Università degli Studi di Cagliari

Facoltà di Scienze

Corso di Laurea Magistrale in Informatica

A.A. 2021/22

Elaborato del Corso di Progetto e Sviluppo di Applicazioni Blockchain

Simulatore di sistemi di tracciamento di processi produttivi in campo agroindustriale - progetto ABATA

Docente del Corso

Prof. MARCHESI Michele

Studente

PISEDDU Enrico

(matr. 60/73/65222)

INDICE

1 Obiettivo dell'elaborato

L'obiettivo del presente elaborato è quello di descrivere un simulatore al fine di simulare la gestione di una catena di produzione (*supply chain*) e certificazione di prodotti agroalimentari attraverso la blockchain, eseguire diverse simulazioni al variare di alcuni parametri e presentare alcuni output di interesse del simulatore stesso al fine di analizzarli, confrontarli fra di loro e studiare le prestazioni dell'intero sistema.

Il simulatore è totalmente modellato attraverso tecniche ad oggetti, in particolare utilizzando il linguaggio Python. Sono modellati, inoltre, tutti i componenti del sistema di certificazione quali sistema server, blockchain, app, DBMS e DMS al fine di simulare le transazioni degli attori che coinvolgono ogni singolo componente.

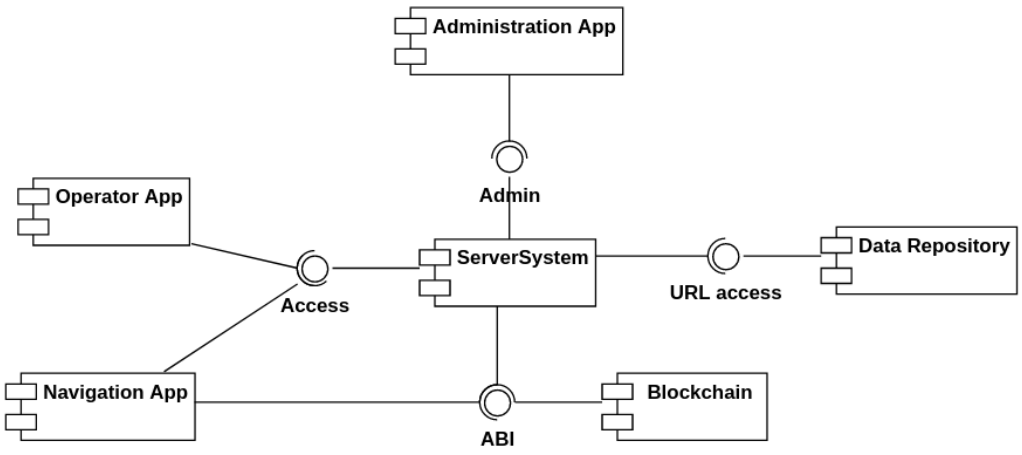
La configurazione dell'intero sistema da simulare è descritta da diversi file .csv, e nella stessa estensione sono poi prodotti gli output di interesse.

2 Il sistema simulato

Il sistema simulato è quello standard definito per il sistema ABATA, ed è mostrato in figura 1. Rispetto a tale figura non è simulato il componente di amministrazione il cui uso avviene in fase di configurazione ed in seguito in modo sporadico. Inoltre, il componente in figura "Data Repository" viene suddiviso in due specifici componenti indipendenti: il DMS e il DBMS.

Il numero specifico degli attori, distinti in Operatori e Consumatori, viene definito in fase di configurazione del sistema.

***Figura 1**. Il sistema simulato*



In sintesi, i componenti base del sistema sono:

- **appSys**: componente "Server system" che coordina i flussi informativi tra le app e il "Data Repository". Esso non è esplicitamente rappresentato nel modello ad oggetti, ma il suo funzionamento è comunque incluso nei ritardi applicati alle transazioni.

- **App**: comprende i componenti “Operator App” e “Navigation App”, app utilizzate rispettivamente dagli operatori che registrano e dai consumatori che navigano.
- **dlt**: la blockchain o DLT
- **dltview**: la blockchain quando viene interrogata in modalità “view” o “read only”, con risposta data direttamente dal nodo di accesso
- **DBMS**: il sistema di gestione del database
- **DMS**: il document management system ovvero il sistema di gestione documentale in grado di registrare documenti complessi come file .pdf, immagini e/o filmati.

Tutto il sistema simulato, che non include dispositivi IoT, viene stimolato dalle operazioni (i.e. transazioni) degli attori, ovvero consumatori e operatori. Gli operatori sono in numero limitato (al più qualche centinaio) e creano soprattutto transazioni di scrittura di dati e documenti riguardo prodotti agroalimentari. Al contrario, i consumatori possono essere in numero anche ingente (migliaia) ed inviano transazioni di lettura dalla blockchain o dal DMS in quanto desiderano verificare l’integrità dei dati dei prodotti agroalimentari.

Le **transazioni base** sono le operazioni base del simulatore, vengono lanciate dai diversi attori e coinvolgono ognuna un solo componente. Si veda la tabella seguente.

Transazione (nome)	Componente	Descrizione
Input	app	Input di dati da parte di un operatore o di un cliente tramite un'app, su smartphone, tablet o PC.
Read	app	Lettura di un dato o documento da parte di un operatore o di un cliente tramite un'app, su smartphone, tablet o PC.
DLT_read	dltview	Lettura di un dato dalla blockchain. Tipicamente, senza consumo di gas (nel caso di blockchain che richieda gas per “pagare” le transazioni) e con risposta diretta e veloce del nodo, senza transazione blockchain vera e propria.
DLT_write	dlt	Scrittura di un dato nella blockchain. Può richiedere gas se la blockchain ha bisogno di gas per “pagare” le transazioni.
DB_read	dbms	Lettura di un dato dal database.
DB_write	dbms	Scrittura di un dato nel database.
DMS_read	dms	Reperimento e lettura di un documento dal DMS.
DMS_write	dms	Scrittura di un documento entro il DMS.

Ciascuna di queste transazioni base è dotata di un tempo medio di esecuzione, deviazione standard.

Una precisa sequenza di transazioni base definisce una **transazione complessa** che coinvolge più componenti. Ad esempio, la lettura di un documento da parte di un consumatore comporterà l’esecuzione delle seguenti quattro transazioni base: l’interrogazione della blockchain in modalità “*read only*”, l’utilizzo dell’app in modalità read, la richiesta di download del documento al DMS e la

lettura del documento nell'app. Le letture semplici sull'app e sulla blockchain possono essere reiterate un numero variabile di volte in modo da simulare una "navigazione" della storia di un prodotto agroalimentare da parte di un consumatore.

Una transazione complessa è caratterizzata da:

name	il nome che la identifica, ad es: "*Document write*".
% operators	percentuale di transazioni di questo tipo iniziate da un operatore, rispetto al totale di transazioni iniziate dagli operatori.
% customers	percentuale di transazioni di questo tipo iniziate da un cliente, rispetto al totale di transazioni iniziate dai clienti.
base txs	sequenza delle transazioni base che compongono la transazione complessa, ciascuna con un tempo specificato tramite media e deviazione standard. Tale sequenza può essere replicata un numero di volte di cui è data media e deviazione standard.

3 Specifica di una simulazione

La simulazione inizia specificando gli attori coinvolti, i componenti del sistema e i diversi tipi di transazioni degli attori stessi. Tutte queste entità sono descritte attraverso dei file `.csv`. Ogni file contiene una o più righe di intestazione.

3.1 Attori

Gli attori, distinti in operatori e consumatori, sono descritti nel file "`configuration/actors.csv`". Il file ha cinque colonne: type, nr. Of actors, nr TXs/day, std. dev. TXs/day, schedule.

Ad esempio, il file

```
Actors data: type, nr. of actors, nr. TXs/day, std.dev. TXs/day, schedule
operators,100,50,30,OP1
customers,10000,3,10,CU1
```

definisce un totale di 100 operatori e 10000 consumatori. Ciascun consumatore esegue in media 3 transazioni complesse al giorno con una deviazione standard di 10 e ha *schedule* "CU1". Lo stesso vale per l'operatore.

Lo *schedule* per gli attori definisce, per ogni ora della giornata, la probabilità che una transazione sia iniziata in quell'ora. A titolo d'esempio si veda lo *schedule* nella seguente tabella:

Attore \ Ora	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**
Operatore	10%	13%	13%	13%	7%	4%	12%	12%	12%	4%							
Consumatore		3%	6%	8%	8%	12%	6%	6%	8%	8%	8%	5%	10%	7%	5%		

3.2 Componenti

I componenti sono descritti nel file "`configuration/components.csv`". Tale file ha cinque colonne:

name	nome del componente
avg. delay	valore medio del ritardo (in ms) introdotto dal componente su tutte le transazioni base
s. d. delay	deviazione standard del ritardo precedente
max txs	massimo numero di transazioni gestibili contemporaneamente. Oltre il 70% di tale numero il ritardo aumenta sino a raddoppiare se il numero massimo delle transazioni è raggiunto
wait when full	tempo di attesa (in ms) prima di ritentare la transazione se il numero di transazioni supera il massimo

Tale esempio definisce un sistema-tipo:

```
name, avg. delay (ms), st.dev. delay (ms), max txs, wait when full (ms)
app,3000,1000,10000000,500
appSys,200,80,300,500
dlt,2700,900,30,1500
dltview,700,300,800,500
dbms,300,150,300,200
dms,500,400,30,1000
```

3.3 Transazioni

Le transazioni base e quelle complesse sono definite nel file "`configuration/transactions.csv`". Tale file contiene la definizione di una transazione complessa seguita da tutte le sue transazioni base. Ha pertanto due righe di intestazione: la prima riga si riferisce alle caratteristiche della transazione complessa, la seconda invece alle caratteristiche delle transazioni base che la compongono:

```
TX data: name, nr. txs,% operators, % customers, avg blocks, s.d. blocks
For each txs: type, component, avg time (ms), std. dev. (ms)
Simple write,5,50,0,1,0
      DLT_read,dlt,400,200
      DB_read,dbms,300,100
      Input,app,5000,3000
      DB_write,dbms,600,300
      DLT_write,dlt,800,300
Document write,6,50,0,1,0
      DLT_read,dltview,400,200
      DB_read,dbms,300,100
      Input,app,30000,20000
      Doc_upload,dms,1500,1000
      DB_write,dbms,600,300
      DLT_write,dlt,800,300
Simple read,2,0,50,4,2
      DLT_read,dltview,300,100
      Read,app,5000,4000
```

```
Document read,4,0,50,3,1
      DLT_read,dltview,300,100
      Read,app,3000,2000
      Doc_download,dms,1000,800
      Read,app,30000,20000
```

Le transazioni complesse sono: *Simple Write*, *Document Write*, *Simple Read* e *Document Read*. Le loro caratteristiche sono le seguenti:

name	Identificatore della transazione complessa
nr. of txs	Numero intero di blocchi di transazione base che la compongono
% operators	Percentuale di transazioni di questo tipo iniziate da un operatore, rispetto al totale delle transazioni iniziate dagli operatori
% customers	Percentuale di transazioni di questo tipo iniziate da un cliente, rispetto al totale delle transazioni iniziate dai clienti
avg blocks	Numero medio di blocchi di transazioni base che compongono, in sequenza, la transazione complessa
s.d. blocks	Deviazione standard del numero di blocchi di transazioni base: se è zero, il numero di blocchi è fisso ed è uguale ad "avg blocks"

Per ogni transazione complessa, sono definite le transazioni base che la compongono. Tali transazioni base sono caratterizzate dai seguenti attributi:

type	Identificatore della transazione base
component	Identificatore del componente a cui si riferisce
avg time	Valore medio della durata della transazione base, in ms, eseguita dal componente suddetto
st. dev. Time	Deviazione standard della durata precedente (in ms)

Si noti che le prime due transazioni complesse (*Simple Write* e *Document Write*) sono sempre eseguite dagli operatori in quanto hanno, ciascuna, una % customers uguale a 50 e sono composte da un solo blocco di transazioni base poiche "avg block = 1" e "s.d. blocks = 0".

Le ultime due transazioni complesse (*Simple Read* e *Document Read*) sono invece eseguite sempre dai consumatori ma possono essere composte da più blocchi di transazioni base in quanto "avg block > 1" e "s.d. blocks > 1": questa caratteristica è dettata dal fatto che ciascun consumatore può accedere alla "storia" di uno o più prodotti agroalimentari più volte navigando sulla propria applicazione.

4 La simulazione

Una simulazione riguarda sempre un giorno solare, in quanto il carico di un sistema di tracciamento e certificazione agroalimentare avviene durante le ore lavorative e poi la sera da parte dei consumatori. Di notte è ragionevole supporre che il carico delle transazioni sia quasi nullo (i.e. trascurabile).

La simulazione parte leggendo i dati di configurazione presenti nei file `.csv` all'interno della cartella "`/configuration`". Vengono quindi creati in memoria gli oggetti principali: attori, componenti e transazioni per ogni attore.

In particolare, per ogni attore viene generato il rispettivo numero di transazioni utilizzando un generatore di numeri casuali: nel caso dei consumatori, se il numero di transazioni generato è zero, si assume sempre una transazione (questo accade spesso poiché la media delle transazioni per consumatore è un numero poco maggiore di 1). In seguito, ad ogni transazione è associato un *timestamp* di inizio, determinato con probabilità pari allo *schedule* dell'attore stesso.

Una volta generate le transazioni, la simulazione procede con un approccio a coda di eventi, i quali possono essere di tre tipi:

- **StartTx**: inizia l'esecuzione di una transazione base su un componente. Se il componente ha raggiunto il limite massimo di transazioni gestibili contemporaneamente, viene creato un evento ResubmitTx e inserito nella coda degli eventi ad un tempo uguale a quello corrente più un ritardo dipendente dal componente. Se, invece, il componente può accettare transazioni (i.e. non ha raggiunto il proprio limite massimo) viene creato un evento EndTx che viene inserito nella coda degli eventi ad un tempo uguale a quello corrente più un tempo che dipende dalla transazione (durata generata da media e deviazione standard) e dal componente.

- **EndTx**: l'esecuzione di una transazione su un componente termina. Se la transazione è seguita da un'altra transazione, allo stesso tempo di simulazione è generato un evento StartTx per quest'ultima transazione. Se, invece, la transazione base che deve terminare è l'ultima della serie della transazione complessa, è semplicemente marcata la fine di quest'ultima al tempo opportuno.

- **ResubmitTx**: si cerca di riprendere una transazione sospesa a causa di un componente pieno. Se il componente è ancora pieno, si genera un ulteriore evento di ResubmitTx con ritardo opportuno; se, invece, il componente accetta transazioni, si genera l'evento EndTx.

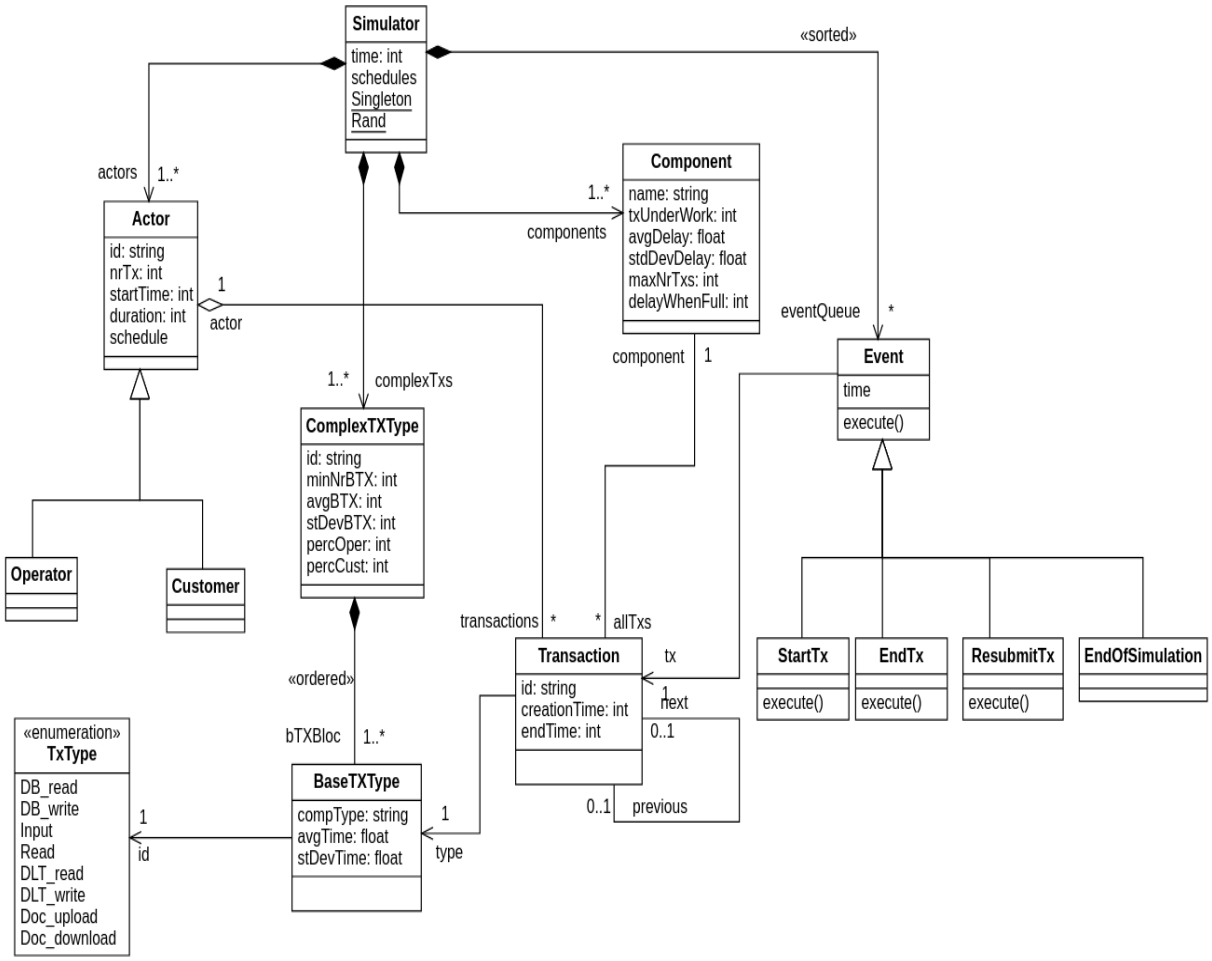
Il simulatore modellato estrae tali eventi dalla coda in ordine di tempo e simula l'esecuzione delle transazioni base che formano quelle complesse, una dopo l'altra.

Tutti gli oggetti come attori, transazioni, componenti ed eventi sono registrati durante la simulazione in modo da poter generare gli opportuni output a fine simulazione. È possibile, prima di far partire la simulazione, modificare i file .csv di configurazione (es. variando il numero degli attori, le tempistiche relative alle transazioni, ritardi dei componenti etc..) al fine di confrontare gli output.

4.1 Il modello a oggetti

In figura 2 è mostrato il modello a oggetti del simulatore

Figura 2. Il modello a oggetti, descritto come diagramma delle classi UML



4.2 La generazione di numeri casuali

La simulazione richiede la simulazione di numeri casuali di cui, in genere, si conosce la media e la deviazione standard al fine di calcolare:

- Il numero di transazioni per ciascun attore
- Il tempo di inizio di ogni transazione
- Il numero di blocchi di transazioni semplici che compongono una transazione complessa
- Il ritardo introdotto da un componente su una transazione
- Il tempo di esecuzione di una transazione base

Per generare un numero casuale distribuito secondo una distribuzione log-normale di media x e deviazione standard σ, a partire da una funzione $G(x_g, \sigma_g)$ che genera un numero casuale gaussiano, di media x_g e deviazione standard σ_g, si procede nel seguente modo:

Siano: $x_g = ln\left(\frac{x^2}{\sqrt{x^2+\sigma^2}}\right)$ e $\sigma_g = \sqrt{\frac{ln(x^2+\sigma^2)}{x^2}}$, allora la variabile r distribuita secondo la log-normale voluta sarà $r = e^{G(x_g,\sigma_g)}$ arrotondata all'intero più vicino

5 Caratteristiche dei componenti principali

I componenti principali sono mostrati in figura 1 nel capitolo 2.

5.1 App System

Si tratta del sistema server che coordina il funzionamento di tutto il sistema, offrendo servizi di autenticazione degli operatori, collegamento coi DBMS e DMS che mantengono i dati permanenti off-chain, servizi di Web server verso le app degli utenti, servizi di amministrazione e ovviamente collegamenti con la blockchain. Può essere un insieme di servizi su Cloud. In generale, è un sistema che, se correttamente dimensionato, non introduce particolari ritardi o colli di bottiglia. È anche possibile non rappresentarlo esplicitamente, ma considerare i (piccoli) ritardi da esso introdotti nei ritardi degli altri sistemi coordinati, e/o delle transazioni. In questa analisi dei componenti principali seguiremo questo approccio, e non discuteremo ulteriormente dell'App System.

5.2 DBMS

Il o i DBMS del sistema memorizzano i dati di gestione dello stesso. Un DBMS correttamente dimensionato tipicamente non è soggetto a degrado significativo di prestazioni, o addirittura a "colli di bottiglia", nel caso di sistemi come quelli simulati. Per quanto riguarda i ritardi introdotti, lo studio di prove reperite in letteratura (si vedano da 1 a 4 in bibliografia), porta ai tempi di risposta medi mostrati nella tabella seguente. I valori sono tutti in ms:

Studio	Ritardo medio in lettura	Ritardo medio in scrittura	Note
[1] Bassil 2012	125 – 420	60	Query complesse: le più lunghe perché in ordinamento
[2] Ilic 2021	140 – 2000		Confronta SQL Server e Oracle in sola lettura. Oracle sembra 2-4 volte più lento.
[3] Goltsis 2022	30 (d.s 10)	500 (d.s. 80)	Caso studio complesso su stazioni 5G
[4] Ilic 2022	300 – 400		Solo lettura. DB con 300.000 record, query complesse.

Considerati questi dati reperiti in letteratura, in tutte le simulazioni si assumeranno i ritardi di 200 ms in lettura (con d.s. 50) e 100 ms in scrittura (d.s. 30)

5.3 Document Management System (DMS)

Il o i DMS del sistema memorizzano i documenti relativi alla supply chain. Si può ipotizzare che siano soprattutto file pdf di dimensione da 20 a 100 KB, più eventualmente qualche file grafico .jpg

di dimensioni analoghe. Anche nel caso dei DMS, se questi è correttamente dimensionato, non è soggetto a degrado significativo di prestazioni, o addirittura a "colli di bottiglia", nel caso di sistemi come quelli simulati. In letteratura, si trova molto poco relativamente a prove di prestazioni di DMS. Un documento di Unisys sul DMS Alfresco [5] riporta tempi di lettura di un documento dell'ordine di 350 ms, e di scrittura di un documento dell'ordine di 690 ms. Si tratta però di tempi che non includo le prestazioni della rete, e quindi poco significativi.

Una prova grossolana di scaricamento di file da Google Drive e da Dropbox ha dato i risultati riportati nella tabella seguente:

Sistema	**Media (ms)**	**Deviazione standard (ms)**
Read – Dropbox	3175	684
Read – Google Drive	2758	381
Write – Dropbox	2372	260
Write – Google Drive	3778	804

Considerati questi dati provenienti da una prova empirica, nelle simulazioni si assumeranno per il DMS un tempo medio *read* pari a 3000 ms con deviazione standard di 550 ms, e un tempo medio *write* pari a 3200 ms con deviazione standard di 700 ms.

5.4 App

Per App si intendono i sistemi che girano su terminali mobili degli operatori e dei consumatori (rispettivamente Operator App e Navigation App). Si tratta di sistemi che non sono soggetti a ritardi o a colli di bottiglia perché operati da un unico utente.

I tempi di esecuzione di una transazione su app dipendono solo dai tempi di risposta dell'utente umano. Si può ipotizzare che tali tempi siano molto variabili e dipendenti da molti fattori.

In generale, per la simulazione si utilizzeranno i tempi riportati nella seguente tabella:

Operazione	**Media (ms)**	**Dev. St (ms)**	**Note**
Read di dati	5000	4000	Lettura ed interpretazione di dati semplici
Read documento	20000	6000	Scaricamento e lettura di un documento
Write di dati	15000	5000	Scrittura di dati tramite un "form"
Write documenti	10000	6000	Upload di un documento pdf o simile

5.5 Blockchain

I tempi di risposta e la possibilità di saturazione del sistema sono molto variabili a seconda della tecnologia DLT simulata. Inoltre, nel caso di Ethereum e blockchain similari, le transazioni che leggono semplici dati senza alterare la blockchain sono gestite localmente dal nodo di accesso, mentre le transazioni di scrittura devono essere accettate dall'algoritmo di consenso, e poi il blocco relativo deve essere trasmesso a tutti i nodi. Ciò porta a differenze sostanziali sia nei tempi di risposta che nella possibilità di avere un numero massimo di transazioni gestibili in contemporanea.

Per questo motivo, si è deciso di dividere in due la descrizione di una blockchain, e cioè:

- Blockchain cui si accede in sola lettura (denominata, nel simulatore, DLTview)
- Blockchain cui si accede in scrittura (denominata, nel simulatore, DLT)

I tempi di risposta e il massimo numero di transazioni gestibili in contemporanea dai vari tipi di DLT sono mostrati nella tabella seguente. I tempi sono in ms; la d.s. è riportata tra parentesi dopo la media:

DLT	Consenso	Tempi risposta read	Tempi risposta write	Max nr. TX write	Note
Ethereum	PoW	800 (200)	70000 (30000)	300	Circa 12 tx/s e un blocco ogni 13". In media, si aspettano 5-6 blocchi per cui il tempo di attesa è circa 70". Si stimano max 300 tx in attesa (ottenuto con 4,3 tx/sec), che è un numero verosimilmente mai raggiungibile.
Ethereum 2.0	PoS	800 (200)	300000 (160000)	10000	Dati simili a Eth 1.0 perché i tempi di accettazione delle tx sono simili
Polkadot	PoS	500 (100)	360000 (200000)	10000	Idem
Etherna	PoA	700 (200)	2500 (1000)	250	Scrittura delle txs ogni 2,5". Accetta circa 100 txs/sec

6 Output del simulatore

I fattori che influenzano le prestazioni del sistema e gli output, sono, oltre ai tempi di risposta dei vari componenti, il dimensionamento del sistema (i.e. numero di attori) e la loro attività in termini di numero di transazioni giornaliere (con conseguente distribuzione temporale e tipologia di transazioni).

6.1 Assunzioni

Per ora si è considerato un sistema centralizzato con singolo DBMS, DMS e DLT. Per cercare di mettere sotto stress un tale sistema si è presupposto un numero elevato di operatori e clienti, cioè un sistema di tracciamento e certificazione adottato da un grande produttore per molti prodotti.

Gli operatori eseguono operazioni di tipo "write" (al fine di tracciare e certificare i prodotti) e qualche "read", mentre i clienti eseguono solo operazioni di tipo "read".

Ipotesi sugli attori:

- Azienda o complesso di grandi aziende agroalimentari, con produzioni massive (es. vino, olio, prodotti alimentari generici)
- Numero di operatori (certificatori) attivi fra 50 e 1000, con numero medio di 10 – 100 registrazioni al giorno ciascuno
- Numero di consumatori interessati elevato e dipendente dalla tipologia di prodotto (fino ad 1.000.000 di consumatori, con poche transazioni di lettura ciascuno)
- Le transazioni si addensano in determinate ore della giornata (nel caso degli operatori nell'orario di lavoro, per i clienti nella pausa pranzo o in tarda serata), si veda lo *schedule* del paragrafo 3.1

Sono state effettuate diverse simulazioni al fine di studiare di confrontare gli output del sistema facendo variare in maniera crescente il numero di attori.

6.2 Definizione degli output

Sono stati definiti i seguenti output di interesse.

Per ogni componente:

1. Dati un tempo iniziale e finale, ed un intervallo di tempo, si considera il numero di transazioni iniziate e terminate in ciascun intervallo
2. Dati un tempo iniziale e finale, ed un intervallo di tempo, rende gli intervalli nei quali il componente ha raggiunto il limite massimo di transazioni
3. Statistiche, per ogni transazione base, della durata (media, deviazione standard, caso migliore e peggiore)
4. Statistiche, per ogni transazione base iniziata in un dato intervallo, della durata (media, deviazione standard, caso migliore e peggiore)
5. Numero di transazioni base eseguite ogni sec (txs/sec)

Per ogni tipo di transazione complessa:

6. Stima della durata media utilizzando i valori medi ottenuti al punto 3

Note sugli output di interesse

Output	**Note**
4	Il caso migliore e il caso peggiore considerano, rispettivamente, la transazione base eseguita nel minor tempo e nel peggior tempo
5	Per il calcolo delle txs/sec, è stata utilizzata la seguente formula: $$N_{TPS}^{C} = \frac{N_{TX}^{C}}{T_C}$$ dove: - N_{TPS}^{C} definisce il numero di transazioni base per secondo eseguite del componente C - N_{TX}^{C} definisce il numero totale delle transazioni eseguite dal componente C - T_C definisce il tempo totale in cui il componente C è attivo per l'esecuzione delle transazioni

6.3 Generazione degli output

Una volta terminata una specifica simulazione, tutti gli oggetti (transazioni, attori e componenti) vengono memorizzati in apposite strutture dati in modo tale da essere manipolati al fine di generare gli output precedentemente definiti.

In particolare, il sistema genera gli output in una *directory* chiamata "./`report`" costituita nel seguente modo:

```
.
└── report/
    ├── components/
    │   ├── <nameOfComponent>_transactions_actors_<numberOfActors>.csv
    │   └── <nameOfComponent>_transactions_actors_<numberOfActors>.png
    ├── base_txs_duration_actors_<numberOfActors>.png
    ├── base_txs_duration_statistics_actors_<numberOfActors>.csv
    ├── complex_txs_duration_actors_<numberOfActors>.png
    ├── interval_base_txs_duration_actors_<numberOfActors>.png
    └── interval_base_txs_duration_statistics_actors_<numberOfActors>.csv
```

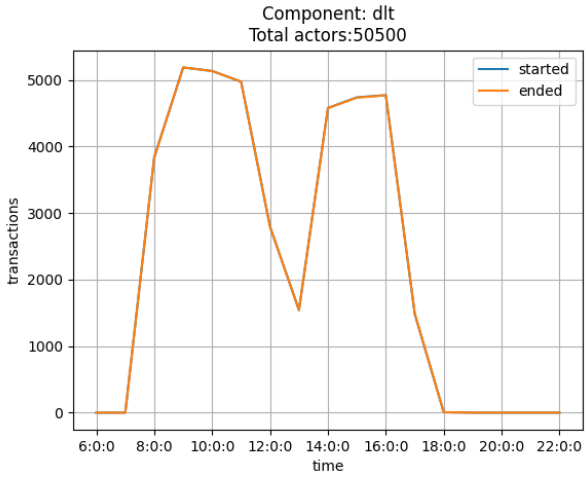
Al suo interno è presente un'ulteriore *directory* detta "`components`" la quale contiene, per ogni componente, due file: il primo file è in formato `.csv` e il secondo è una immagine `.png`. Il file `.csv` presenta tre colonne:

time	Ora di riferimento
started	Numero di transazioni base iniziate nell'ora di riferimento
ended	Numero di transazioni base terminate nell'ora di riferimento

Figura 3

	C1	C2	C3
1	time	started	ended
2	6:0:0	0	0
3	7:0:0	0	0
4	8:0:0	3837	3832
5	9:0:0	5188	5189
6	10:0:0	5135	5134
7	11:0:0	4974	4976
8	12:0:0	2796	2795
9	13:0:0	1541	1542
10	14:0:0	4579	4577
11	15:0:0	4739	4735
12	16:0:0	4772	4769
13	17:0:0	1481	1492
14	18:0:0	6	7
15	19:0:0	0	0
16	20:0:0	0	0
17	21:0:0	0	0
18	22:0:0	0	0

Figura 4



Ad esempio, il file "`report/components/dlt_transaction_actors_50500.csv`" (in figura 3) riporta, per ogni intervallo di tempo di un'ora dalle 6:00 alle 22:00 (intervallo configurabile), il numero di transazioni base iniziate e terminate nell'intervallo stesso dal componente "dlt" e lanciate da 50.500 attori. Il file rinominato allo stesso modo ma con estensione `.png` (figura 4) è il risultato del plot del file `.csv`: esso consente di visualizzare, in funzione del tempo, l'andamento del numero di transazioni iniziate e terminate in un intervallo di un'ora e di dedurre gli intervalli in cui è stato raggiunto il picco delle transazioni.

Sempre all'interno della *directory* "`report/`" sono presenti ulteriori file. Di ciascuno di essi se ne analizzerà il contenuto.

- Il file `base_txs_duration_statistics_actors_<numberOfActors>.csv` (in figura 5) riporta le statistiche ottenute per ogni tipo di transazione base eseguita nella simulazione. È composto da sei colonne:

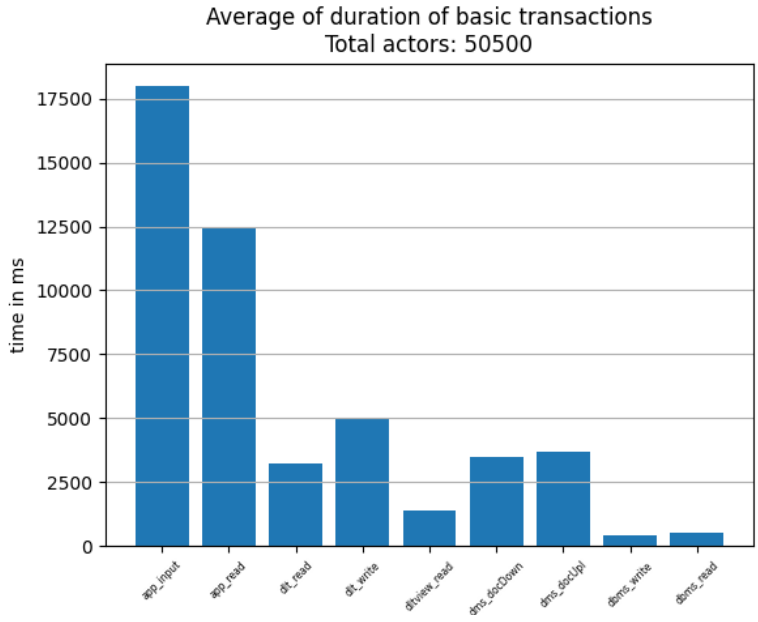
base_tx_name	Nome della transazione base (che è sempre legata ad un componente)
avg	Media, espressa in ms, della durata della transazione base
median	Valore mediano, espresso in ms, della durata precedente
stdDev	Deviazione standard, espressa in ms, della durata
max	Durata, in ms, della transazione che ha impiegato più tempo per essere eseguita
min	Durata, in ms, della transazione che ha impiegato meno tempo per essere eseguita

Figura 5

	A	B	C	D	E	F
1	base_tx_name	avg	median	stdDev	max	min
2	app_input	17984.83536380185	17240	5077.570594859112	57471	5640
3	app_read	12495.800620763337	8998.0	8354.997922552455	93791	1262
4	dlt_read	3197.6702330603066	3024	1017.654519765359	12070	908
5	dlt_write	4996.912693232357	4811	1408.669835120899	15660	1150
6	dltview_read	1399.3848660980843	1372	281.97078541496904	3523	541
7	dms_docDown	3498.914526531764	3427	674.4917882072909	12281	1470
8	dms_docUpl	3699.7877818452766	3614.0	802.422424673473	10904	1521
9	dbms_write	399.68242796735575	369	152.07204874686457	1992	105
10	dbms_read	498.9775479508147	471	156.4030907496774	1966	157
11						

- Il file rinominato `base_txs_duration_actors_<numberOfActors>.png` (in figura 5) riporta un grafico ad istogramma nel quale, per ogni transazione base dell'intera simulazione ne viene visualizzata la durata media in ms (media estratta dal file `.csv` precedentemente nominato)

Figura 6



- Mentre i due file precedenti si contenevano informazioni sulle transazioni base prodotte in tutta la simulazione, i file `interval_base_txs_duration_statistics_actors_<numberOfActors>.csv` e `interval_base_txs_duration_actors_<numberOfActors>.png` sono simili ai precedenti ma i dati che contengono (per ogni tipo di transazione: media della durata, mediana etc..) si riferiscono ad un particolare intervallo temporale stabilito in fase di configurazione. Potrebbe essere interessante, ad esempio, concentrarsi sulla durata media delle transazioni nell'intervallo di tempo in cui il componente ha raggiunto il massimo numero di transazioni. In tutte le simulazioni è stato scelto di concentrarsi sull'intervallo di tempo fra le 12:00 e le 13:00 (tale intervallo è personalizzabile)

- Il file `complex_txs_duration_actors_<numberOfActors>.png` riporta graficamente (figura 7), per ogni transazione complessa definita nel simulatore, la propria durata media, in ms, ottenuta dalle statistiche dei file precedentemente prodotti.

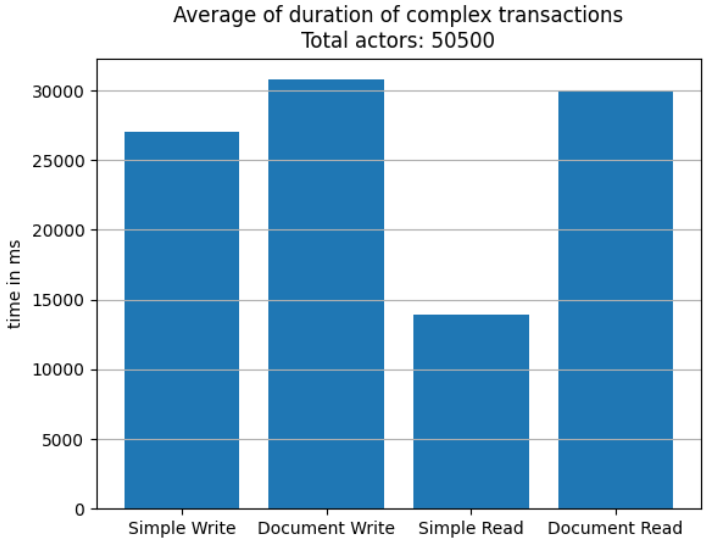


Figura 7

Il numero di transazioni base eseguite ogni secondo da ciascun componente, così come (eventualmente) gli intervalli di tempo in cui il componente stesso ha raggiunto il limite massimo di transazioni gestibili contemporaneamente, sono stampati a video al termine della simulazione (figura 8).

Figura 8

6.4 Risultati ottenuti

Sono state effettuate dieci simulazioni: è stato supposto, per gli attori, un rapporto pari a 1:100, cioè 100 consumatori per ogni operatore. La seguente tabella mostra il numero di attori utilizzato in ogni simulazione:

Nr° Simulazione	Attori	
	#Operatori	#Consumatori
1	200	20.000
2	500	50.000
3	1.000	100.000
4	2.000	200.000
5	5.000	500.000
6	7.500	750.000
7	10.000	1.000.000
8	10.500	1.050.000
9	11.000	1.100.000
10	13.000	1.300.000

In ciascuna delle dieci simulazioni sono state considerate le tempistiche delle transazioni dei componenti descritti nel capitolo 5, la blockchain *Etherna* e la configurazione degli attori definita nel paragrafo 3.1 (numero di transazioni giornaliere, media, deviazione standard).

I risultati delle simulazioni lanciate sono stati aggregati in una cartella "`./ABATA_output`" condivisa al seguente link. Tale cartella è strutturata nel seguente modo:

```
.
└── ABATA_output/
    └── TESTS/
        ├── 200_20.000/
        │   └── report
        ├── 500_50.000/
        │   └── report
        ├── 1.000_100.000/
        │   └── report
        ├── .../
        │   └── ...
        ├── 11.000_1.100.000/
        │   └── report
        └── 13.000_1.300.000/
            └── report
```

All'interno dell'ambiente condiviso è presente una cartella detta "`TEST/`" che contiene al suo interno dieci cartelle (una per ogni simulazione) rinominate nel seguente modo: `<numOfOper>_<numOfCust>`. Ad esempio, la cartella "11.000_1.100.000" presenta gli output generati dal simulatore che ha coinvolto le transazioni lanciate da 10.000 operatori e 1.000.000 consumatori. All'interno della cartella sono presenti gli output generati (file `.csv` e `.png`) e descritti nel paragrafo precedente.

Una volta terminate le dieci simulazioni (che hanno visto variare il numero degli attori in maniera crescente, si veda la tabella precedente) sono stati confrontati gli output al fine di indagare, principalmente, su eventuali relazioni di dipendenza fra tempi medi di esecuzione delle transazioni e

numero degli attori ed eventuali colli di bottiglia (i.e. componenti che raggiungono il limite massimo di transazioni gestibili contemporaneamente).

- **Risultati delle simulazioni da 20.200 a 1.060.500 attori**

È emerso che per le prime otto simulazioni, ovvero quelle che hanno coinvolto un numero di attori fino ad 1.060.500, non ci sono stati comportamenti anomali del simulatore e all'aumentare del numero degli attori fino a tale soglia, per tutti i componenti le transazioni base hanno sempre mantenuto una durata molto simile così come le transazioni complesse. All'aumentare del numero degli attori, sono rimaste molto simili anche le durate relative alle transazioni che hanno avuto la durata peggiore (i.e. durata massima). Inoltre, nessun componente ha raggiunto il numero massimo di transazioni gestibili contemporaneamente. A tal proposito, si vedano le figure 9 e 10, che mostrano, rispettivamente, le statistiche di ciascuna transazione base delle simulazioni che coinvolgono 20.200 attori e 1.060.500 attori, e le figure 11 e 12 che mostrano la durata media di ogni transazione complessa.

***Figura 9**. Statistiche per ogni transazione base (20.200 attori)*

	A	B	C	D	E	F
1	base_tx_name	avg	median	stdDev	max	min
2	app_input	18000.604408465813	17290	5048.934530844104	45145	6359
3	app_read	12489.064248270493	8991	8364.762933824608	74984	1474
4	dlt_read	3201.8313702394976	3022.5	998.864380920667	8017	1230
5	dlt_write	5002.2343704281675	4838	1397.2956924107507	13966	1756
6	dltview_read	1398.9152316296868	1371.0	281.6542481425018	3607	584
7	dms_docDown	3498.396526677232	3426.0	675.2681789253147	9005	1661
8	dms_docUpl	3704.8511339406864	3623	791.3441237740084	7871	1723
9	dbms_write	402.0588120550083	371	155.93674858962322	1629	124
10	dbms_read	498.77840632010145	470	156.5837911967398	1703	185
11						

***Figura 10**. Statistiche per ogni transazione base (1.060.500 attori)*

	A	B	C	D	E	F
1	base_tx_name	avg	median	stdDev	max	min
2	app_input	18009.285587154336	17266.0	5086.105146214073	59331	5381
3	app_read	12497.151095534184	8997	8355.70654141969	105919	1184
4	dlt_read	3197.237254939685	3026	1014.1309397277355	12277	942
5	dlt_write	4997.124007026898	4810.0	1405.5188811741023	16175	1481
6	dltview_read	1399.7050849649445	1372.0	282.27586885676783	3634	523
7	dms_docDown	3498.6849382843666	3427.0	675.2348145697226	13051	1517
8	dms_docUpl	3699.989555007849	3612	802.585585464368	11106	1558
9	dbms_write	399.3262075554562	369.0	151.61899940912102	2151	86
10	dbms_read	499.66896530469154	471.0	157.1085870132872 3	2122	151
11						

Figura 11

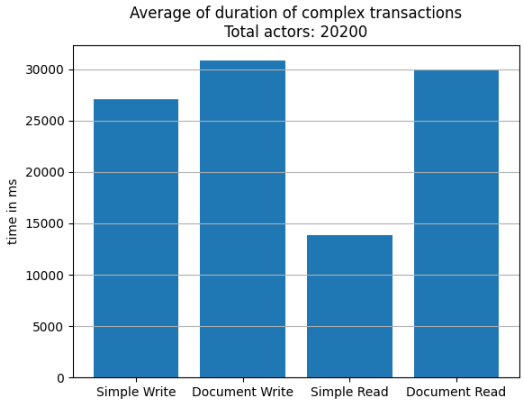
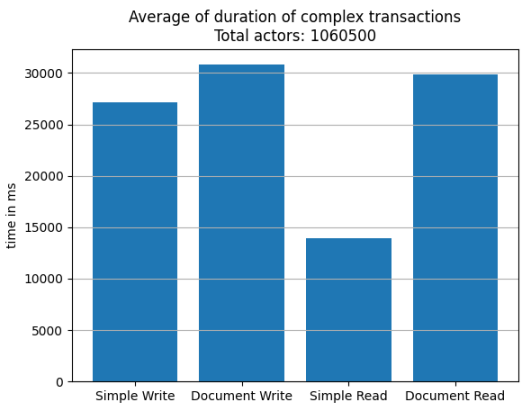


Figura 12



Sono stati confrontati, inoltre, i tempi medi di esecuzione delle transazioni base nell'arco delle 24 ore con i tempi medi di esecuzione negli intervalli di tempo in cui i componenti hanno raggiunto il picco di transazioni: anche in questo caso le durate medie sono state pressoché identiche. Anche il numero di transazioni eseguite ogni secondo, per ogni componente, è rimasto invariato per tutte le simulazioni effettuate.

- **Risultati delle simulazioni da 1.111.000 a 1.313.000 attori**

Le ultime due simulazioni (nona e decima) eseguite hanno coinvolto rispettivamente 1.111.000 e 1.313.000 attori (mantenendo, sempre, un rapporto pari a 100 consumatori per ogni operatore).

È emerso che durante la nona e la decima simulazione, il solo componente che ha raggiunto il numero massimo di transazioni gestibili contemporaneamente è il DLTview utilizzato dagli consumatori (che in queste due simulazioni sono in numero ingente). Esso, infatti, ha raggiunto il picco durante la fascia oraria 13:00-14:00 (si veda la figura 13) facendo aumentare, in tale arco temporale, il tempo di esecuzione delle transazioni ad esso associato. Si noti in figura 14A e 14B come il simulatore stampa il *timestamp* di quando il componente DLTview raggiunge il suo limite massimo di transazioni gestibili.

Figura 13

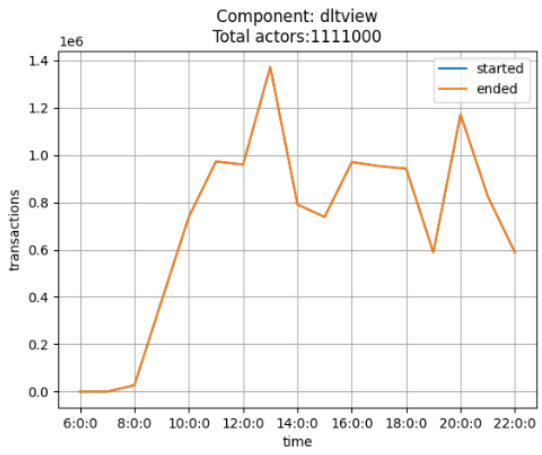


Figura 14A. *Simulazione con 1.111.000 attori*

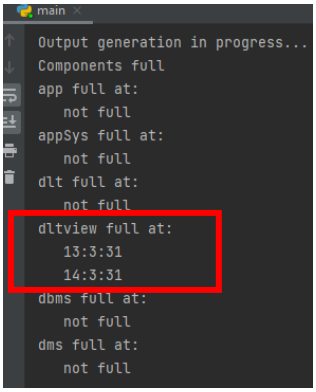
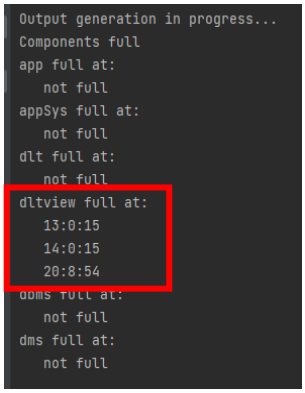


Figura 14B. *Simulazione con 1.313.000 attori*



Riguardo le tempistiche delle transazioni base riguardanti il DLTview, è emerso che nella nona simulazione, che ha coinvolto 1.111.000 attori (in figura 15), nonostante il DLTview abbia raggiunto il limite massimo di transazioni contemporanee, la durata media di tali transazioni non si è discostata da quella delle simulazioni che hanno coinvolto un minor numero di attori: ciò potrebbe essere dovuto al fatto che poche transazioni hanno subìto un ritardo causato dal componente quando esso raggiunge il limite (tale limite è infatti raggiunto solamente in poche fasce orarie). Di contro, nella stessa simulazione il tempo massimo per la transazione base è aumentato, così come è aumentata la deviazione standard passando da circa 282 per 1.060.500 attori (figura 10) a circa 401 per 1.313.000 attori (figura 16).

In figura 16 emerge come anche la durata media delle transazioni del DLTview aumenta (rispetto alle simulazioni precedenti) risentendo dell'alto numero di attori e di conseguenza dell'alto numero di transazioni che essi inviano. L'aumento della durata media delle transazioni base produce anche un

aumento del tempo medio di esecuzione delle transazioni complesse nell'arco di tempo durante il quale è stato raggiunto il limite.

***Figura 15**. Statistiche simulazione con 1.111.000 attori*

	A	B	C	D	E	F
1	base_tx_name	avg	median	stdDev	max	min
2	app_input	18006.62292446792	17261.5	5081.443320683642	62456	5836
3	app_read	12499.104367172666	9001.0	8355.150672748032	102025	1122
4	dlt_read	3198.2055764288584	3026	1015.4669610260436	12044	951
5	dlt_write	4999.564458283368	4814.0	1409.1217220753638	17197	1465
6	dltview_read	1475.310087114268	1413	372.6424594835801	5420.527136541903	507
7	dms_docDown	3498.972855823519	3427.0	675.6626716006174	13407	1443
8	dms_docUpl	3699.105389665435	3611	801.8626249888683	9967	1536
9	dbms_write	399.7447601321278	370.0	152.17443489656827	2130	101
10	dbms_read	499.32716106754253	471.0	157.0172590847383	2054	157
11						

***Figura 16.** Statistiche simulazione con 1.313.000 attori*

	A	B	C	D	E	F
1	base_tx_name	avg	median	stdDev	max	min
2	app_input	18003.82282364509	17264.0	5080.04663873389	65343	5212
3	app_read	12499.13327379569	8999.0	8355.526695854573	121815	1206.4123438447714
4	dlt_read	3203.312176947335	3034.0	1016.5165465921314	12794	865
5	dlt_write	5001.141205590785	4817.0	1408.7588957914584	19710	1171
6	dltview_read	1507.785116123353	1433	401.1085332607355	6084.915221139789	485
7	dms_docDown	3498.9396698150636	3427	675.1016912243016	13056	1401
8	dms_docUpl	3699.7388133113345	3612.0	800.9904943632943	10968	1535
9	dbms_write	399.39767444754904	369.0	151.81825387070774	2179	91
10	dbms_read	499.43828158149876	471.0	156.8479384053209	2138	152
11						

Gli altri componenti, e le statistiche delle transazioni base e ad essi legate, sembrano non aver risentito del numero crescente di attori e transazioni.

Ciò significa che una eventuale e reale implementazione del progetto ABATA, che coinvolge i componenti descritti nel capitolo 5 con blockchain *Etherna* e i rispettivi tempi di esecuzione delle transazioni, può gestire un numero di attori giornalieri pari a 1.060.500 distinto in 10.500 operatori e 1.050.000 consumatori (i.e. decine di milioni di transazioni giornaliere) senza particolari "colli di bottiglia".

È comunque possibile effettuare numerose simulazioni variando i dati dei file `.csv` contenuti nella cartella `./configuration` e nel file "`main.py`" (nel quale si lancia la simulazione, si generano gli output e si definiscono gli eventuali intervalli di tempo su cui ci si deve concentrare) al fine di elaborare gli output e stabilire l'esistenza di altri parametri o fattori che possono influenzare le prestazioni dell'intero sistema. Occorre tener presente che, all'aumentare del numero di attori aumentano il numero di transazioni, e la rispettiva generazione degli output richiede una complessità in termini temporali sempre crescente (calcolo, per ogni componente e per ogni tipo di transazione base della media, mediana, tempo minimo e massimo).

6.5 Tempi di esecuzione

Per tutte le simulazioni è stato calcolato il rispettivo tempo di esecuzione, tenendo conto che l'architettura hardware utilizzata è Intel(R) Core (TM) i5-10600K CPU (4.10GHz) con RAM 32GB. I risultati sono mostrati nella seguente tabella. Tali tempi includono anche la generazione di tutti gli oggetti (es. eventi e rispettive transazioni base per ogni attore).

Simulazione	**#Attori**	**Tempo (in secondi)**
1	20.200	9 s
2	50.500	30 s
3	101.000	62 s
4	202.000	125 s
5	505.000	310 s
6	707.500	524 s
7	1.010.000	655 s
8	1.060.500	727 s
9	1.111.000	1075 s
10	1.313.000	1577 s

Bibliografia

[1] Bassil, Y. (2012). A comparative study on the performance of the Top DBMS systems. arXiv preprint arXiv:1205.2889.

[2] Ilić, M., Kopanja, L., Zlatković, D., Trajković, M., & Ćurguz, D. (2021, June). Microsoft SQL Server and Oracle: Comparative performance analysis. Book of proceedings of the 7th International conference Knowledge management and informatics, Vrnjačka Banja.

[3] Goltsis, A. (2022). A Performance Comparison of SQL and NoSQL Database Management Systems for 5G Radio Base Station Configuration. Online at: https://www.diva-portal.org/smash/get/diva2:1681550/FULLTEXT01.pdf

[4] Ilić, S., Ilić, S., Milovanović, I., & Miljković, D. (2022). A Comparison of Query Execution Speeds for Large Amounts of Data Using Various DBMS Engines Executing on Selected RAM and CPU Configurations. Tehnički vjesnik (Technical Gazette), 29(1), 346-353.

[5] (2010) Benchmark Results: Scalability of Alfresco Content Management System in a Unisys ES7000/one Enterprise Server Environment, White Paper, Unisys Corporation, online at: https://www.slideshare.net/davidftv/alfresco-benchmark-reportbl100093