dei principali processi aziendali di tutti gli attori che ne fanno parte.

Infatti, più l'ecosistema diventa grande e si dirama su più livelli, maggiore diventa il flusso di informazioni che le entità devono gestire, e un adeguato sistema informatico è fondamentale per non impattare sull'efficienza.

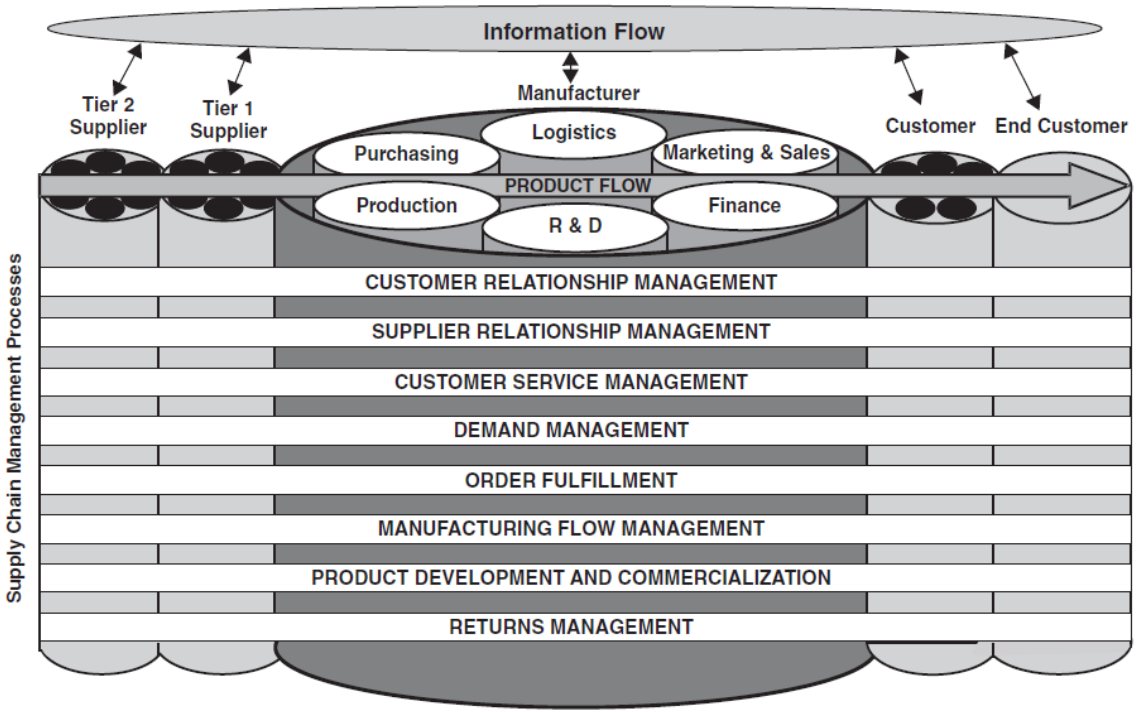

Figura 12 - Attori e processi in una supply chain[18]

Per tali motivi, la transizione ad un ecosistema aziendale coeso non può essere repentina, ma deve essere un percorso organico nel quale ogni progresso funga da fondamenta per il passo successivo. A tal proposito, il modello sviluppato da Kevin McCormack e Archie Lockamy propone cinque livelli per la valutazione della maturità dei processi di gestione della supply chain, basati sulla cooperazione tra le aziende e le performance che ne derivano[19].

- **Ad hoc:** i processi della catena sono mal definiti e sprovviste di metriche per il monitoraggio, con scarso adattamento alla struttura orizzontale. Di conseguenza, la cooperazione è bassa, i costi di gestione alti e le performance non prevedibili, portando un impatto negativo sul cliente finale
- **Defined:** i processi base sono definiti e documentati, ma la divisione dei lavori e delle strutture organizzative rimangono quelle tradizionali, mantenendo la compartimentazione delle funzioni per questioni di governance e obiettivi conflittuali
- **Linked:** la gestione della supply chain viene svolta in maniera strategica, con una visione sull'intera filiera anziché sulla singola azienda. Alle tradizionali funzioni dell'impresa vengono affiancati ruoli e strutture più ampie, e la collaborazione tra gli attori porta alla possibilità di definire obiettivi e metriche comuni
- **Integrated:** la cooperazione viene estesa al livello del singolo processo, e le funzioni tradizionali iniziano a sparire. Le tecniche di pianificazione e previsione sono profondamente incorporate, e i costi di gestione diminuiscono mentre performance e soddisfazione del cliente aumentano
- **Extended:** il livello massimo di maturità della filiera, nella quale le aziende sono assimilabili ad un'unica entità

Il processo di evoluzione della gestione della catena di approvvigionamento è perciò lungo e dispendioso in termini di tempo e sforzo organizzativo, e richiede la volontà e la fiducia reciproca tra tutti i partner.

All'atto pratico, sono molteplici gli ostacoli e le problematiche da affrontare durante questo percorso, e vengono esacerbate dalla crescente dinamicità del mercato e dalla conseguente riduzione del lifecycle dei prodotti.

2.2 Sfide e problematiche nella Supply chain

Una delle problematiche più pressanti in una supply chain è rappresentata dalla necessità di tracciare il prodotto lungo tutta la filiera.

Normalmente, i sistemi di tracciamento sono limitati all'interno dell'azienda e ai passaggi della merce nei punti di snodo, causando vuoti sulla posizione e lo stato del prodotto[20].

Ciò rappresenta una pesante criticità in modo particolare in settori con severe legislazioni riguardanti la provenienza e la conservazione della merce, quali l'industria farmaceutica e quella agroalimentare.

Discorso analogo per la rintracciabilità del prodotto, processo parallelo al tracciamento volto a ricostruire lo storico di un prodotto, quindi a ritroso, lungo la filiera produttiva.

L'importanza di una adeguata rintracciabilità non si limita alla sola garanzia della compliance, ma fornisce un valore aggiunto al cliente, al quale vengono forniti gli strumenti per verificare la qualità e la provenienza del prodotto, e una riduzione nel rischio di frodi, con un miglioramento nella sua soddisfazione e fidelizzazione.

Ovviamente, la tracciabilità e in modo particolare la rintracciabilità sono processi complessi e articolati che richiedono grande sforzo inter-aziendale: in primis, è necessaria la coesistenza tra le pratiche aziendali e le regolamentazioni nazionali, che possono andare in conflitto non solo con quelle delle altre imprese, ma anche con la legislazione degli altri paesi in cui si svolge l'import/export.

L'altro punto critico riguarda la scarsità di soluzioni tecnologiche adeguate alle sfide delle supply chain odierne. Come già spiegato, il concetto di supply chain moderno è relativamente recente, e la maggioranza dei sistemi legacy sono rivolti alla gestione dei processi interni all'azienda. Inoltre, la scarsa interoperabilità aggrava lo scambio di informazioni, e spesso porta alla necessità di dover eseguire alcune attività di controllo e auditing in maniera manuale, con conseguenti rallentamenti e possibili errori.

Inoltre, una inadeguata visibilità lungo la supply chain ha effetti negativi non solo sulla qualità del prodotto in sé, ma anche sulla corretta gestione della sua domanda: una mancata previsione degli aumenti nei consumi, così come disguidi e rallentamenti nella produzione o nella spedizione possono portare a un approvvigionamento insufficiente al cliente finale.

D’altro canto, reazioni eccessive alle oscillazioni nella domanda possono causare un effetto farfalla che si ripercuote in maniera incrementale risalendo la filiera.

Questo evento, denominato effetto *Bullwhip*, deriva da una errata previsione della variazione della domanda basata sulle fluttuazioni del numero di ordini che porta a risposte spropositate nell’ambito dell’approvvigionamento dei prodotti, provocando un effetto a catena che aumenta di intensità tanto più si risale la catena di fornitori.

Le conseguenze sono un aumento dei costi di stoccaggio e di spedizione, diminuzione dell’efficienza nella produzione e rischio di esaurimento scorte[21].

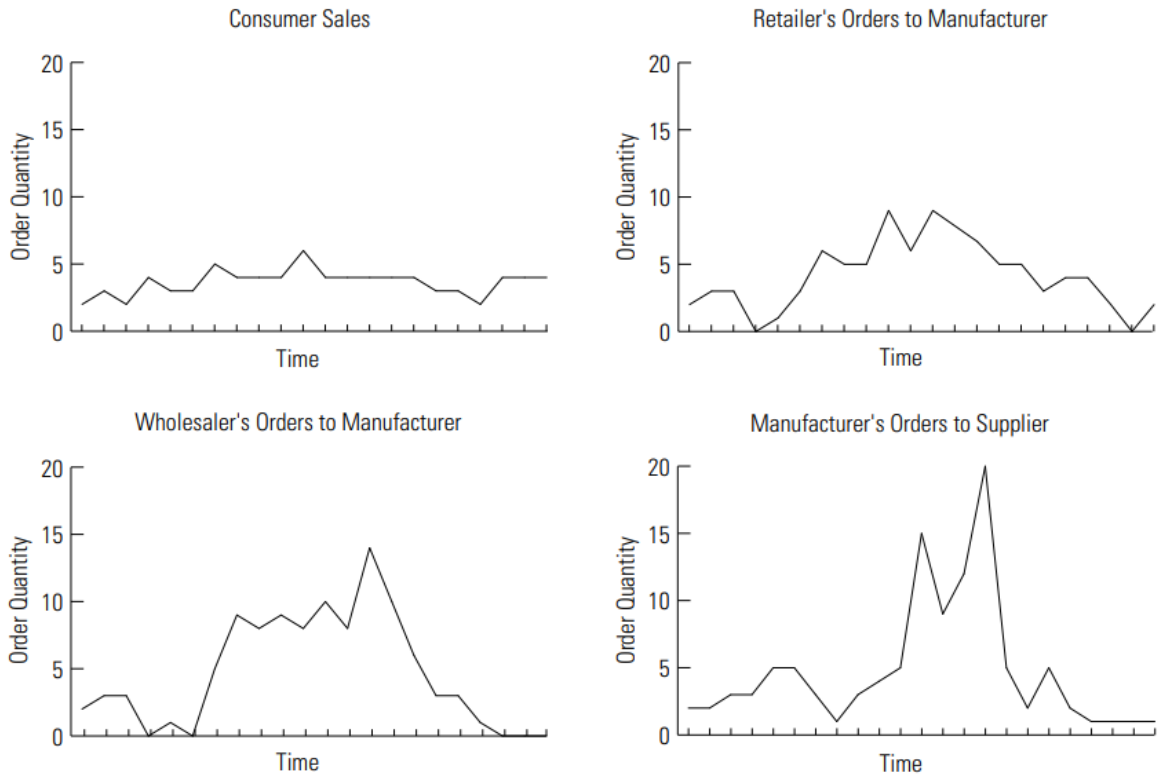

Figura 13 - Effetto Bullwhip[21]

Tralasciando questi punti, bisogna considerare come la premessa alla base della cooperazione sia la fiducia reciproca tra gli attori della supply chain. All'atto pratico, interessi conflittuali, la necessità di proteggere le informazioni sensibili ai competitors o il rischio di frodi e azioni illecite minano le opportunità di crescita della filiera.

Per ovviare ad una parte di queste problematiche è possibile fare affidamento ad un soggetto terzo imparziale che svolga la funzione di garante, denominato *trusted third party*, con il compito di definire le regolamentazioni per la gestione delle informazioni e processare tutte le transazioni così da fornire un set di dati comune sul quale tutti i partner possano essere d'accordo.

L'adozione di un intermediario presenta però diversi svantaggi: il più ovvio è il costo rappresentato dal compenso richiesto per il servizio offerto, ma vanno anche considerate le perdite derivanti dai ritardi introdotti dalle operazioni di certificazione. Inoltre, a seconda della credibilità dell'entità garante, non si può scartare a priori la possibilità di collusioni o falle nella sicurezza.

Un altro problema riguarda la gestione dei flussi finanziari, dato che, specialmente in caso di pagamenti classici attraverso fattura, sono diffusi ritardi o mancati compensi per le forniture. Nonostante i dati siano in miglioramento, secondo un report sui pagamenti delle imprese italiane nel terzo trimestre 2021, il valore delle fatture non pagate è pari al 10,5% di quelle scadute o in scadenza, mentre la percentuale di imprese che pagano i propri fornitori rispettando le scadenze concordate si attesta al 53,6%[22].

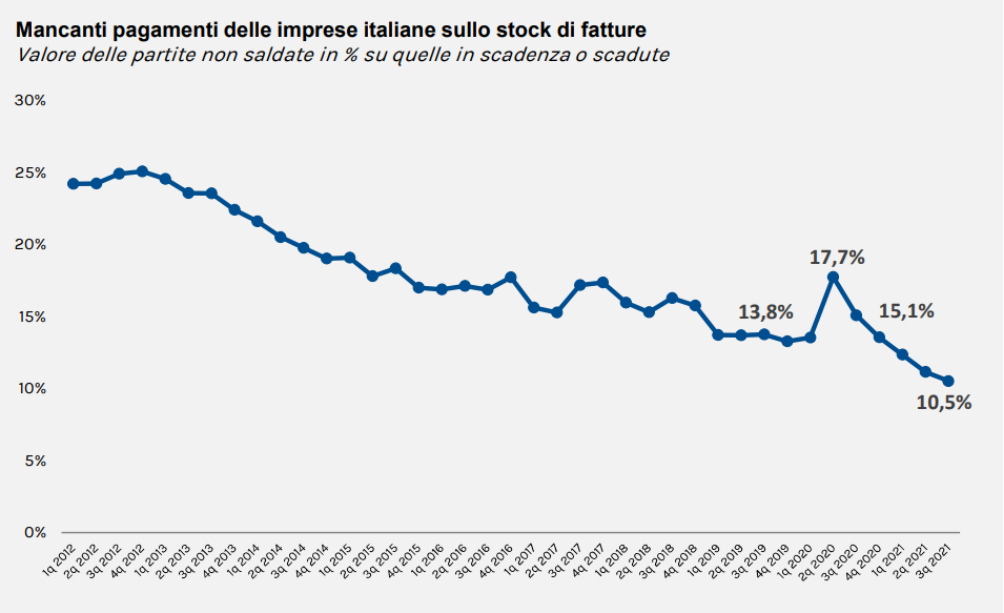

Figura 14 - Mancati pagamenti imprese italiane[22]

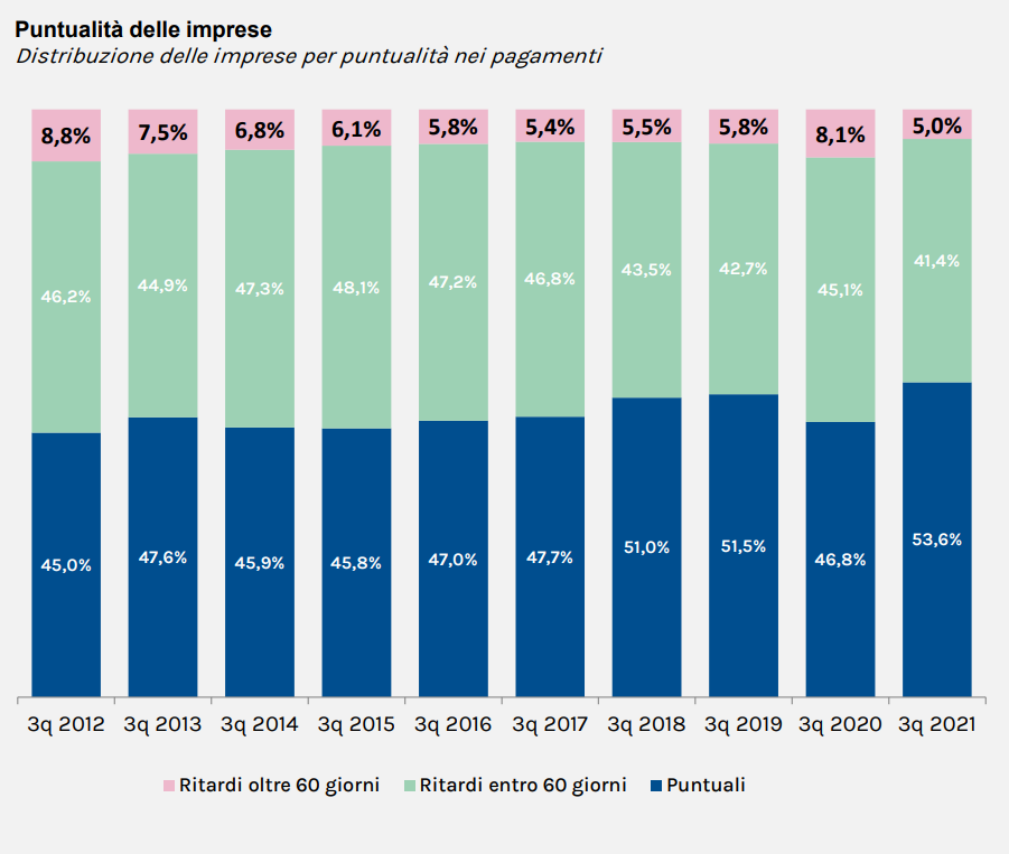

Figura 15 - Puntualità pagamenti imprese italiane[22]

Oltretutto risulta complessa la definizione delle responsabilità legali in caso di disguidi, così come il pagamento di penali per i fornitori al verificarsi di inottemperanze nei termini di contratto, e nel caso si rendesse necessario procedere per vie legali può essere di grande aiuto la raccolta minuziosa dei dati riguardanti la produzione, la spedizione e lo stoccaggio della merce.

2.3 Vantaggi della soluzione Blockchain nella Supply chain

Come già introdotto, una delle caratteristiche principali della Blockchain consiste nel fornire un database condiviso, immutabile e sicuro.

Se correttamente implementata da tutti gli attori della filiera, tale tecnologia porterebbe ad una transizione da un insieme di sistemi isolati e con scarsa capacità di comunicazione ad un ecosistema coeso, fornendo una visione globale della supply chain.

Ciò si tradurrebbe in una risoluzione dei problemi riguardanti la tracciabilità e rintracciabilità lungo la catena, dato che qualunque passaggio della merce registrato nel sistema viene reso disponibile a ogni partner e cliente, i quali vengono messi nella condizione di poter verificare stato e provenienza del prodotto, con impatti benefici sulla verifica della compliance, ritorno d'immagine e maggiore controllo sulle fonti di inefficienza e di frode.

Viene anche resa più semplice la verifica effettiva del volume di merce stoccata e venduta, portando a una risoluzione dei problemi derivanti dall'errata previsione della domanda e gestione dell'inventario, permettendo un corretto controllo quantitativo nella produzione.

A tal proposito, è opportuno introdurre i sistemi IoT come tecnologia complementare alla Blockchain in ambito di trasparenza.

L'*Internet of Things* si riferisce al processo di connessione alla rete di oggetti fisici. Nel dominio della supply chain, si possono elencare dispositivi quali sensori e *tag* RFID (*Radio Frequency IDentification*).

L'uso di sensori di temperatura, pressione, luminosità e via dicendo sono specialmente utili nelle già citate supply chain in cui la gestione dell'ambiente è un fattore particolarmente sensibile per una corretta conservazione della merce, mentre i sensori GPS possono aiutare a tracciare in maniera precisa la posizione in tempo reale durante la spedizione.

L’RFID è una tecnologia di identificazione basata sulla propagazione di onde elettromagnetiche. Le etichette RFID, composte da un chip e da un’antenna a radiofrequenza, sono codificate in modo univoco e non modificabile, mentre possono essere riscritte altre informazioni relative al prodotto in modo da favorirne il tracciamento.

Tali informazioni vengono acquisite e modificate senza bisogno di contatto da un lettore, che può interagire non solo sul singolo tag ma sull’intero lotto.

Questa tecnologia rappresenta quindi un’alternativa al classico codice a barre, più economico ma con capacità di storage molto più ridotta, necessità di scansione a breve distanza e vulnerabilità alla contraffazione.

Sensori e tag possono essere impiegati anche in assenza di una infrastruttura Blockchain, ma l’applicazione in questo contesto risolve i problemi di sicurezza e trasparenza derivanti dalla loro implementazione in sistemi centralizzati.

Grazie alla sua struttura *trustless*, la tecnologia Blockchain genera la fiducia necessaria alla cooperazione tra i partner, permettendo di bypassare la necessità di assumere una terza parte fidata e riducendo quindi i costi e le inefficienze burocratiche.

Inoltre, l’opportunità di utilizzo di smart contracts rende possibile l’automazione e lo snellimento delle attività di controllo e gestione finanziaria, ed evita interpretazioni soggettive dei termini di contratto.

Ad esempio, è possibile definire dei parametri limite a cui è possibile sottoporre la merce, ad esempio un delta massimo di temperatura. Se tali valori, raccolti dai sensori, superano la soglia prestabilita, è possibile segnalare l’annullamento del contratto e il pagamento automatico di una penale o il riordino del prodotto. Viceversa, se la consegna viene svolta entro i termini, il compenso può venire erogato in maniera autonoma.

Analogamente al discorso relativo agli strumenti IoT, gli smart contract rappresentano una tecnologia a sé stante, che trova però una naturale applicazione nella Blockchain dato che il contenuto non solo è visibile ad ogni nodo della catena, ma è protetto anche dalle modifiche a meno che queste non prevedano il consenso.

Infine, riguardo al discorso sulla previsione della domanda, l'impiego di tecniche di intelligenza artificiale può rappresentare uno strumento efficace nell'ottimizzazione degli acquisti e della gestione degli ordini, in quanto possono processare, analizzare e predire grandi moli di dati.

In più, possono discernere schemi e pattern in modo da poter definire strategie di marketing e produzione più efficaci[23].

Un esempio è rappresentato da *National Grid*, una compagnia energetica inglese che, attraverso la piattaforma *DeepMind* sviluppata da Google, può predire in modo accurato le variazioni nella domanda e nella offerta considerando anche variabili esogene quali il tempo atmosferico[24].

L'applicazione dell'IA in unione alla Blockchain risolve in parte la problematica riguardante la qualità dei dati in input. Infatti, affidarsi ad algoritmi di intelligenza artificiale utilizzando dati potenzialmente manipolabili può minare severamente la bontà dell'output, e la natura immutabile dei dati in Blockchain, ponendo come assodata la loro correttezza e l'utilizzo di algoritmi efficaci, garantirebbe buoni risultati predittivi.

2.4 Limiti e complicazioni nell'adozione della Blockchain

Finora si è discusso dei vantaggi che la Blockchain può apportare nell'ambito delle supply chain. A fronte di questi benefici, questa tecnologia è, ad oggi, afflitta da alcune criticità e limitazioni che devono essere adeguatamente considerate prima di valutarne l'adozione.

Una delle problematiche più rilevanti è la scarsa scalabilità, ovvero la capacità della rete di gestire grandi quantità di dati in un breve periodo di tempo. Vitalik Buterin, il co-fondatore di Ethereum, ha adattato il teorema CAP (*Consistency, Availability, Partition tolerance*) al contesto Blockchain, coniando il termine "trilemma della Blockchain", che indica la difficoltà nel combinare sicurezza, scalabilità e decentralizzazione, e di come, almeno al momento, se ne possano garantire soltanto due.

A titolo esemplificativo, le Blockchain pubbliche quali Bitcoin e Ethereum sono studiate per fornire decentralizzazione e sicurezza a scapito della scalabilità.

Infatti l'algoritmo di consenso impiegato, il Proof of Work, essendo particolarmente esoso in termini di sforzo computazionale, limita queste due piattaforme a poter processare rispettivamente una media di 5 e 15 transazioni al secondo[25], anche se Ethereum sta lavorando ad una implementazione dell'algoritmo Proof of Stake con l'aggiornamento a Eth2, per il quale si attende un miglioramento dell'ordine delle migliaia di transazioni al secondo[26].

Sotto questo aspetto, le Blockchain private o consortium hanno performance superiori, dato che rispetto a quelle pubbliche viene sacrificata, almeno parzialmente, la decentralizzazione.

Hyperledger Fabric sostiene la possibilità di ottenere nella pratica velocità di migliaia di transazioni al secondo, comunque ancora insufficiente per gestire i potenziali picchi nelle catene di fornitura più complesse[27].

Fa sperare in un miglioramento della situazione un esperimento che ha portato al risultato di scalare le transazioni al secondo di Fabric da 3000 a 20000, grazie a interventi di ottimizzazione sulla gestione dei metadati, sul parallelismo e sulla gestione della memoria[28].

L'altro problema legato alla scalabilità, e che influisce direttamente sul throughput, è la latenza, ovvero il lasso di tempo che intercorre tra la validazione di un nuovo blocco e la sua visibilità su tutti i nodi della rete.

Così come per le transazioni al secondo, la latenza viene affetta negativamente dalle attività di mining, e perciò le meno performanti da questo punto di vista sono nuovamente le Blockchain pubbliche.

Prendendo nuovamente ad esempio Bitcoin, la media del tempo di conferma per una transazione si attesta intorno a i 10-15 minuti, ma può anche raggiungere picchi di alcuni giorni[29]. Tali valori sono ben lontani dalla risposta quasi in tempo reale richiesta in supply chain, ed è quindi opportuno adottare soluzioni nelle quali le attività di mining siano limitate o assenti, come in quelle di tipo privato o consortium.

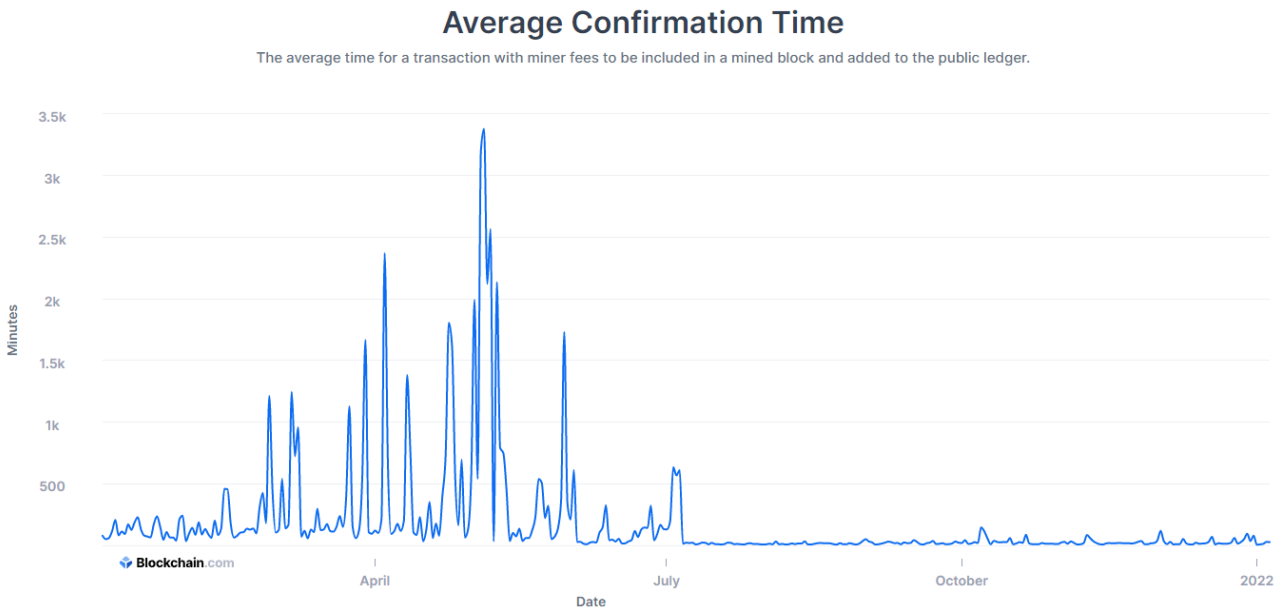

Figura 16 - Tempo di conferma medio delle transazioni in Bitcoin[29]

Un'eventuale soluzione al problema della scalabilità aggraverebbe però un'altra criticità della Blockchain, rappresentata dalla dimensione e dal consumo di risorse.

La sicurezza e affidabilità della Blockchain, uno dei maggiori punti di forza di questa tecnologia, è garantita grazie all'enorme quantità di ridondanza dei dati, che vengono memorizzati in ogni nodo della rete. Per questo motivo, con il passare del tempo e all'aumentare delle transazioni eseguite, incrementa sempre di più lo spazio di archiviazione richiesto per mantenere le copie del ledger.

Basti pensare che durante il 2021 la dimensione di Bitcoin è aumentata da 320 GB fino a superare i 380 GB, un incremento di circa il 20%[30].

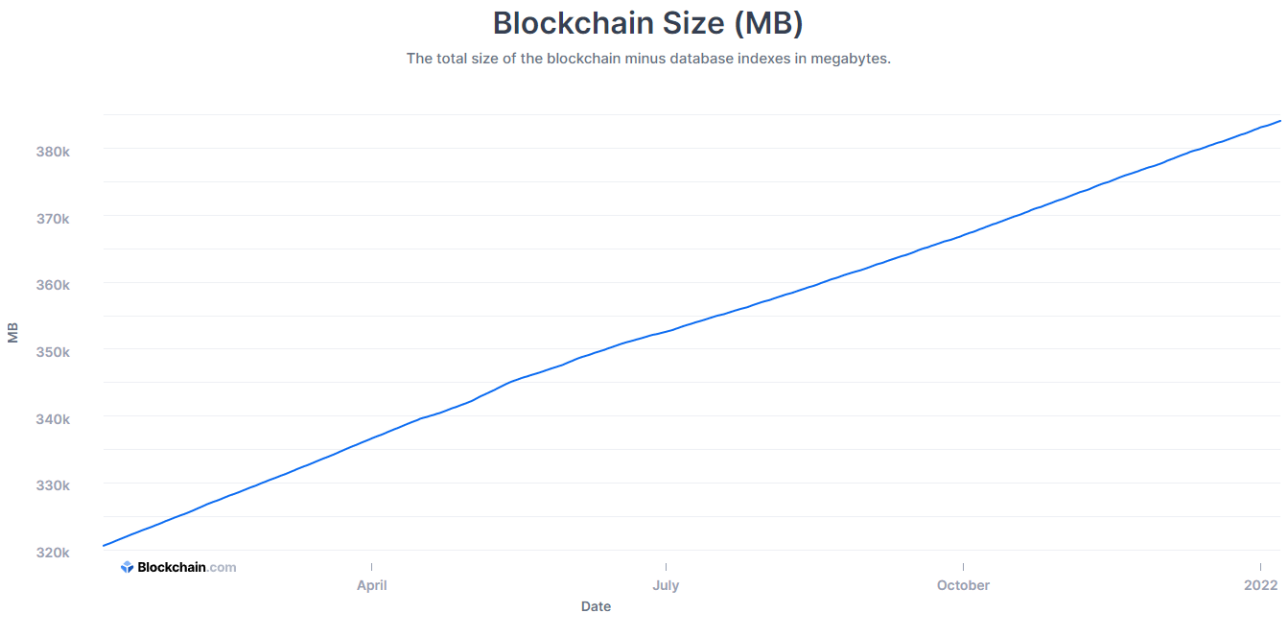

Figura 17 - Dimensione database Bitcoin[30]

Tutto ciò causa un forte impatto sui costi necessari per operare un nodo, sia in termini dell'hardware richiesto, sia per il consumo di banda, limitando di conseguenza la decentralizzazione e perciò la sicurezza della rete, così come la possibilità di sfruttare dispositivi *mobile* come nodi attivi.

Al momento non esiste una soluzione definitiva a questo problema. Comprimere i dati non apporta grandi benefici dato che la Blockchain contiene dati per loro natura incomprimibili, quali chiavi pubbliche e firme digitali, e il *pruning*, ovvero la rimozione di dati non strettamente rilevanti, nega al nodo "incompleto" la possibilità di partecipare al processo di validazione delle transazioni, oltre a rendere necessario affidarsi ai *full nodes* nel caso si voglia recuperarne lo storico.

Può portare a risultati più cospicui sia in termini di scalabilità che in riduzione della dimensione occupata ai nodi l'uso di tecniche di *sharding*, ovvero il partizionamento della rete in sottoinsiemi più piccoli in modo da ridurre il carico di lavoro per i nodi non direttamente coinvolti.

Uno dei punti di dibattito contro l'impiego sarebbe la riduzione della sicurezza, dato che il controllo sul consenso non risulterebbe più globale, ed è per questo che, specialmente per l'adozione in Blockchain pubbliche, necessita di ulteriore sviluppo e perfezionamento, mentre il problema è molto più contenuto per quelle private, essendo il controllo già affidato a nodi sicuri.

Ethereum, con Eth2, punta a introdurre le shard chains nel 2023[31], mentre è già disponibile sotto forma di canali per Hyperledger Fabric.

L'adozione in ambito supply chain introduce anche una criticità non presente nel contesto finanziario per il quale questa tecnologia è stata sviluppata. Nelle reti di criptovalute come Bitcoin e Ethereum, l'*asset* gestito è puramente digitale, e risiede solamente nella catena. Invece, la supply chain si occupa, durante le fasi di produzione, spedizione e stoccaggio, di beni fisici e del loro stato, e perciò si rende necessario l'uso di sistemi IoT da affiancare alla merce per far fronte a questo divario.

Ciò porta diversi svantaggi: in primis, un sensore esterno va considerato all'atto pratico come una trusted third party, dato che ogni attore deve fidarsi dei dati da esso forniti. Inoltre, essendo potenzialmente separati dal prodotto da monitorare, è possibile che si verifichino errori nella lettura, se non manomissioni vere e proprie.

L'aggiunta di sistemi collegati alla rete rappresenta anche un possibile vettore di attacchi informatici se non vengono impiegate le adeguate tecniche di sicurezza, e potrebbe portare alla creazione di backdoor nei sistemi aziendali o l'utilizzo per attacchi DoS e DDoS, con impatti economici e danno di immagine enormi.

Infine, i sensori forniscono una frequenza di raccolta dati e un grado di precisione variabile a seconda della qualità stessa del prodotto IoT impiegato, e l'impiego di versioni costose può risultare ingestibile considerando la necessità di affiancarli a ogni partita.

Di conseguenza, considerando l'importanza della validità dei dati in ingresso nella Blockchain, occorre certificare i dati provenienti dai sensori, e ciò richiede l'assunzione di un ulteriore intermediario, i cui costi però dovrebbero risultare contenuti rispetto ad un garante incaricato di gestire tutte le transazioni tra i partner.

Ulteriore problematica strutturale della Blockchain è la mancanza, al momento attuale, di interoperabilità tra le diverse piattaforme. Esistono soluzioni volte a connettere Blockchain operanti sulla stessa rete, come Polkadot[32] e Cosmos[33], ma non è possibile il passaggio di dati, ad esempio, tra Ethereum e Bitcoin.

Ciò non rappresenta un problema finché l'interazione avviene tra i soli partner della supply chain, ma può diventare limitante in caso di partnership temporanee o fusioni tra catene di fornitura diverse che usufruiscono di una loro rete Blockchain, dal momento che ricostruire l'infrastruttura non rappresenta una soluzione sostenibile.

3. Caso studio

L'obiettivo di questo lavoro di tesi è quello di studiare l'applicabilità della tecnologia nel settore delle supply chain. A tale scopo è stato definito, in collaborazione con *Stellantis*, un caso d'uso semplificato che funga da banco di prova per la fattibilità dell'implementazione e, in caso di risultati soddisfacenti, valutarne un successivo sviluppo per il *deploy* su campo o una sua estensione ad altri casi d'uso.

Il *case study* in questione consiste in una catena d'approvvigionamento votata alla produzione di batterie per autovetture elettriche.

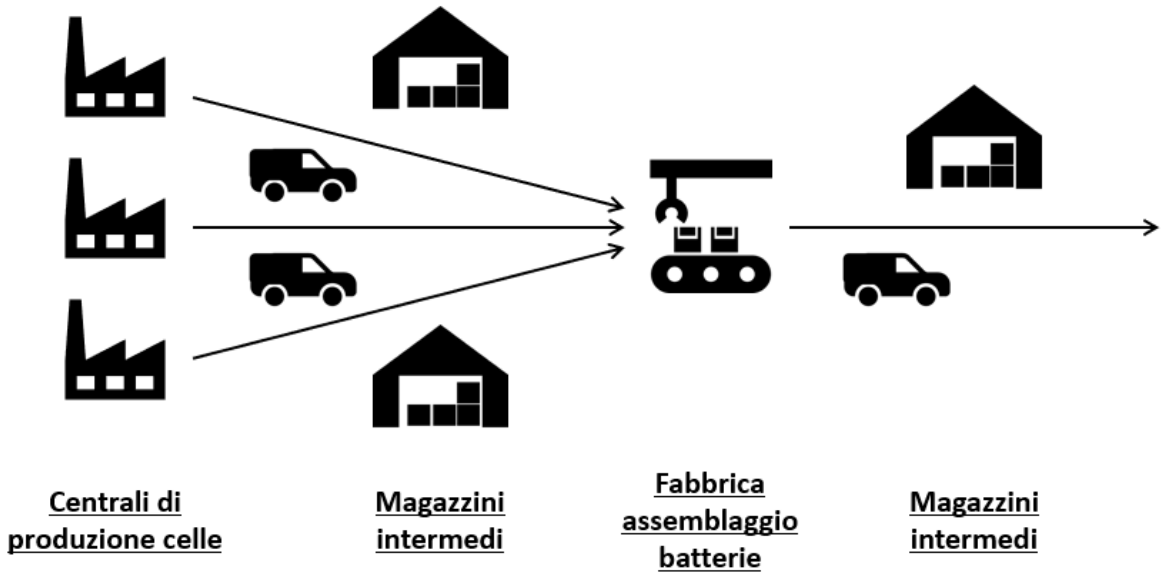

Figura 18 - Caso studio

Nelle sezioni successive verranno elencati gli attori del caso studio, il loro ruolo nella supply chain, così come le problematiche che tali aziende devono affrontare nella loro cooperazione, per poi valutare se la Blockchain rappresenti una soluzione adatta a tale scopo.

3.1 Attori

Il primo passo nel valutare l'utilità della Blockchain è identificare le parti interessate e i loro obiettivi.

Queste si possono riassumere in quattro entità distinte:

- **Produttore di celle per batteria:** si occupa della produzione degli elementi necessari alla creazione di batterie per automobili elettriche. Le unità prodotte vengono spedite all'azienda automobilistica attraverso i servizi di un operatore logistico. Non solo è tenuto a garantire la qualità dei pezzi, ma è anche responsabile dei processi di spedizione e stoccaggio
- **Casa automobilistica:** procede all'assemblaggio dei pezzi ricevuti dal produttore. Deve assicurarsi della qualità degli elementi acquistati, così come del loro approvvigionamento in conformità con le tempistiche richieste, oltre che della qualità del prodotto finito
- **Magazzino intermedio:** gestisce lo stoccaggio dei prodotti per conto dell'operatore logistico. È responsabile dell'immagazzinamento e della conservazione della merce
- **Operatore logistico:** si occupa della spedizione della merce da e verso le centrali di produzione e i magazzini intermedi. È quindi responsabile dei prodotti durante il loro trasporto

3.2 Processo

Ciò su cui ci si concentra in questo studio è quindi una parte della supply chain, ovvero il processo aziendale che porta dalla creazione delle celle all'assemblaggio delle batterie. Non vengono quindi considerate le fasi inerenti alla produzione definitiva dell'autovettura o la sua commercializzazione.

È possibile perciò dividere il processo nelle seguenti attività:

- **Produttore celle** => creazione elementi per batterie
- **Casa automobilistica** => acquisto pezzi dal produttore
- **Produttore celle** => richiesta servizio di spedizione all'operatore logistico
- **Operatore logistico** => richiesta servizio di stoccaggio al magazzino intermedio
- **Operatore logistico** => spedizione merci
- **Magazzino intermedio** => stoccaggio prodotti
- **Casa automobilistica** => assemblaggio batterie

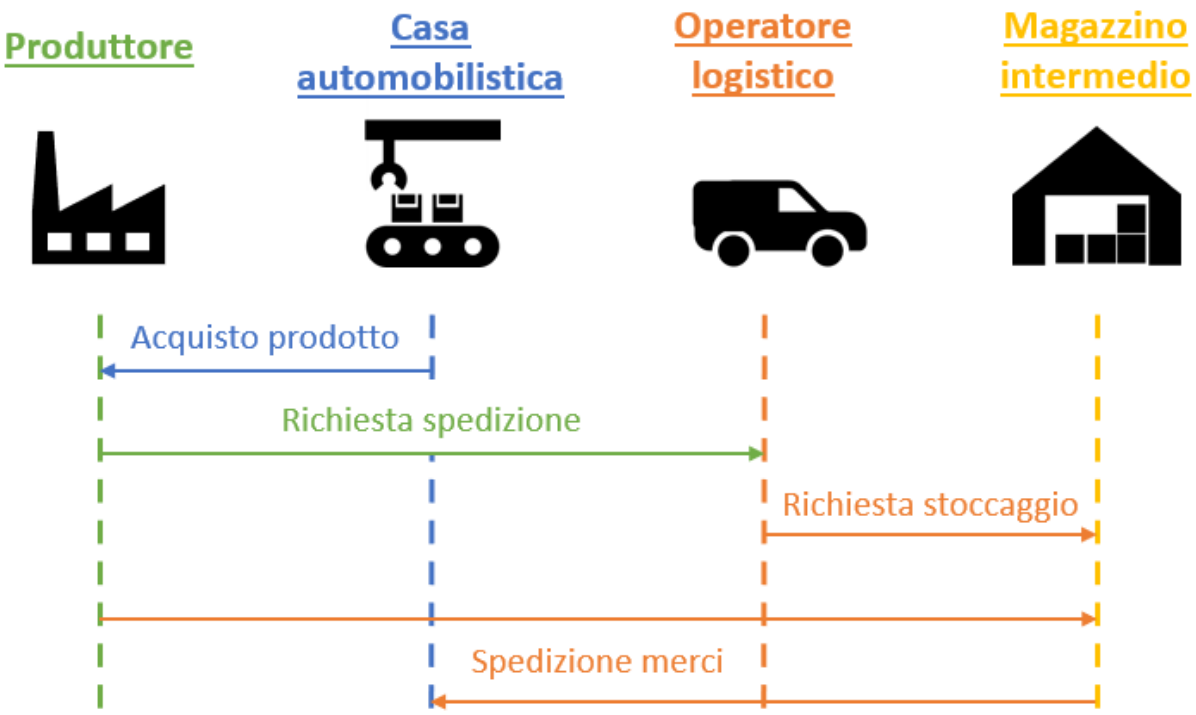

Figura 19 - Processo caso d'uso

3.3 Requisiti

Dovendo difendere i propri interessi, le imprese in gioco non possono fidarsi incondizionatamente l'una dell'altra, e si aspettano perciò delle garanzie riguardanti le attività interaziendali del processo.

La casa automobilistica richiede non solo il mantenimento degli standard qualitativi degli elementi acquistati e che questi non siano stati danneggiati durante il viaggio, ma anche che la loro spedizione sia puntuale in modo da non dover interrompere la propria produzione. Non potendo constatare anticipatamente il rispetto di questi requisiti, l'azienda pretende che il pagamento sia effettuato successivamente alla consegna.

Essendo quindi il produttore delle celle a dover garantire non solo la qualità del prodotto, ma anche che la spedizione vada a buon fine, vorrebbe la sicurezza che il distributore a cui si è affidato si comporti correttamente. Inoltre, vuole assicurarsi che al termine della transazione riceva dalla casa automobilistica il compenso pattuito entro i termini stabiliti.

Può essere fatto un discorso analogo per il fornitore di servizi logistici, che ha la necessità di monitorare i magazzini di stoccaggio e garantire il pagamento da parte del produttore.

Infine, i magazzini intermedi vogliono la sicurezza di venire compensati dall'operatore logistico per i propri servizi.

Ne deriva che gli attori hanno forte interesse a lavorare insieme, ma la cooperazione tra di loro può venire limitata da problemi di fiducia.

3.4 Introduzione della Blockchain nel caso d'uso

L'adozione della tecnologia Blockchain può risolvere la gran parte delle problematiche riguardanti l'affidabilità tra gli attori.

In primis, è possibile tracciare in tempo reale lo stato della merce attraverso dei sensori, e salvando tali informazioni sulla Blockchain le si rendono disponibili a tutti i partner interessati. In aggiunta ai dati sulla posizione, quindi latitudine e longitudine, è possibile verificare il corretto mantenimento dell'ambiente di conservazione monitorando temperatura, umidità, pressione e luminosità a cui sono sottoposti i pezzi.

In questo modo si ha un riscontro immediato in caso di malagestione del prodotto, fornendo dati indiscutibili per il controllo della responsabilità in caso di danneggiamento, furti o ritardi.

I sensori devono quindi risultare a prova di manomissione per essere definiti come affidabili. Il loro corretto funzionamento e resilienza agli attacchi esterni deve essere certificato dal produttore, mentre durante l'uso bisogna garantire la loro non-manomissione, ad esempio sigillandoli insieme al pezzo che devono monitorare: nel caso in cui il sigillo venga rotto o si rilevi una differenza anomala nei dati, come un aumento della luminosità, lo stato dell'elemento viene segnalato come invalido. Può essere utile anche l'impiego di controllori che raccolgano i dati in arrivo dai sensori e li inoltrino in modo sicuro ai nodi della Blockchain.

Viene risolta anche la problematica riguardante i pagamenti, dato che è possibile impostare il sistema in modo tale che il corrispettivo per i servizi offerti venga emesso in modo automatico al termine con successo della transazione, così come il versamento di eventuali penali in caso di inadempienza.

Le modalità di tali pagamenti variano a seconda della Blockchain scelta, in quanto ve ne sono alcune con una criptovaluta propria, mentre altre ne sono sprovviste, ed è necessario quindi utilizzare altre forme di pagamento, come ad esempio via bonifico o impostando un sistema di token proprietario.

Una volta che le parti in gioco acconsentono all'uso della Blockchain, è necessario standardizzare il formato dei dati e la loro metodologia di raccolta e lettura. Fatto ciò, è possibile procedere alla stesura dello smart contract.

Si possono definire a grandi linee le funzioni del contratto e la struttura generale del sistema: la prima funzione del processo sarebbe l'invio di una richiesta d'acquisto dalla casa automobilistica al produttore delle celle, e l'inoltro al fornitore di servizi logistici e ai magazzini.

Fornendo una visione semplificata del sistema, la transazione comprenderebbe i compensi in caso di successo e le penali, data di inizio e scadenza della spedizione e stato del contratto, mentre il prodotto verrebbe modellato sulla Blockchain con un identificativo, i parametri raccolti dai sensori e l'entità attualmente responsabile. Inoltre, le operazioni eseguibili devono essere limitate a seconda del ruolo associato all'indirizzo che le esegue, così da garantire il controllo degli accessi.

In caso di eventi anomali, lo stato del pezzo verrebbe segnato come invalido, annullando la spedizione e facendo pagare le penali pattuite all'azienda che ne detiene attualmente la responsabilità. Se non si verifica nessun errore e la spedizione termina entro la scadenza, vengono effettuati i pagamenti richiesti.

Inoltre, tali dati sono visibili in tempo reale per tutti gli attori abilitati, fornendo precise informazioni sullo stato della spedizione e consentendo azioni di ripristino tempestive in caso di ritardi o interruzioni.

Essendo il progetto ancora su piccola scala, la rete impiegata sarebbe verosimilmente di dimensione ridotta, con circa una decina di nodi detenuti dalle entità.

4. Framework Blockchain

Modellato il caso d'uso e descritti i requisiti, è possibile procedere con la scelta del framework più adatto allo scopo.

Tale decisione presenta molteplici fattori da valutare, come ad esempio l'efficienza e la scalabilità, la sicurezza e la privacy fornita, la maturità del progetto e il supporto offerto in termini di documentazione e tutorial, così come la flessibilità concessa nel suo utilizzo.

Il primo passo riguarda la definizione della tipologia di Blockchain che verrà impiegata.

4.1 Tipologia di Blockchain adatta

Si può scartare a priori la possibilità di utilizzare una Blockchain pubblica: la limitazione più grande di questa tipologia è rappresentata dalle performance, dato che lo scarso throughput e l'alta latenza mal si prestano alla necessità di update delle informazioni in tempo reale.

Inoltre, in una Blockchain pubblica non è possibile limitare la visibilità delle informazioni sensibili e non esistono restrizioni riguardanti l'accesso al sistema: ciò causerebbe problemi legati alla privacy e alla protezione dei dati dai *competitors*.

Queste problematiche sono risolvibili con l'utilizzo di una Blockchain privata o una ibrida, grazie alle loro performance superiori e alla possibilità di controllare gli accessi. Inoltre, nel caso di quest'ultima, è possibile rivelare le informazioni necessarie a garantire una eventuale compliance alle normative di privacy o di produzione attraverso uno smart contract adeguato allo scopo.

Come già esposto, però, questo tipo di reti sono assimilabili a sistemi centralizzati controllati da un'unica entità, e non risolverebbero quindi la questione riguardante la fiducia tra gli attori della supply chain.

La soluzione più adatta a tale scopo è quindi rappresentata da una Blockchain di tipo consortium, il cui controllo è condiviso tra i soli membri della supply chain. Questa tipologia offre buone prestazioni rispetto ad una pubblica e controllo sull'accesso alla rete.

L'impiego di una Blockchain consortium implica il pagamento di costi fissi relativi al mantenimento dell'infrastruttura, che vengono divisi tra i gestori dei nodi. Potendo

considerare l'attività del caso d'uso come continuativa, il bilancio tra le spese di gestione e il risparmio derivante dall'impiego della Blockchain, dato dall'assenza di intermediari necessari alla mediazione delle transazioni, riduzione degli errori e delle frodi, miglioramento nella capacità decisionale, snellimento della documentazione e generale aumento dell'efficienza, risulterebbe positivo.

È importante ricordare come non esista una soluzione univoca e ottimale, ma come la scelta della Blockchain più adatta vada considerata caso per caso tenendo conto dei bisogni e delle limitazioni ricavati dall'analisi dei processi aziendali. A titolo esemplificativo, considerando un caso d'uso nel quale l'attività produttiva si concentri in periodi specifici, come nella vendita di prodotti stagionali, una soluzione di tipo consortium potrebbe non rappresentare una opzione ottimale, dato che la rete rimarrebbe inutilizzata per gran parte del tempo mentre i suoi costi di gestione rimarrebbero costanti.

4.2 Framework industriali

In seguito alla fase preliminare di decisione della tipologia di Blockchain, sono stati considerati quattro tra i più supportati framework Blockchain di livello industriale che permettono l'implementazione di una rete consortium: *Hyperledger Fabric, Hyperledger Besu, Hyperledger Sawtooth* e *Quorum*.

- **Fabric:** così come Besu e Sawtooth, Fabric è un progetto appartenente all'ecosistema Hyperledger, un progetto open-source avviato nel 2015 dalla *Linux Foundation* allo scopo di sviluppare framework e tools per Blockchain di livello industriale[34].
  È una piattaforma altamente modulare progettata per la creazione di reti private in ambito aziendale. È quindi un sistema permissioned, e i membri della rete sono tenuti a registrarsi attraverso un *Membership Service Provider (MSP)* fidato.
  Offre diverse funzionalità quali l'invio di transazioni private e la possibilità di creare canali, ovvero catene separate visibili solo a un gruppo ristretto di partecipanti, particolarmente utile per nascondere informazioni a eventuali competitors.
  Il sistema del ledger è composto da due elementi: il *world state*, che ne descrive lo stato, e il log delle transazioni, che rappresenta lo storico degli aggiornamenti al world state.

Generalmente gli smart contract, definiti all'interno di un *chaincode* che ne gestisce il deployment sulla Blockchain, interagiscono solo con il world state, e non con il log. Attualmente i linguaggi ufficialmente supportati per la scrittura di un chaincode sono Go, Node.js e Java[35].

Al momento, Fabric è limitato dall'assenza di un'implementazione ufficiale di un algoritmo di consenso BFT, così come da un setup relativamente difficile rispetto ai competitors. Essendo però il primo progetto di Hyperledger risulta ben documentato, ed è probabilmente il framework più testato per l'uso in produzione

- **Besu:** Besu è un client Ethereum open source, ovvero un software che implementa il protocollo Ethereum.

  È utilizzabile non solo sulla rete pubblica di Ethereum, ma anche in reti private e reti di test come Rinkeby, Ropsten e Görli (utili durante le fasi di sviluppo in modo da non impiegare criptovalute reali per l'esecuzione di transazioni).

  Supporta molteplici algoritmi di consenso quali PoW, varianti di PoA come IBFT e Clique, e comprende schemi di permissioning specifici per reti consortium[36]

- **Sawtooth:** Sawtooth è una piattaforma per la creazione di reti permissioned e permissionless che fornisce una divisione netta tra il livello applicativo e il sistema principale, semplificando lo sviluppo e il deployment di un programma sulla Blockchain.

  È infatti possibile specificare la business logic utilizzando un linguaggio di propria scelta, senza la necessità di conoscere il funzionamento del sistema sottostante, e permette a diversi tipi di applicazione di co-esistere nella medesima istanza di rete.

  Include anche uno scheduler che divide le transazioni in flussi paralleli, fornendo un sostanziale aumento prestazionale rispetto all'esecuzione seriale[37].

  È molto flessibile e ben documentato, ma di base non supporta l'uso di transazioni private

- **Quorum**: nato come fork di Ethereum, Quorum permette la creazione di reti private e la definizione di regole per l'accesso a parti di dati visibili a un sottoinsieme di nodi. Supporta quindi transazioni private oltre che pubbliche, e riguardo agli algoritmi di consenso offre sia CFT che BFT.

Sono state sviluppate due suite di questo progetto, una basata sull'originale implementazione di J.P. Morgan e composta dal client Ethereum GoQuorum e dal modulo Tessera per la gestione delle transazioni private[38], e una che sfrutta Besu come client e impiega Orion come modulo per le transazioni private[39], ma recentemente Orion è stato completamente assorbito da Tessera, creando un singolo codebase[40]

Nelle sezioni successive vengono comparate le soluzioni proposte, sfruttando come metriche le seguenti dimensioni:

- **Efficienza e scalabilità:** la quantità di dati processabili nell'unità di tempo e l'impatto sulle prestazioni derivante dall'incremento della dimensione della rete
- **Resilienza e finalità:** la resistenza a crash e azioni malevole e la vulnerabilità ai fork
- **Privacy:** la possibilità di condividere informazioni con un solo sottoinsieme della rete
- **Maturità e supporto:** indica se il framework è *production-ready* e quanto questo sia utilizzato
- **Flessibilità:** modularità e adattabilità del sistema

4.3 Efficienza e scalabilità

I test sulle performance dei framework sono stati svolti dal *ICELab@POLITO* in collaborazione con TIM sfruttando un simulatore da loro sviluppato. È importante tenere a mente che i risultati presentati sono dipendenti dall'ambiente di prova e dai parametri utilizzati, e potrebbero differire al variare di questi.

Il primo test misura l'impatto sulle TPS variando il numero di strutture dati a cui viene fatto accesso da 1, ovvero un accesso sequenziale, a un valore massimo pari al numero di transazioni eseguite.

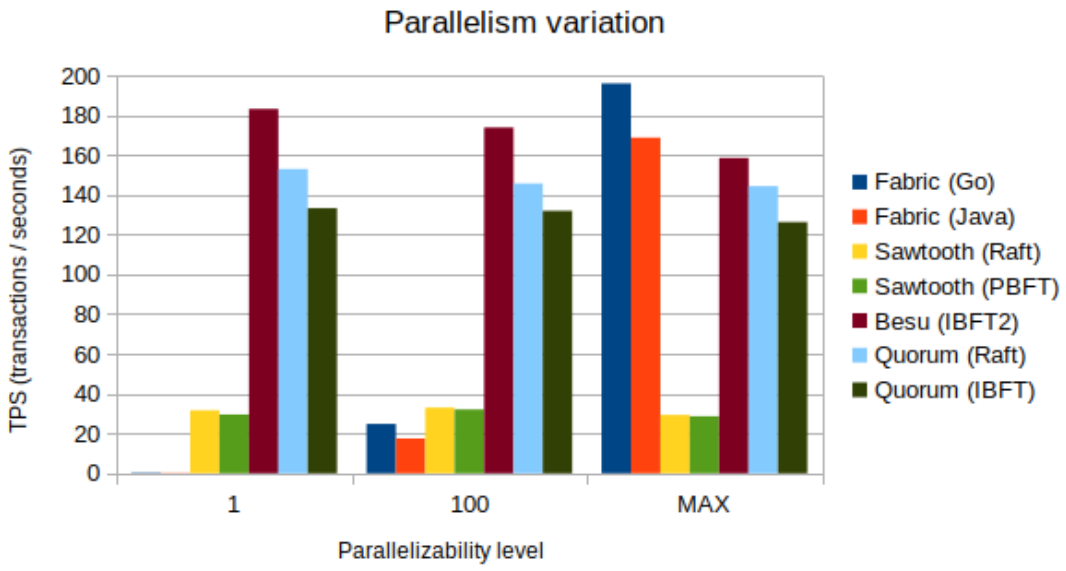

Figura 20 - Comparazione parallelismo

Come ci si può aspettare, l'algoritmo di consenso CFT (ovvero Raft) ha prestazioni superiori rispetto alla controparte BFT.

Dal grafico si può notare come Fabric scali in modo eccellente all'aumentare del parallelismo, ma soffra pesantemente in caso di esecuzione sequenziale. Ciò è dovuto al paradigma *Execute-Order-Validate,* dove la transazione viene prima eseguita da un numero predefinito di nodi e i risultati, se consistenti, vengono ordinati e racchiusi in blocchi da un servizio specifico, che infine li propagherà a tutti i nodi della rete. Si può anche osservare come la versione scritta con Go sia più efficiente rispetto a quella Java.

Le performance degli altri framework rimangono contenute a causa dell'impiego del classico flusso d'esecuzione delle transazioni, *l'Order-Execute*, nel quale la rete ordina le transazioni prima di eseguirle sui peer in modo sequenziale.

Il test successivo verte sulla variazione della quantità di operazioni eseguite nella stessa transazione, da 1 (update singolo) a 10^3 (update di una lista).

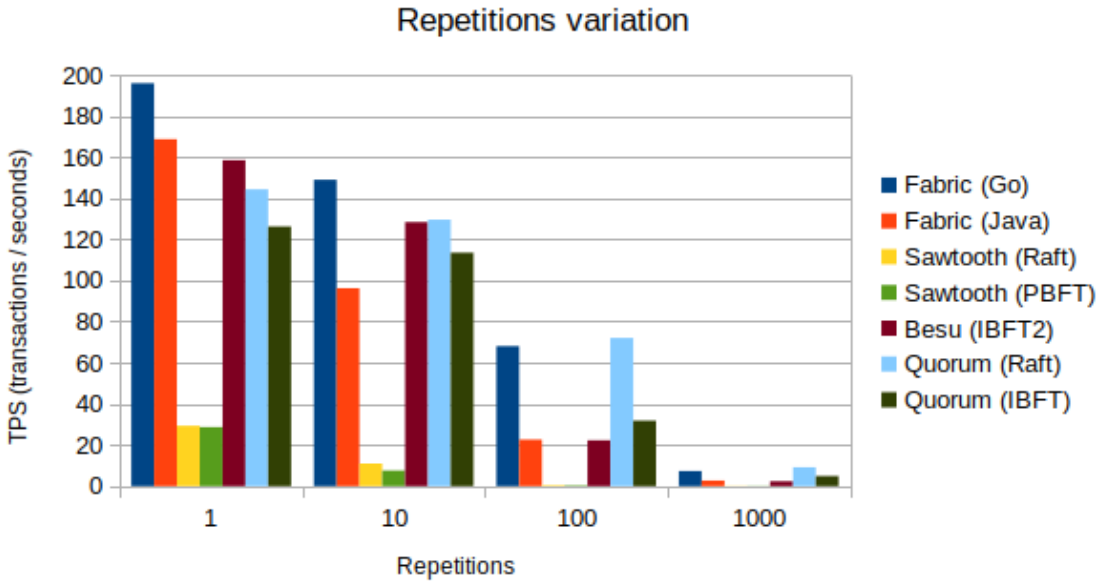

Figura 21 - Comparazione ripetizioni

In questo test Quorum sembra il meno affetto dall'aumento di ripetizioni rispetto agli altri framework. Va però notato che, nonostante il decremento del TPS, in numero di read/write eseguite aumenta grazie al maggior numero di operazioni per transazione.

L'ultimo test varia la dimensione della stringa in input, compresa tra 10^{-1} e 50 KB.

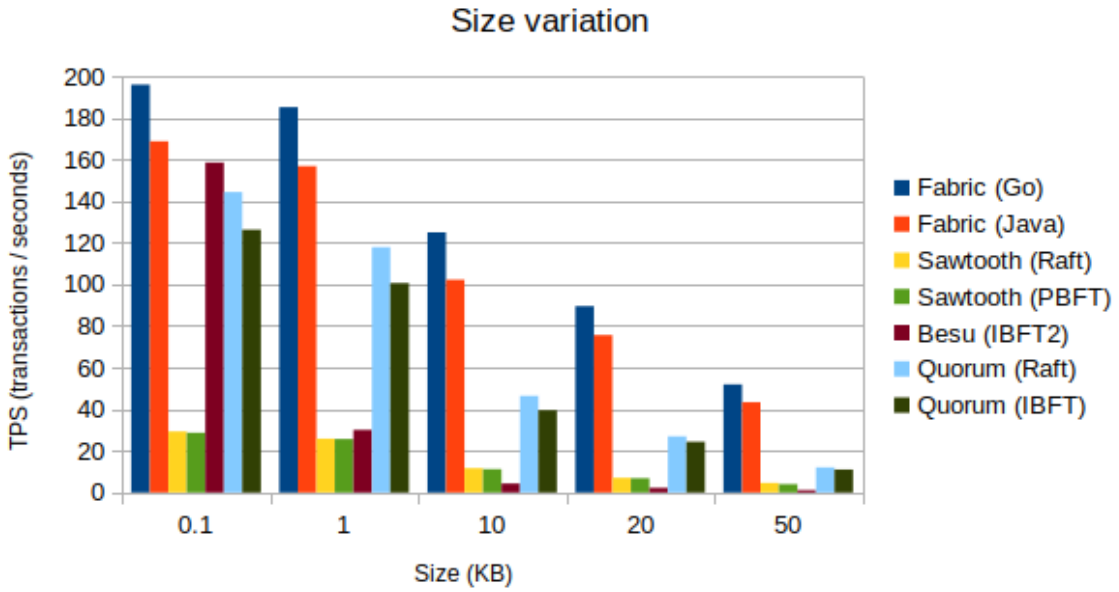

Figura 22 - Comparazione dimensione

Besu risulta il più suscettibile all'aumento delle dimensioni dell'input, specialmente se queste superano 0,1 KB. Come per le ripetizioni, vi è in generale un aumento della quantità di dati letti o scritti data la maggior dimensione delle transazioni.

Va considerato però che in un contesto reale le transazioni sarebbero verosimilmente di piccole dimensioni.

4.4 Resilienza e finalità

La resilienza del sistema dipende dall'algoritmo di consenso utilizzato: Sawtooth supporta sia CFT con Raft e *Proof-of-Elapsed-Time* (*PoET*), sia BFT con PBFT e PoET se sfruttato in un *Trusted Execution Environment* (*TEE*), ovvero un'area sicura del processore[41]. Tale impostazione richiede l'impiego di hardware specifico, al momento fornito solo da *Intel* via *Software Guard Extensions* (*SGX*), e data la presenza di alcune vulnerabilità su questi moduli il suo uso è sconsigliato[42].

Besu e Quorum implementano BFT attraverso Ethash, Clique, IBFT e IBFT 2.0, con un supporto da parte di Quorum ai CFT con Raft[43-44].

Fabric è il più limitato, fornendo al momento unicamente Raft come algoritmo per il servizio di ordinamento[45]. A meno di creare una propria implementazione di BFT, il sistema risulta vulnerabile.

Riguardo la finalità, Raft, IBFT e PBFT sono deterministici, quindi non affetti dal problema dei fork, mentre PoET, Clique e Ethash sono probabilistici.

4.5 Privacy

Quorum e Besu supportano Tessera per l'invio di transazioni criptate point-to-point.

Fabric offre sia la possibilità di creare canali accessibili a un numero ristretto di partecipanti, sia di creare raccolte di dati private, con un funzionamento simile al modulo Tessera e senza l'overhead amministrativo necessario alla gestione di un canale[46].

Sawtooth non offre una soluzione ufficiale al problema, ed è quindi necessario svilupparne una o creare una rete per ogni sottoinsieme di nodi, provocando però una riduzione nella sicurezza.

4.6 Maturità e supporto

Tutte e quattro le soluzioni sono considerabili come production-ready, e possiedono un buon supporto sia su canali ufficiali che in termini di community.

Essendo però Fabric il progetto più datato, è stato testato in modo più estensivo ed è maggiormente supportato.

La figura sottostante mostra l'attività su GitHub dei framework elencati.

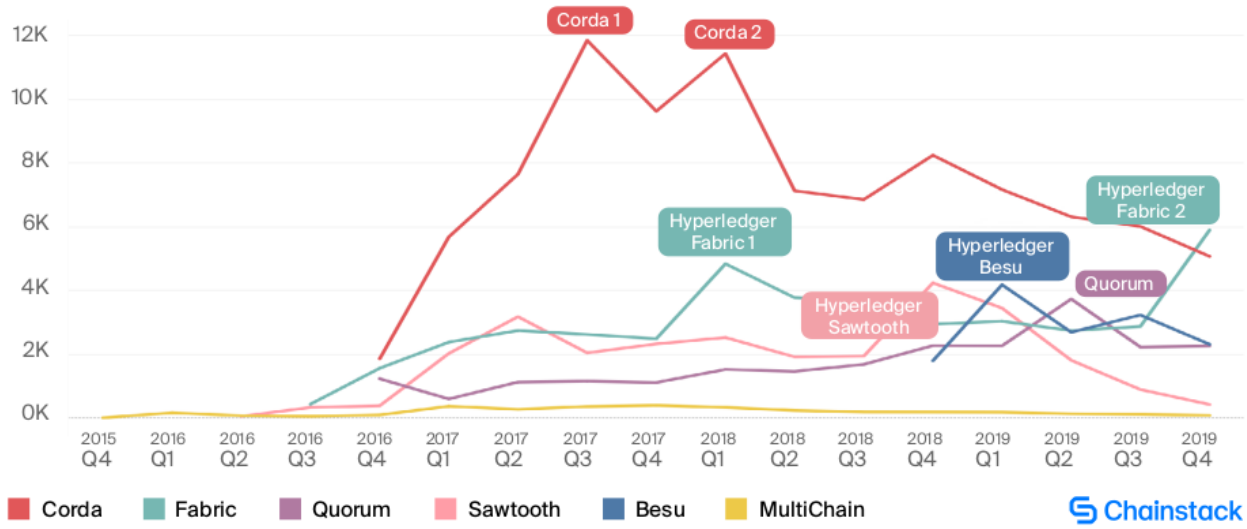

Figura 23 - Attività framework su GitHub[47]

4.7 Flessibilità

Sawtooth rappresenta la scelta più flessibile, data la sua architettura che permette l'aggiornamento dinamico dei moduli che lo compongono.

Le altre alternative rappresentano una scelta peggiore sotto questo parametro: Fabric permette la scrittura degli smart contract utilizzando alcuni tra i linguaggi più diffusi come Go e Javascript, mentre Quorum e Besu supportano principalmente Solidity.

4.8 Conclusioni sulla scelta del framework

Riassumendo i risultati si può concludere che nel caso sia necessaria una soluzione capace di scalare orizzontalmente, Fabric rappresenta l'unica soluzione affidabile.

Nel caso la modularità sia un fattore determinante, Sawtooth è il framework più adatto.

Besu e Quorum si possono considerare pressoché intercambiabili, e offrono un buon compromesso tra efficienza, scalabilità e resilienza.

Nel contesto del caso studio proposto, Besu o Quorum rappresentano la scelta migliore: Fabric è particolarmente ostico da impostare, e richiede l'implementazione di un algoritmo BFT o l'impiego di una terza parte fidata, mentre le performance di Sawtooth e l'assenza di soluzioni proprietarie per la privacy non sono adeguate allo scopo.

Essendo il numero di nodi non superiore a dieci, la scalabilità del framework non è una problematica, ed è perciò possibile l'impiego di algoritmi con performance particolarmente suscettibili alle dimensioni della rete, come ad esempio l'IBFT.

5. Studio di fattibilità

L'obiettivo di questa fase consiste nel testare se una rete Blockchain sia adatta a sopportare il carico di lavoro richiesto dal processo, se sia possibile saturare le risorse del sistema e in generale definire una struttura del codice che possa incrementare le prestazioni nelle operazioni di salvataggio delle misure.

A tale scopo, sono stati forniti da Stellantis i dati provenienti dai sistemi IoT connessi attraverso rete mobile, utilizzati nel sistema attuale. Da questi è stato possibile estrarre un esempio di formato di dato, e di conseguenza la dimensione dello stesso, così come una media sul volume di transazioni generate.

È importante precisare che i test svolti hanno lo scopo di testare la risposta del sistema al variare dei parametri maggiormente controllabili dal programmatore, quali la codifica dei dati e la struttura dello smart contract.

I risultati non vanno quindi elaborati sotto un'ottica quantitativa, dato che i limiti della rete sono profondamente dipendenti dalla configurazione hardware impiegata, ma andrebbero considerati in maniera qualitativa, in modo tale da favorire l'ottimizzazione del codice e massimizzare il più possibile il throughput nel caso si decida di portare avanti il progetto.

5.1 Ambiente di test

Il framework scelto per i test è Hyperledger Besu con protocollo di consenso IBFT 2.0.

La rete privata è composta da quattro nodi validatori, il numero minimo necessario richiesto dall'algoritmo per garantire la Byzantine Fault Tolerance, schierati all'interno di un contenitore Docker in modo tale da velocizzarne il *deployment*.

Il costo per unità di *gas*, e di conseguenza il costo per transazione, è stato impostato a zero, ed è verosimile che rimanga tale anche in produzione data la struttura consortium della rete. Il gas è una rappresentazione numerica del costo computazionale richiesto per l'esecuzione di una transazione, e nell'ambito dei test svolti rappresenta un valido parametro per testare l'ottimizzazione dei contratti.

L'interazione con i nodi avviene attraverso l'esposizione di un endpoint *JSON-RPC* (*JSON-Remote Procedure Call*) e/o *WebSocket*. È stato necessario sfruttare quest'ultimo, dato che consente la comunicazione attraverso una singola connessione TCP. Con la libreria in uso, l'impiego di JSON-RPC porta all'apertura di una nuova connessione a ogni transazione, e nel caso in cui ne vengano inviate molteplici in un breve periodo di tempo ciò causa la creazione di istanze concorrenti e l'eventuale rifiuto di nuove da parte dei nodi quando queste superano un certo numero.

Nel contesto delle transazioni, esse sono inviate tramite script scritti in linguaggio *JavaScript* appoggiandosi alla libreria *Web3.js*, che consente l'interazione con i nodi locali o remoti di una rete basata su Ethereum.

Tali script vengono lanciati da un modulo Raspberry Pi 4 collegato alla stessa rete LAN dei nodi, in modo tale da simulare l'invio delle misure provenienti da un sensore verso la Blockchain.

5.2 Codifiche dei dati

Per mantenere il più possibile l'imparzialità dei test, il tipo di dati inviati, che in questo contesto indica le proprietà dell'oggetto che modella la misura, è stato mantenuto inalterato.

Ciò su cui si è andati ad agire è stata la codifica degli stessi, e nello specifico sono stati testati tre formati:

- **Nessuna codifica:** l'oggetto in JavaScript viene inviato direttamente senza alcuna modifica.
  La struttura del dato deve essere definita rigidamente all'interno dello smart contract sotto forma di *struct*, e ciò richiede accorgimenti particolari nella definizione del tipo di variabili in modo da ottimizzare lo spazio occupato e adeguarsi ai limiti imposti da Solidity. Infatti, non è supportato l'uso dei numeri decimali, ovvero il tipo *float*, ed è quindi necessario salvarli come interi moltiplicandoli per un fattore pari alla precisione del numero e eliminando la parte decimale, per poi effettuare la divisione per lo stesso fattore alla lettura del dato fuori dalla Blockchain.
  È inoltre importante dimensionare adeguatamente le variabili, ad esempio anziché salvare le stringhe come *string*, l'uso dei *bytes32* risulta molto meno oneroso dato che le variabili dinamiche comportano un maggiore overhead: con solo due stringhe,

utilizzare la lunghezza fissa ha portato ad una riduzione del consumo di gas richiesto per una transazione di quasi 2500 unità, nonostante l'aumento della dimensione dei dati

- **Codifica testuale:** l'oggetto viene formattato come stringa, e in particolare è stata usata la codifica *JSON* (*JavaScript Object Notation*). La struttura chiave-valore degli oggetti in JavaScript è mantenuta intatta nella codifica, e la stringa risultante è in un formato facilmente leggibile dall'uomo.
  Sullo smart contract il dato è salvato semplicemente sotto forma di string, mentre lato JavaScript è presente di default la libreria JSON

- **Codifica binaria:** in questo caso l'oggetto è convertito in una sequenza di Bytes. A tale scopo è stata usata la codifica *CBOR* (*Concise Binary Object Representation*), ispirata dal modello JSON, che serializza i dati in uno *stream* di items estremamente conciso rispetto alle precedenti opzioni.
  Relativamente allo smart contract con codifica in JSON, è sufficiente modificare il dato da string a bytes, e per l'invio e la decodifica del dato è disponibile la libreria ufficiale cbor.js[48]

5.3 Smart contract

L'altro parametro considerato è rappresentato dalla metodologia con cui il dato viene memorizzato nella Blockchain.

In Solidity, sono due le tecniche per il salvataggio delle informazioni:

- **Nello storage:** il dato viene salvato in una variabile all'interno del contratto. È ciò che è stato precedentemente definito come lo stato della Blockchain.
  Essendo interessati allo storico dei dati, è necessario che la variabile sia un array dentro il quale le misure vengono appese in coda. Questo perché il codice scritto in Solidity, e in generale tutto il codice appartenente alla EVM, ha accesso allo stato del solo blocco nel quale viene chiamato, e perciò, benché sia possibile richiedere i valori salvati in un blocco passato, recuperare tutto lo storico richiederebbe l'invio di un numero di transazioni pari alla quantità di blocchi

- **Tramite il log degli eventi:** la EVM offre una funzionalità di *logging*, che viene sfruttata emettendo un evento all'interno del contratto. In questo modo gli argomenti dell'evento vengono salvati nel log delle transazioni, il quale è disponibile nella Blockchain o recuperabile fintanto che si ha accesso ai blocchi. Alla versione del compilatore EVM attuale, i log passati vengono mantenuti in eterno anche in caso di pruning dei nodi, ma ciò potrebbe cambiare nelle versioni successive.
  Le informazioni dei log non sono accessibili all'interno del contratto, ma è possibile ricavarle lato JavaScript, così come sottoscrivere un *listener* all'evento in modo da ricevere i dati nel momento in cui vengono emessi dalla rete.
  È possibile indicizzare fino a tre parametri in un evento, fornendo l'opportunità di filtrare quali eventi ascoltare o quali log recuperare[49]

5.4 Test e risultati

I test si sono svolti su ogni configurazione limitando il numero massimo di transazioni al secondo inviabili dal sensore. Nel caso in cui il sistema avesse retto il carico per un tempo sufficientemente lungo, il tetto della velocità d'invio veniva aumentato.

A causa delle limitazioni hardware, a velocità più elevate l'apparato non è riuscito a mantenere la velocità di invio limite, ma ciò non ha inficiato sui risultati dato che si è raggiunta la saturazione della Blockchain.

Per tutti i grafici riportanti le transazioni al secondo, l'asse delle ascisse rappresenta i minuti trascorsi, dato che la velocità è stata calcolata considerando la media su periodi di un minuto.

Le prime configurazioni testate hanno in comune la metodologia di salvataggio del dato, ovvero sfruttando i log degli eventi.

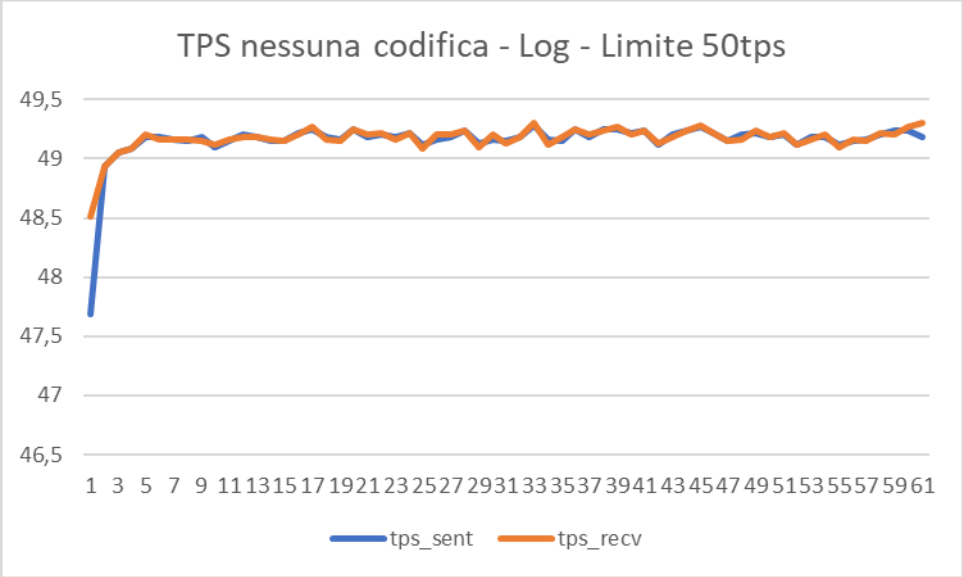

Figura 24 - TPS - Nessuna codifica - Log - Limite 50tps

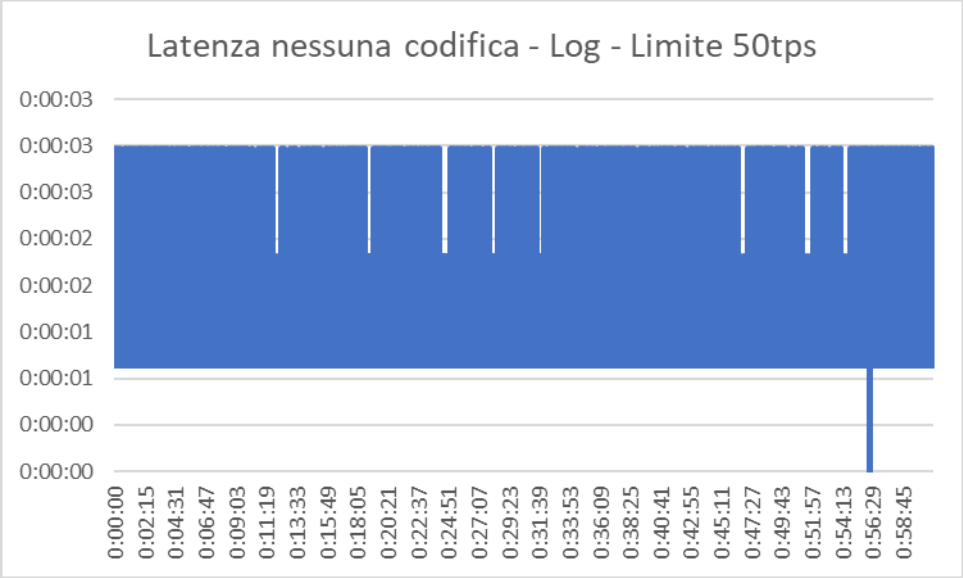

Figura 25 - Latenza - Nessuna codifica - Log - Limite 50tps

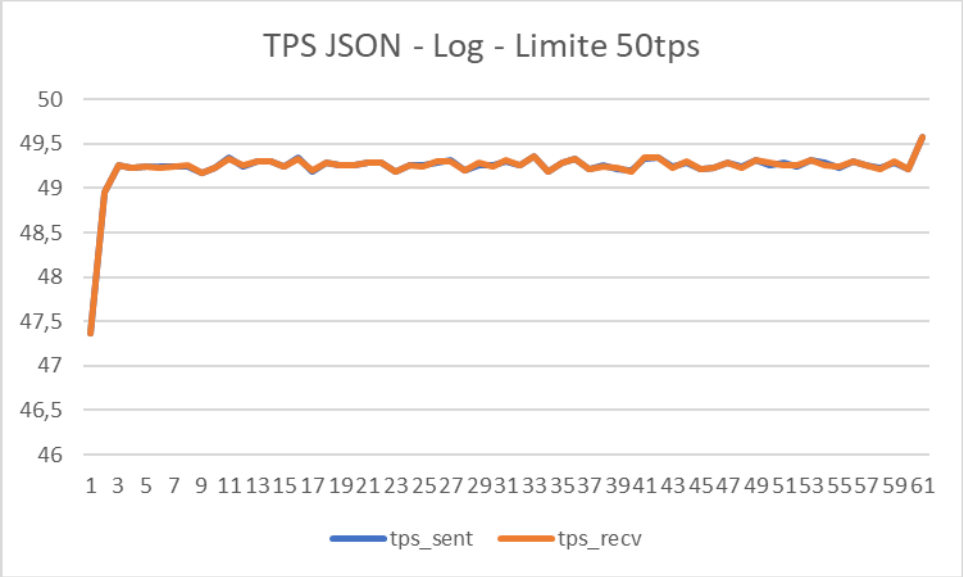

Figura 26 - TPS - JSON - Log - Limite 50tps

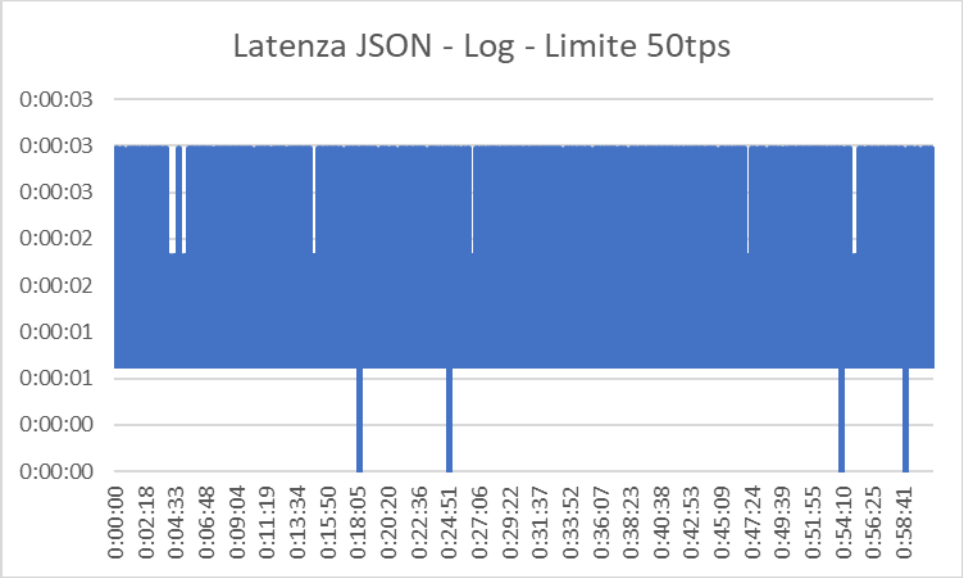

Figura 27 - Latenza - JSON - Log - Limite 50tps

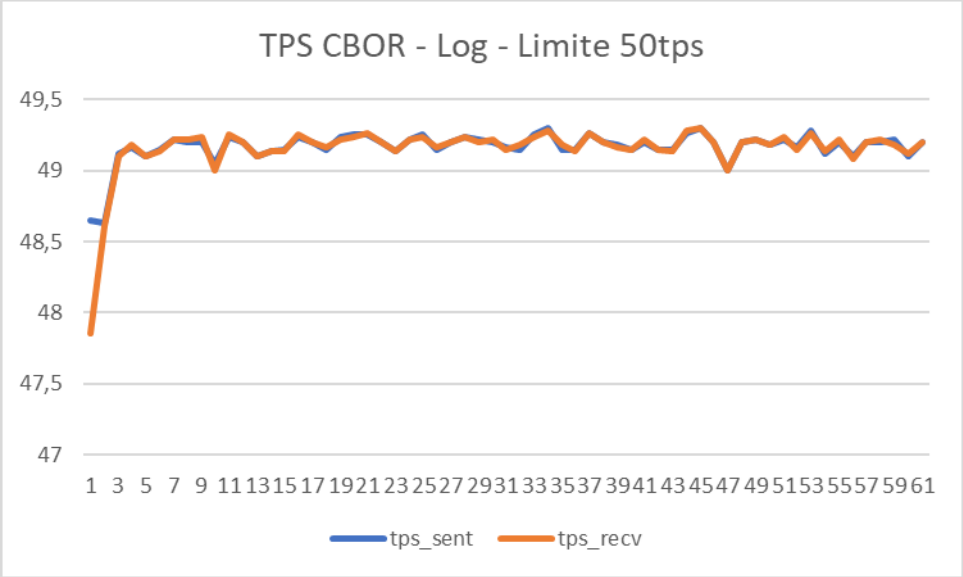

Figura 28 - TPS - CBOR - Log - Limite 50tps

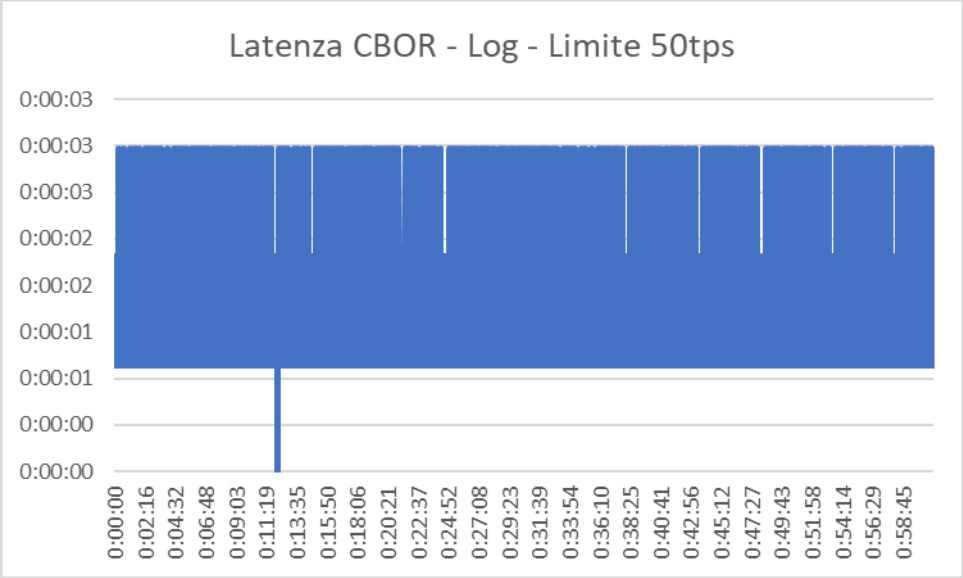

Figura 29 - Latenza - CBOR - Log - Limite 50tps

A velocità inferiori alle 50 transazioni al secondo, tutte le configurazioni si sono dimostrate capaci di soddisfare il carico riuscendo a processare i dati alla stessa velocità con cui venivano ricevuti, come confermato dalla bassa latenza.

È stato quindi possibile procedere ad aumentare il limite in modo da trovare la frequenza massima mantenibile dalla rete.

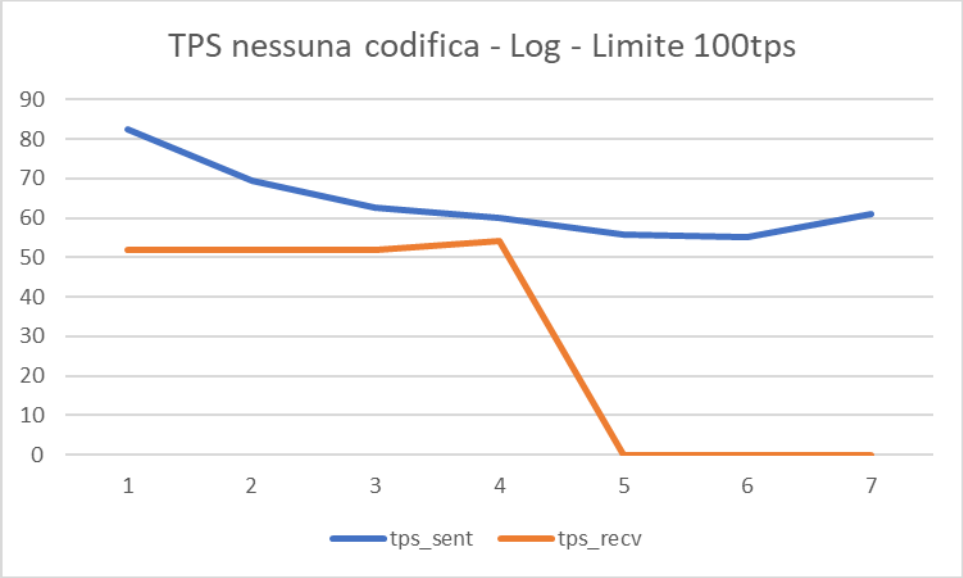

Figura 30 - TPS - Nessuna codifica - Log - Limite 100tps

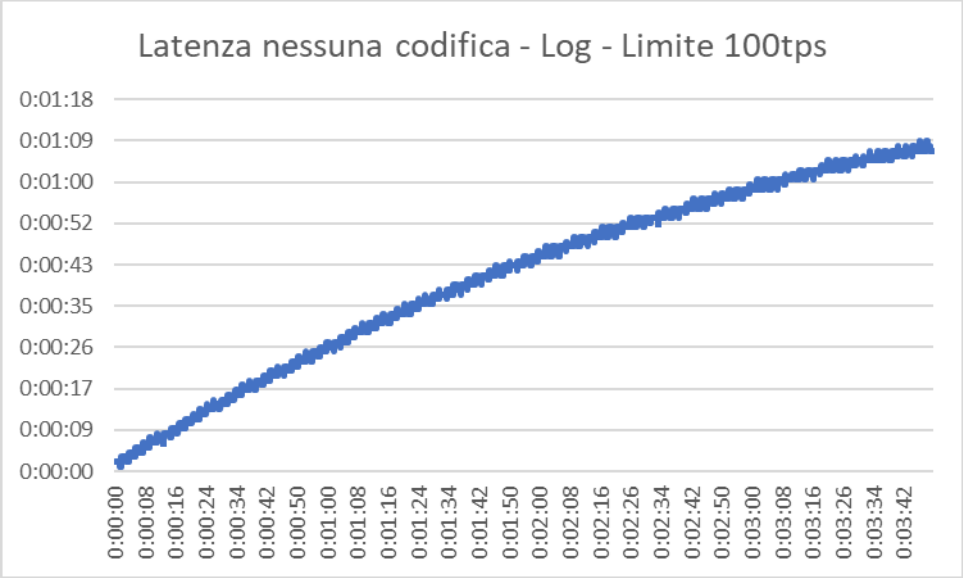

Figura 31 - Latenza - Nessuna codifica - Log - Limite 100tps

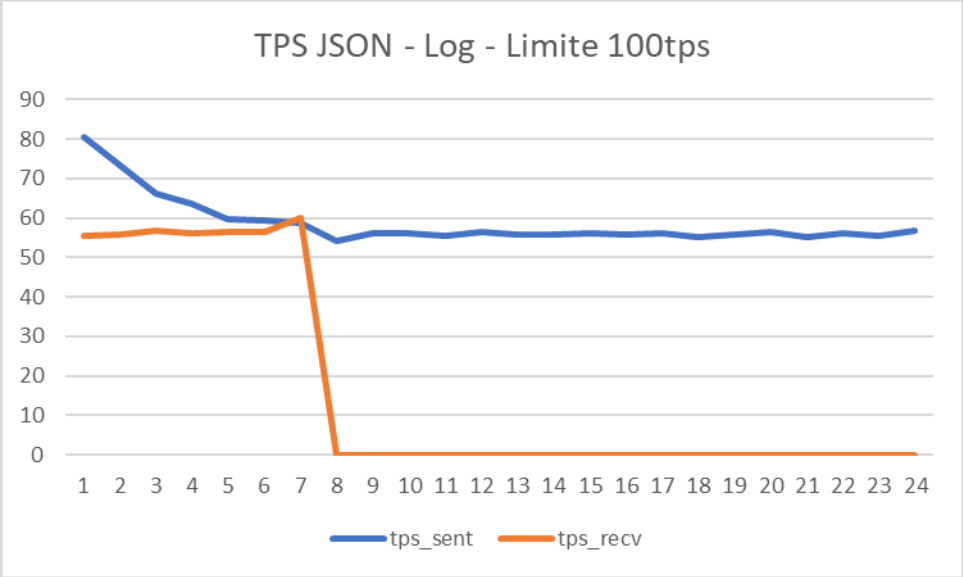

Figura 32 - TPS - JSON - Log - Limite 100tps

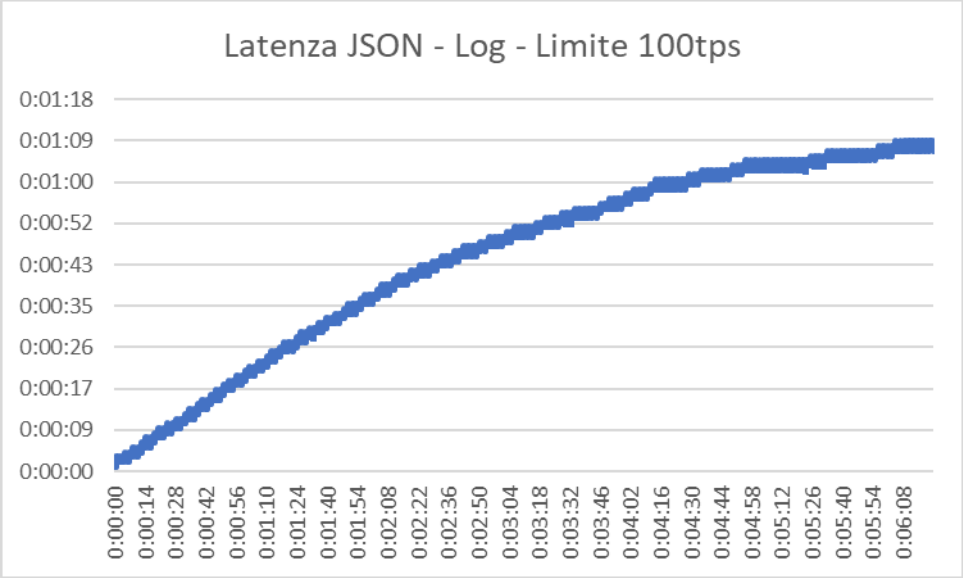

Figura 33 - Latenza - JSON - Log - Limite 100tps

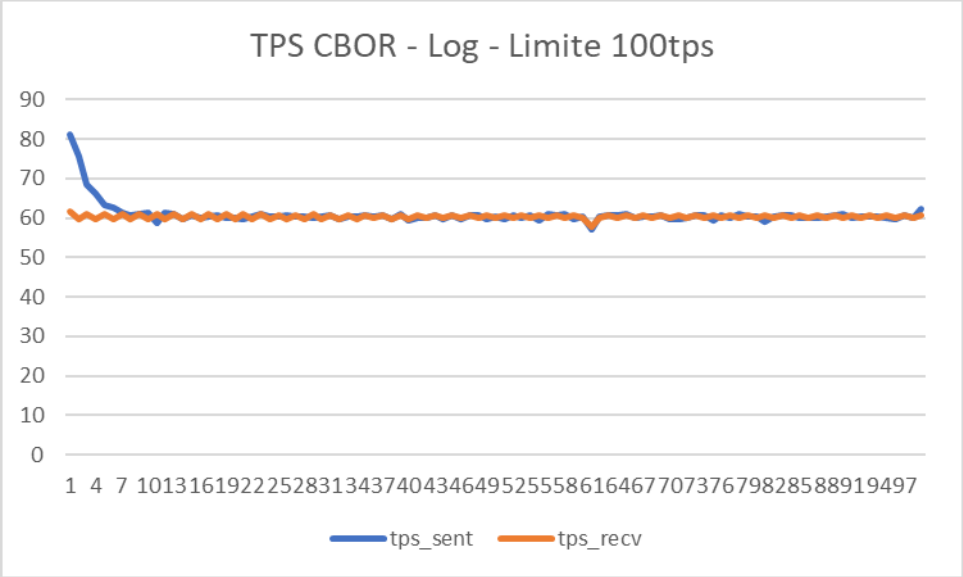

Figura 34 - TPS - CBOR - Log - Limite 100tps

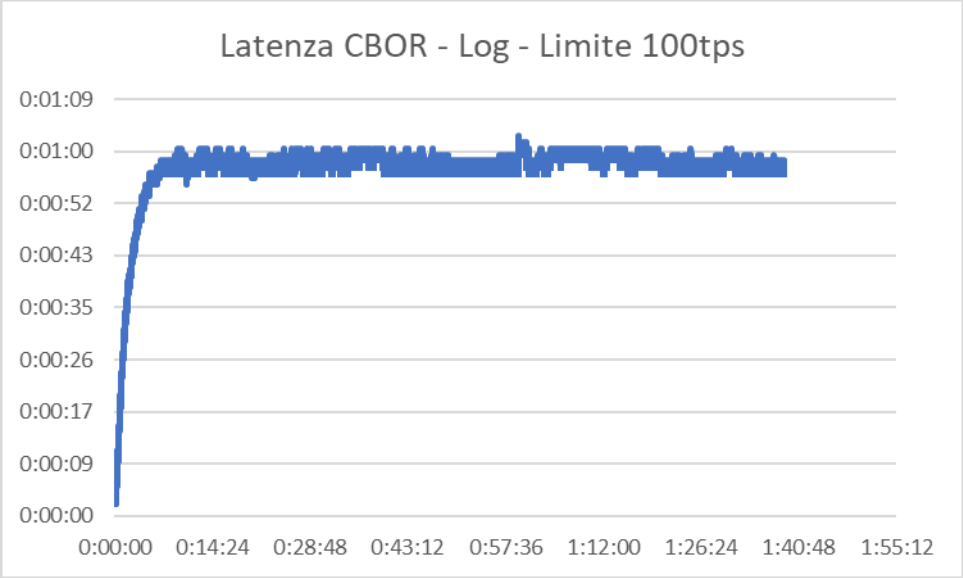

Figura 35 - Latenza - CBOR - Log - Limite 100tps

È possibile notare come la configurazione senza codifica e quella in formato JSON non siano riusciti a gestire il picco iniziale, portando a una conclusione prematura del test.

L'azzeramento della velocità di ricezione delle transazioni è dovuto alle particolari interazioni tra il codice di test e l'ambiente Besu.

Per l'invio delle transazioni è necessario specificare un nonce che indichi il numero di transazioni eseguite da un indirizzo. Attendendo il termine di una transazione, il nonce corretto si può ottenere su Besu tramite la funzione eth_getTransactionCount, che ritorna il nonce recuperato dall'ultimo blocco.

Nel nostro caso, nel quale si cerca di inviare il maggior numero possibile di transazioni dallo stesso terminale senza attendere la conferma della corretta ricezione del dato, il valore ritornato sarebbe però errato, dato che le ultime transazioni inviate non sarebbero ancora parte della catena. Il nonce viene perciò ottenuto all'inizio dell'esecuzione e aumentato a ogni ciclo, pratica che ha anche il pregio di limitare la latenza e il carico sul nodo.

I nodi mantengono però una coda di dimensione limitata nella quale le transazioni attendono di essere processate. Se questa viene riempita, le transazioni meno recenti vengono scartate, e non essendoci più la corretta sequenzialità dei nonce tutte le transazioni successive sono considerate come invalide.

Nonostante ciò, i risultati dei test rimangono significativi, avendo dimostrato che entrambe non riescono a smaltire le transazioni a una velocità superiore ai 60 TPS, ma come la codifica JSON abbia prestazioni lievemente superiori che le permettono di resistere per qualche minuto in più rispetto alla controparte.

Discorso diverso è rappresentato dalla codifica CBOR: il primo picco ha portato alla quasi saturazione della coda di transazioni, ma essendo la velocità di invio diminuita fino a circa 60 transazioni al secondo, la Blockchain è riuscita a smaltire i dati con la stessa frequenza di arrivo. Possiamo quindi nuovamente confermare come il limite del sistema sia di circa 60 TPS. Il grafico successivo mostra come aumentando ulteriormente il tetto massimo, la velocità massima di ricezione rimanga costante, provocando una situazione simile alle precedenti.

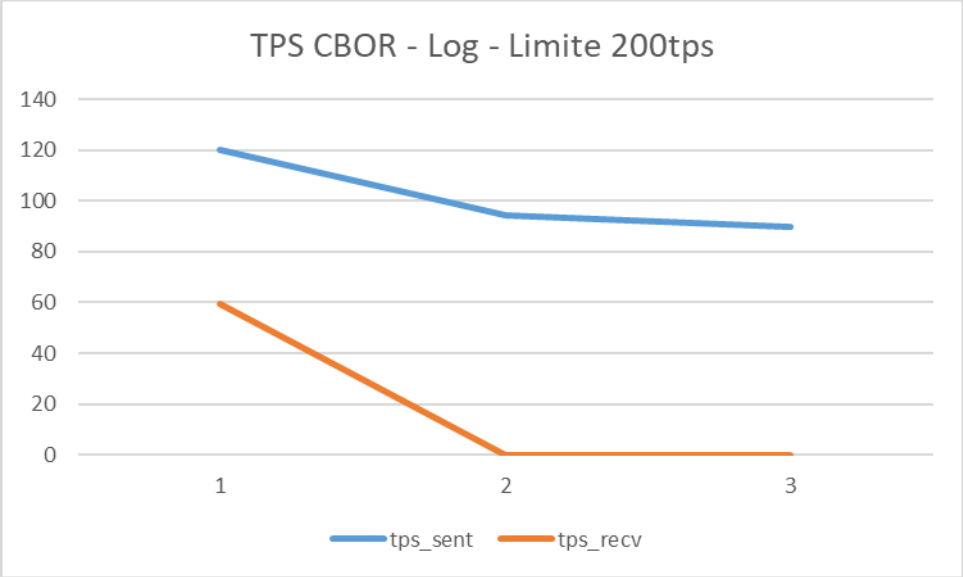

Figura 36 - TPS - CBOR - Log - Limite 200tps

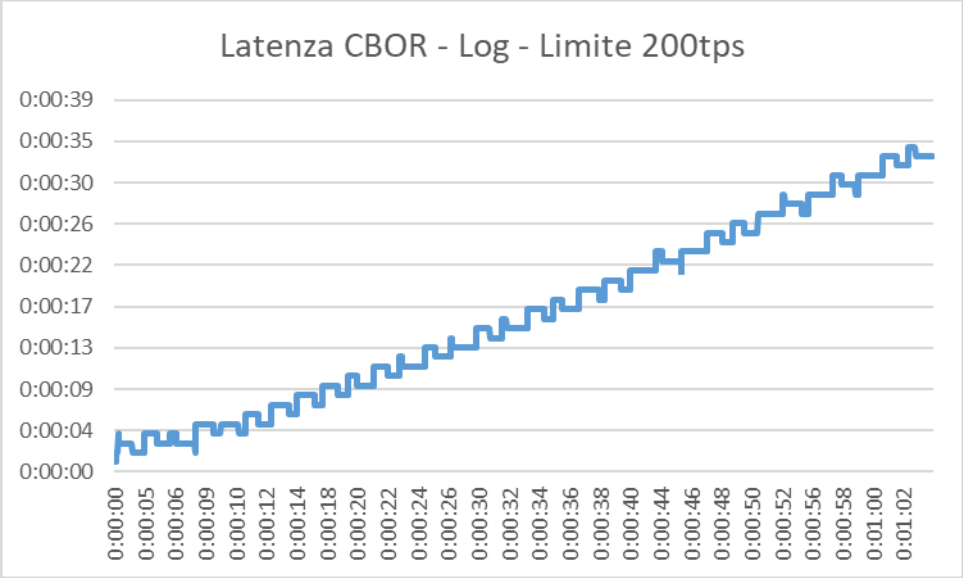

Figura 37 - Latenza - CBOR - Log - Limite 200tps

I grafici sottostanti mostrano invece le configurazioni ottenute memorizzando i dati nello stato della Blockchain.

L'assenza dell'opzione senza codifica è dovuta ai vincoli imposti da Solidity sulla profondità dello stack nella chiamata delle funzioni, che limita il numero di argomenti accettabili dalle stesse. Non potendo modificare la struttura dati poiché i risultati non sarebbero correttamente comparabili, è stato deciso di ignorare questa impostazione, ma ci si possono aspettare risultati simili a quelli riportati con le codifiche JSON e CBOR.

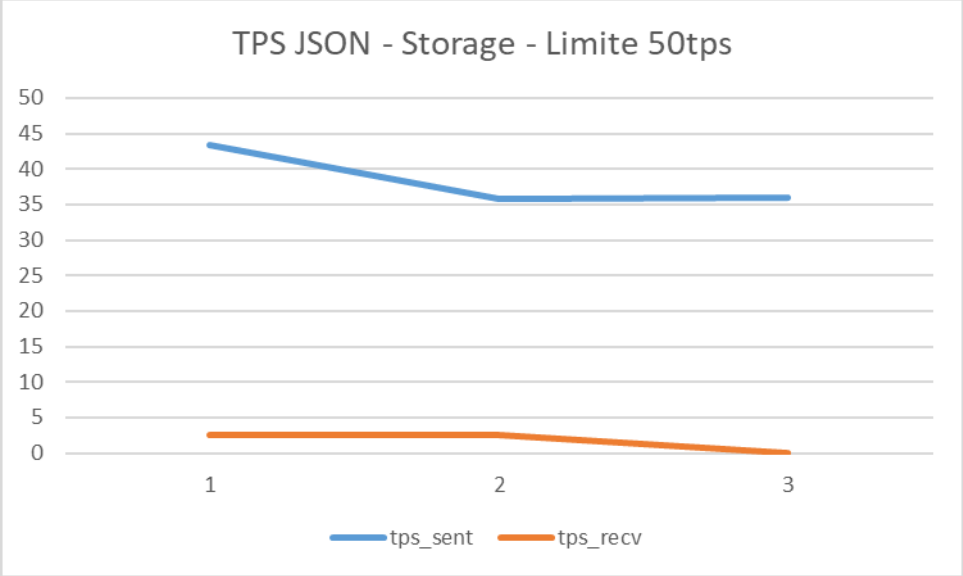

Figura 38 - TPS - JSON - Storage - Limite 50tps

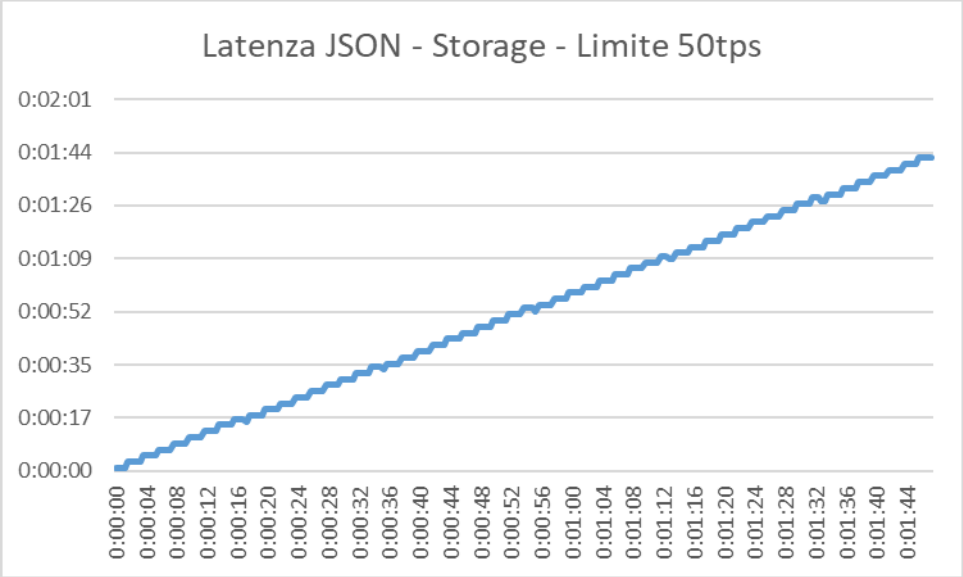

Figura 39 - Latenza - JSON - Storage - Limite 50tps

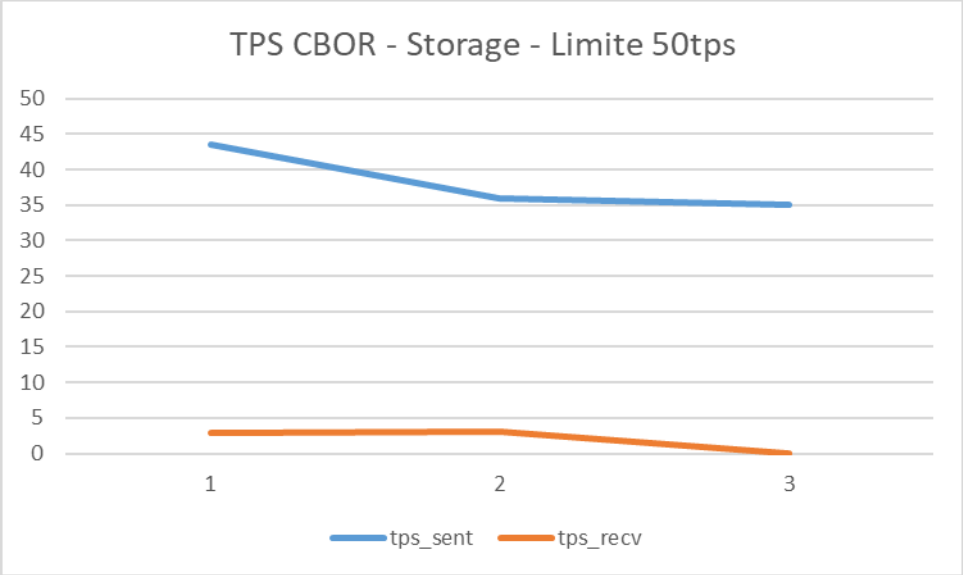

Figura 40 - TPS - CBOR - Storage - Limite 50tps

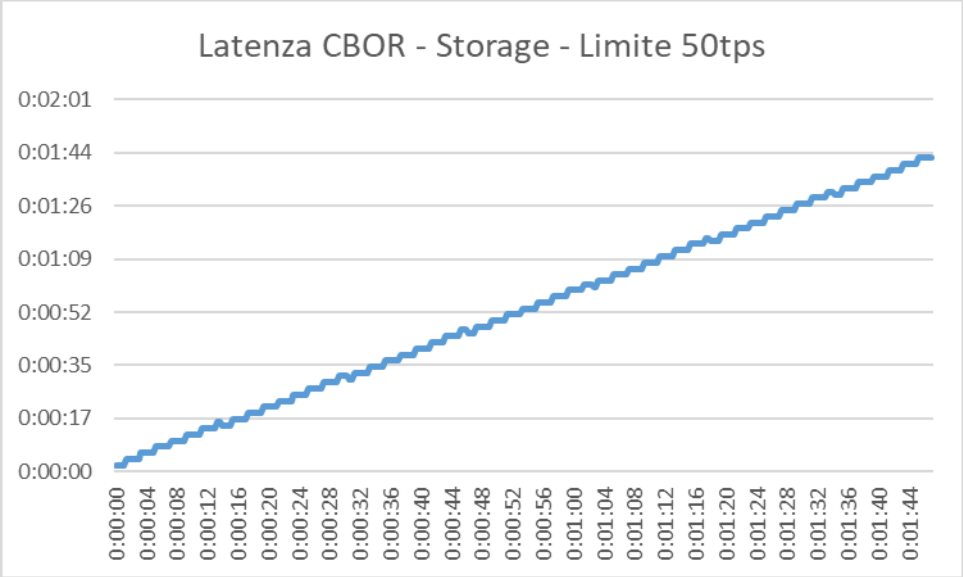

Figura 41 - Latenza - CBOR - Storage - Limite 50tps

I risultati ottenuti sono nettamente inferiori rispetto alle controparti precedenti, non riuscendo a superare la barriera delle 5 transazioni al secondo.

Ciò è dovuto all'enorme costo che la modifica dello stato della Blockchain comporta, che si attesta in media sulle 20000 unità di gas per 32 byte contro le 8 per byte delle operazioni di log[50].

Il grafico successivo indica la dimensione della misura impiegata nei vari test, considerando la dimensione di un carattere come di un Byte anziché i due impiegati da JavaScript.

Come atteso, l'uso di una codifica efficiente diminuisce notevolmente le dimensioni del payload.

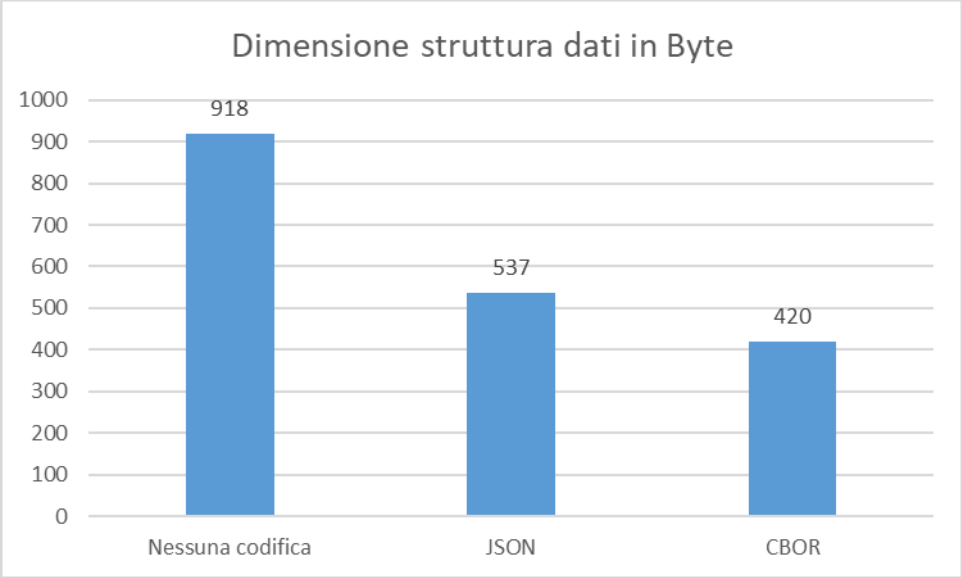

Figura 42 - Dimensione struttura dati

Il grafico sottostante mostra la dimensione del codice ABI della transazione codificato come stringa esadecimale. Tale stringa contiene l'hash della firma della funzione chiamata in aggiunta ai parametri passati al contratto, e rappresenta il pacchetto effettivo che viene trasmesso al nodo.

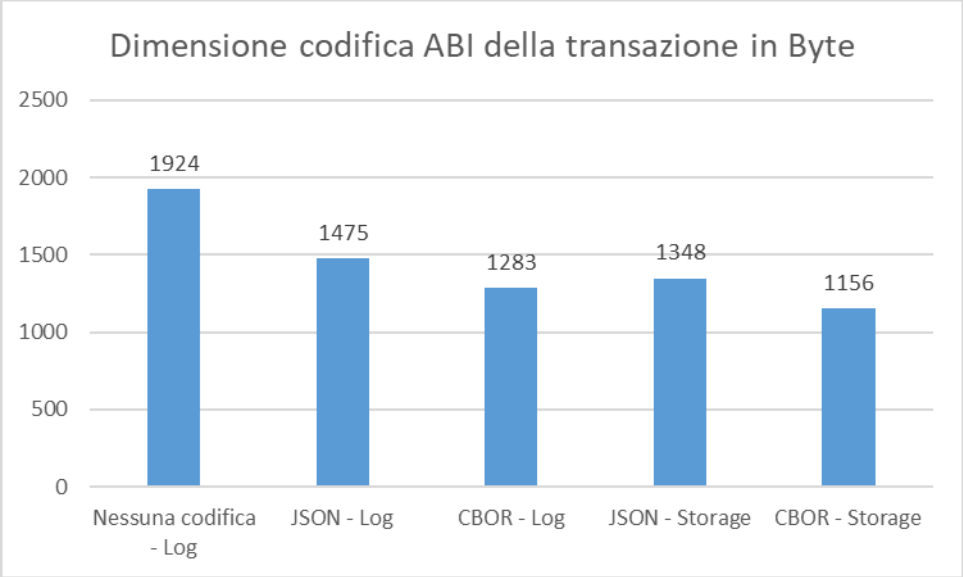

Figura 43 - Dimensione codifica ABI

L'ultimo grafico mostra la potenza computazionale media richiesta ai nodi per l'esecuzione della transazione. Si può notare come questo dato sia strettamente correlato ai risultati ottenuti negli stress-test, così come all'impatto della codifica e della scelta della modalità di salvataggio sul consumo di gas.

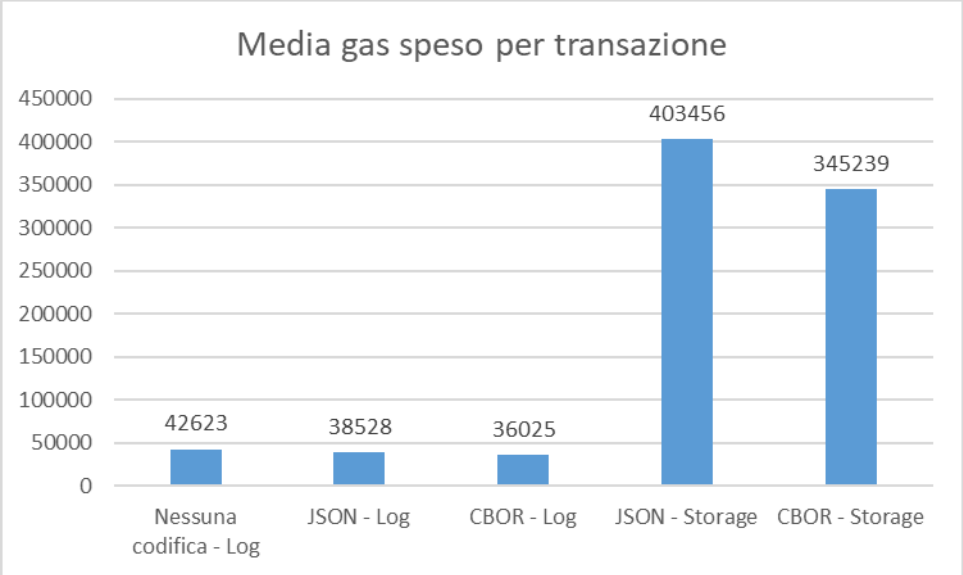

Figura 44 - Gas speso

5.5 Conclusioni

Per quanto limitati dal punto di vista della funzionalità, i test svolti hanno dimostrato la fattibilità sul lato tecnico dell'implementazione della soluzione Blockchain al caso studio proposto, fornendo prestazioni consone alle richieste fornite.

In caso di sviluppi futuri è opportuno dimensionare adeguatamente le macchine impiegate, specialmente in termini di CPU in modo da velocizzare i processi di validazione, ma anche valutando l'impiego di generose quantità di RAM, dato che è possibile aumentare la dimensione della coda di transazioni in attesa in modo tale da ridurre il rischio di perdita delle transazioni.

Lato software invece è estremamente consigliato l'uso delle funzionalità di log per la memorizzazione dei dati storici, dato l'enorme impatto sulle prestazioni della soluzione mediante storage. Si rinuncerebbe in questo modo alla possibilità di gestire le transazioni passate dentro lo smart-contract, ma per le funzioni che ci si aspetta debba svolgere ciò non rappresenta un problema.

È da valutare più attentamente l'impiego di una codifica sui dati: le prestazioni vengono sì migliorate, almeno per dati di una certa dimensione, ma si perde l'opportunità di operare sulle misure all'interno del contratto, ad esempio per controlli on-chain sul rispetto dei vincoli della spedizione. Esistono librerie per la conversione dei dati su Solidity, ma l'uso rappresenterebbe una soluzione controproducente dato l'aumento nella complessità delle operazioni e una conseguente riduzione nelle prestazioni.

Bisogna quindi decidere se sfruttare appieno le capacità della piattaforma e convivere con la perdita di efficienza o utilizzarla unicamente come database distribuito sicuro e delegare l'elaborazione dei dati all'infuori di essa.

Infine, è di vitale importanza osservare con attenzione le novità future in arrivo nel mondo Blockchain, dato che eventuali evoluzioni dei framework industriali potrebbero risolvere alcune criticità riscontrate in questo studio, come, a titolo esemplificativo, il rilascio di un'implementazione ufficiale di un algoritmo IBFT efficiente per Hyperledger Fabric o più in generale miglioramenti nelle prestazioni dei meccanismi di consenso.

Bibliografia e sitografia

1. Satoshi Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System, 2008
2. Jamsheed Shorish. Blockchain State Machine Representation, 2017
3. Konstantinos Christidis, Michael Devetsikiotis. Blockchains and Smart Contracts for the Internet of Things, 2016
4. https://it.wikipedia.org/wiki/Funzione_crittografica_di_hash
5. https://www.blockchain.com/charts/n-transactions-per-block
6. https://it.wikipedia.org/wiki/Albero_di_Merkle
7. https://www.bitcoinpeople.it/cosa-e-la-blockchain/
8. https://it.wikipedia.org/wiki/Problema_dei_generali_bizantini
9. https://www.kaleido.io/blockchain-blog/consensus-algorithms-poa-ibft-or-raft
10. https://hyperledger-fabric.readthedocs.io/en/release-2.2/orderer/ordering_service.html
11. https://docs.chainstack.com/blockchains/quorum
12. Zibin Zheng et al. An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends, 2017
13. Parth Thakkar et al. Performance Benchmarking and Optimizing Hyperledger Fabric Blockchain Platform, 2018
14. https://searchcio.techtarget.com/feature/What-are-the-4-different-types-of-blockchain-technology
15. https://quadrans.io/content/files/quadrans-white-paper-rev01.pdf
16. V. Gatteschi, F. Lamberti et al. Blockchain and smart contracts for insurance: Is the technology mature enough?, 2018
17. V. Gatteschi, F. Lamberti et al. To blockchain or not to blockchain: That is the question, 2018
18. Douglas M. Lambert. Supply Chain Management: Processes, Partnerships, Performance, 2008
19. K. McCormack, A. Lockamy. The Development of a Supply Chain Management Process Maturity Model Using the Concepts of Business Process Orientation, 2004
20. P. Brody. How blockchain is revolutionizing supply chain management, 2017
21. Hau L, Lee V, Padmanabhan Seungjin Whang. The Bullwhip Effect in Supply Chains, 1997
22. https://know.cerved.com/wp-content/uploads/2021/12/oss_pag_3q_2021.pdf
23. Rupa Dash et al. Application of artificial intelligence in automation of supply chain management, 2019
24. Yao W. Analysis on the Application of the Artificial Intelligence Neural Network on the New Energy Micro Grid, 2017

25. https://www.blockchain.com/charts/transactions-per-second
26. https://ethereum.org/en/eth2/
27. https://www.hyperledger.org/blog/2020/01/30/welcome-hyperledger-fabric-2-0-enterprise-dlt-for-production
28. Christian Gorenflo, Stephen Lee, et al. FastFabric: Scaling Hyperledger Fabric to 20,000 Transactions per Second, 2019
29. https://www.blockchain.com/charts/avg-confirmation-time
30. https://www.blockchain.com/charts/blocks-size
31. https://ethereum.org/en/eth2/shard-chains/
32. Dr. Gavin Wood. Polkadot: Vision for a heterogeneous multi-chain framework, 2016
33. https://v1.cosmos.network/resources/whitepaper
34. https://www.hyperledger.org/about
35. https://hyperledger-fabric.readthedocs.io/en/release-2.2/blockchain.html
36. https://www.hyperledger.org/blog/2019/08/29/announcing-hyperledger-besu
37. https://sawtooth.hyperledger.org/docs/core/releases/1.0/introduction.html
38. https://docs.goquorum.consensys.net/en/stable/
39. https://besu.hyperledger.org/en/stable/
40. https://docs.orion.consensys.net/en/latest/Tutorials/Migrating-from-Orion-to-Tessera/
41. https://sawtooth.hyperledger.org/faq/consensus/#what-consensus-algorithms-does-sawtooth-support
42. Michael Schwarz, Samuel Weiser, Daniel Gruß. Practical enclave malware with Intel SGX, 2019
43. https://besu.hyperledger.org/en/stable/Concepts/Consensus-Protocols/Overview-Consensus/
44. https://consensys.net/docs/goquorum/en/latest/concepts/consensus/overview/
45. https://hyperledger-fabric.readthedocs.io/en/release-2.2/whatis.html#pluggable-consensus
46. https://hyperledger-fabric.readthedocs.io/en/release-2.2/private-data/private-data.html
47. https://chainstack.com/resources/#enterprise-blockchainprotocols-evolution-index-2020
48. https://cbor.io/
49. https://docs.soliditylang.org/en/v0.8.11/contracts.html#events
50. https://media.consensys.net/technical-introduction-to-events-and-logs-in-ethereum-a074d65dd61e