Quarkus car-rental and E2E workflow execution time analysis

Edoardo Sarri

Software Architecture and Methodologies
&
Quantitative Evaluation of Stochastic Models

Settembre 2025

Car Rental

Applicazione

Applicazione a microservizi sviluppata in *Quarkus in Action*.

Architettura

Composta da cinque microservizi:

- Billing-service
- Inventory-service
- Rental-service
- Reservation-service
- Users-service

Car Rental

Dipendenze

I microservizi principali hanno delle dipendenze esterne:

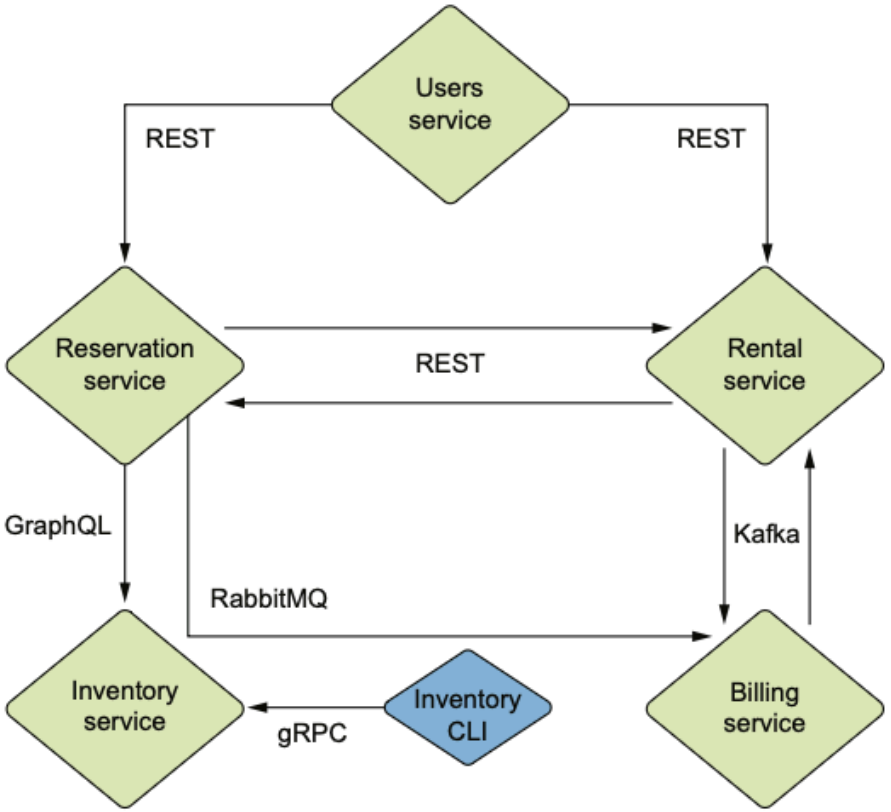
- Billing-service: MongoDB, Kafka, RabbitMQ.
- Inventory-service: MySQL.
- Rental-service: MongoDB, Kafka.
- Reservation-service: PostgreSQL, RabbitMQ.

Car Rental

Comunicazione

La comunicazione tra servizi avviene in vari modi:

- REST
- GraphQL
- Kafka
- RabbitMQ
- gRPC



Deployment

Tecnologie

- Minikube
- Docker
- Helm

File

I file interessanti sono:

- *application.properties*
- *pom.xml*
- Codice di *users-service*: rimosso servizio di autenticazione *KeyCloak*; unico utente *guest*.

Deployment

Problemi risolti

- Il deployment era per OpenShift
- Le immagini venivano caricare su *Quay.io*
- Riferimenti da settare

Estensioni

```
<dependency>
    <groupId>io.quarkus</groupId>
    <artifactId>quarkus-container-image-docker</artifactId>
</dependency>
<dependency>
    <groupId>io.quarkus</groupId>
    <artifactId>quarkus-kubernetes</artifactId>
</dependency>
```

Deployment

Problemi risolti

- OpenShift
- Registry
- Riferimenti

Kubernetes

```
quarkus.container-image.build=true
quarkus.container-image.push=false
quarkus.kubernetes.deployment-target=kubernetes
quarkus.kubernetes.service-type=NodePort
quarkus.kubernetes.image-pull-policy=Never
```

Riferimenti

```
%dev.quarkus.rest-client.reservations.url=http://localhost:8081
%prod.quarkus.rest-client.reservations.url=http://reservation-service
```

Deployment

Helm

Usato per il deployment di servizi esterni.

- Mette a dispozione Chart su ArtifactHUB.
- Deployment personalizzato tramite *value.yaml* o CLI.

MySQL Helm chart

```
helm install mysql-inventory bitnami/mysql \
    --set auth.rootPassword=root-pass \
    --set auth.database=mysql-inventory \
    --set auth.username=user \
    --set auth.password=pass
```

Health

Problema

Il microservizio *reservation-service* in stato *Running* e *Ready*, ma senza connesione al database PostgreSQL.

Soluzione e motivazione

- Riavviare manualmente il pod.
- *reservation-service* necessita che il database sia avviato e pronto a ricevere richeiste.

Health

StartUp probs

Definita una custom health check in *DatabaseConnectionHealthCheck.java* annotata con *@Startup*.

Estensioni

```
<dependency>
    <groupId>io.quarkus</groupId>
    <artifactId>quarkus-smallrye-health</artifactId>
</dependency>
```

Configurazioni

```
quarkus.kubernetes.startup-probe.failure-threshold=1
quarkus.datasource.jdbc.url=
    =jdbc:postgresql://postgresql-reservation:5432/reservation
```

Tracing

Tecnologie

- OpenTelemetry (OTLP)
- Jaeger

Estensioni

```
<dependency>
    <groupId>io.quarkus</groupId>
    <artifactId>quarkus-opentelemetry</artifactId>
</dependency>
```

Tracing

Jaeger: configurazione

```
quarkus.otel.service.name=inventory-service
quarkus.otel.exporter.otlp.endpoint=
    =http://jaeger-collector:4317
quarkus.otel.traces.sampler=always_on
```

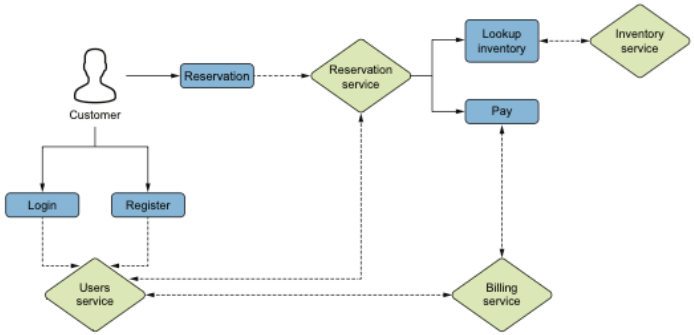
Jaeger: Helm deployment

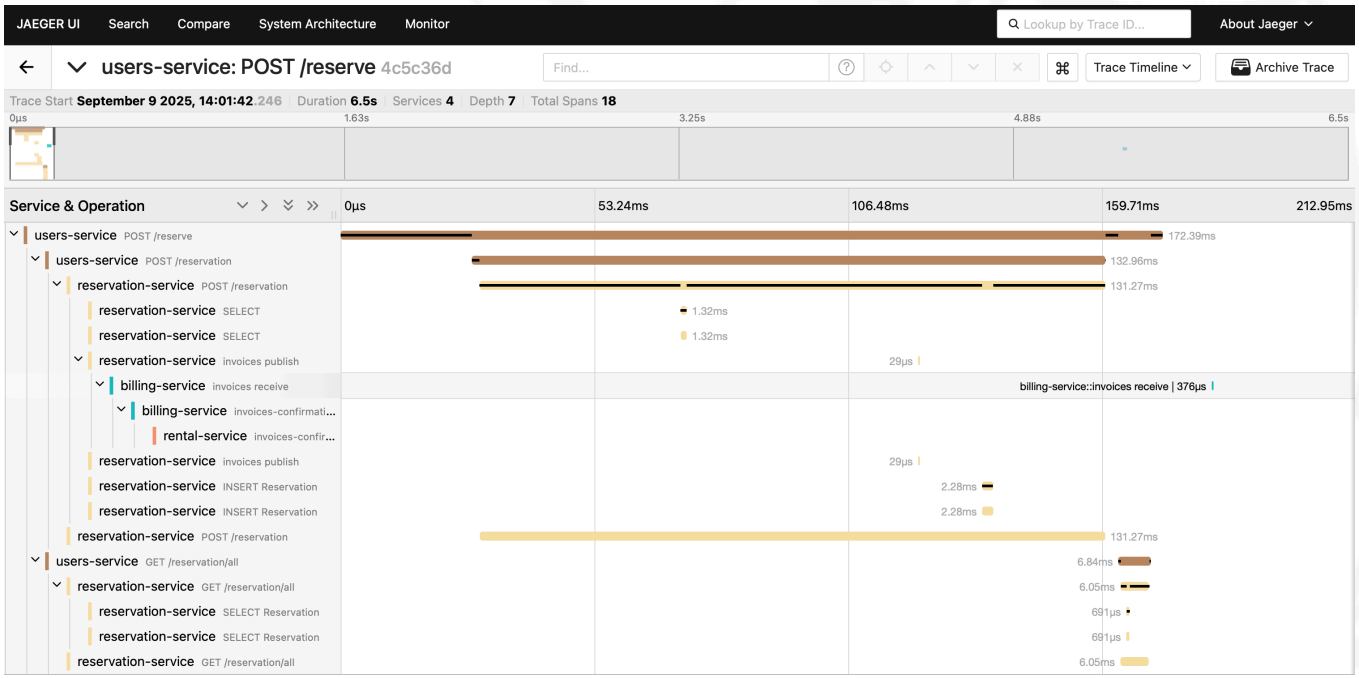
```
helm install jaeger jaegertracing/jaeger \
    --set allInOne.enabled=true \
    --set agent.enabled=false \
    --set collector.enabled=false \
    --set query.enabled=false \
    --set provisionDataStore.cassandra=false \
    --set storage.type=memory
```

Tracing

Use case

Utente che effettua una prenotazione.





Metrics

Tecnologie

- Prometheus
- Grafana

Configurazione

```
# prometheus and grafana
quarkus.kubernetes.prometheus.
    generate-service-monitor=true
quarkus.kubernetes.labels.release=prometheus
```

Metrics

Deployment

Chart Helm con i server Prometheus e Grafana insieme.

```
helm install prometheus \
    prometheus-community/kube-prometheus-stack \
    --set grafana.service.type=NodePort \
    --set grafana.adminUser=admin \
    --set grafana.adminPassword=admin \
    --set grafana.fullnameOverride=grafana
```

Quarkus car-rental and E2E workflow execution time analysis

Edoardo Sarri

Software Architecture and Methodologies
&
Quantitative Evaluation of Stochastic Models

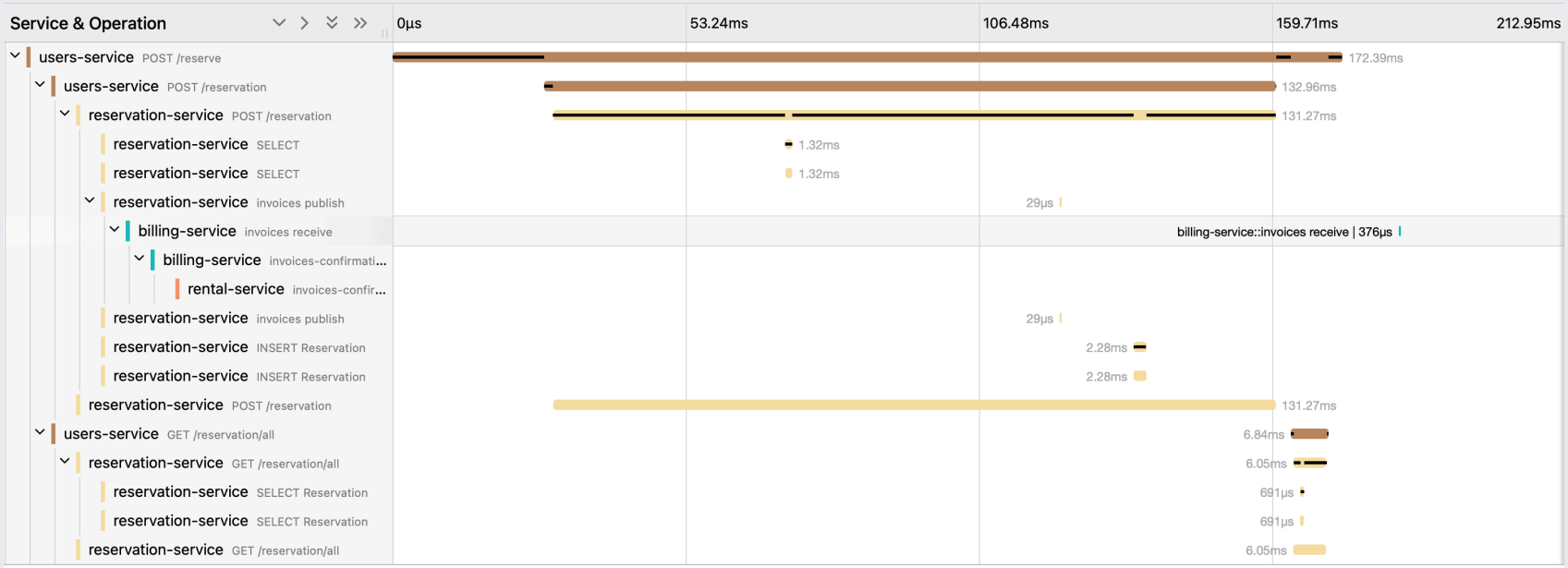
parte 2

Workflow

Obiettivo

Simulare un workflow complesso, i.e., non totalmente sequenziale.

Workflow precedente - jaeger



Workflow

Blocchi aggiunti

Aggiunti due tipi di blocchi:

- XOR
  Scelta esclusiva pesata tra più flussi.
- AND
  Esecuzione parallela di più flussi.

Implementazione

Microservizi che eseguono una busywait.

- XOR: tre scelte con probabilità $p_1 = 0.2$, $p_2 = 0.5$ e $p_3 = 0.3$.
- AND: due server che espongono API gRPC.

Workflow

Implementazione

Microservizi che eseguono una busywait.

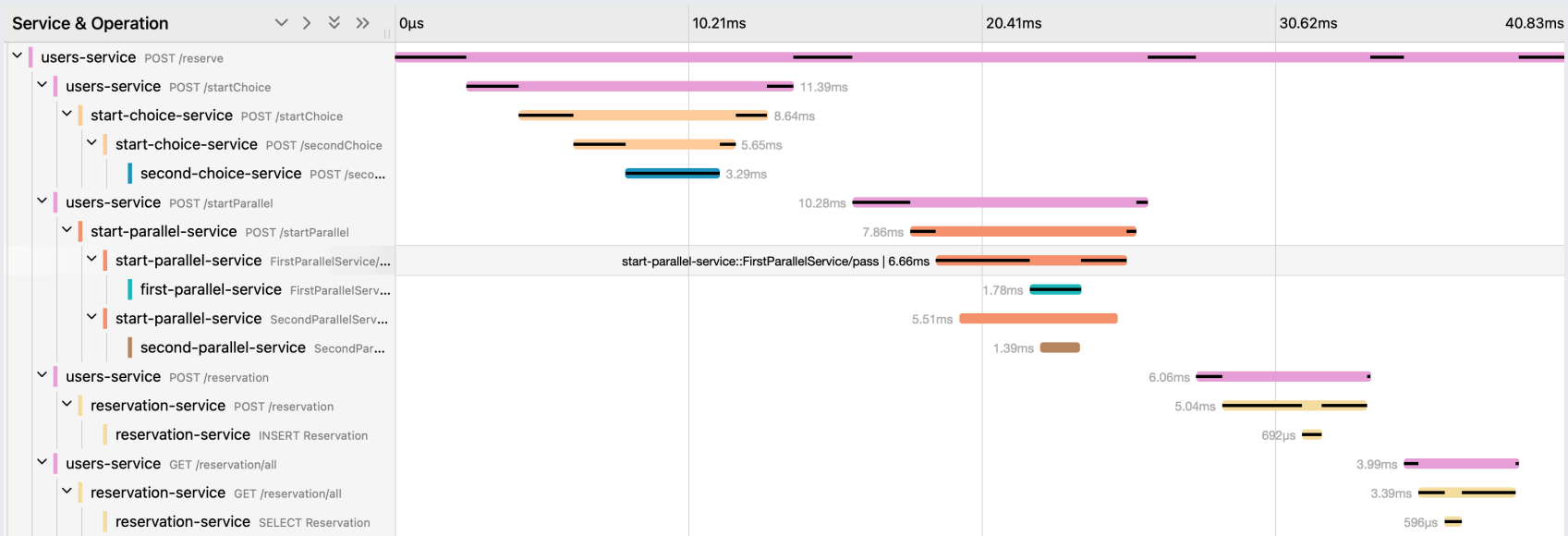
- XOR: tre scelte con probabilità $p_1 = 0.2$, $p_2 = 0.5$ e $p_3 = 0.3$.
- AND: due server che espongono API gRPC.

gRPC - Quarkus

- Il server definisce il file *.proto*. Con l'estensione *quarkus-grpc* quando si fa building, Quarkus costruisce tutte le interfacce.
- Il server implementa un'interfaccia e ridefinisce i suoi metodi.
- Il clinet compila il file *.proto* e può chiamare i metodi del server come fossero privati.

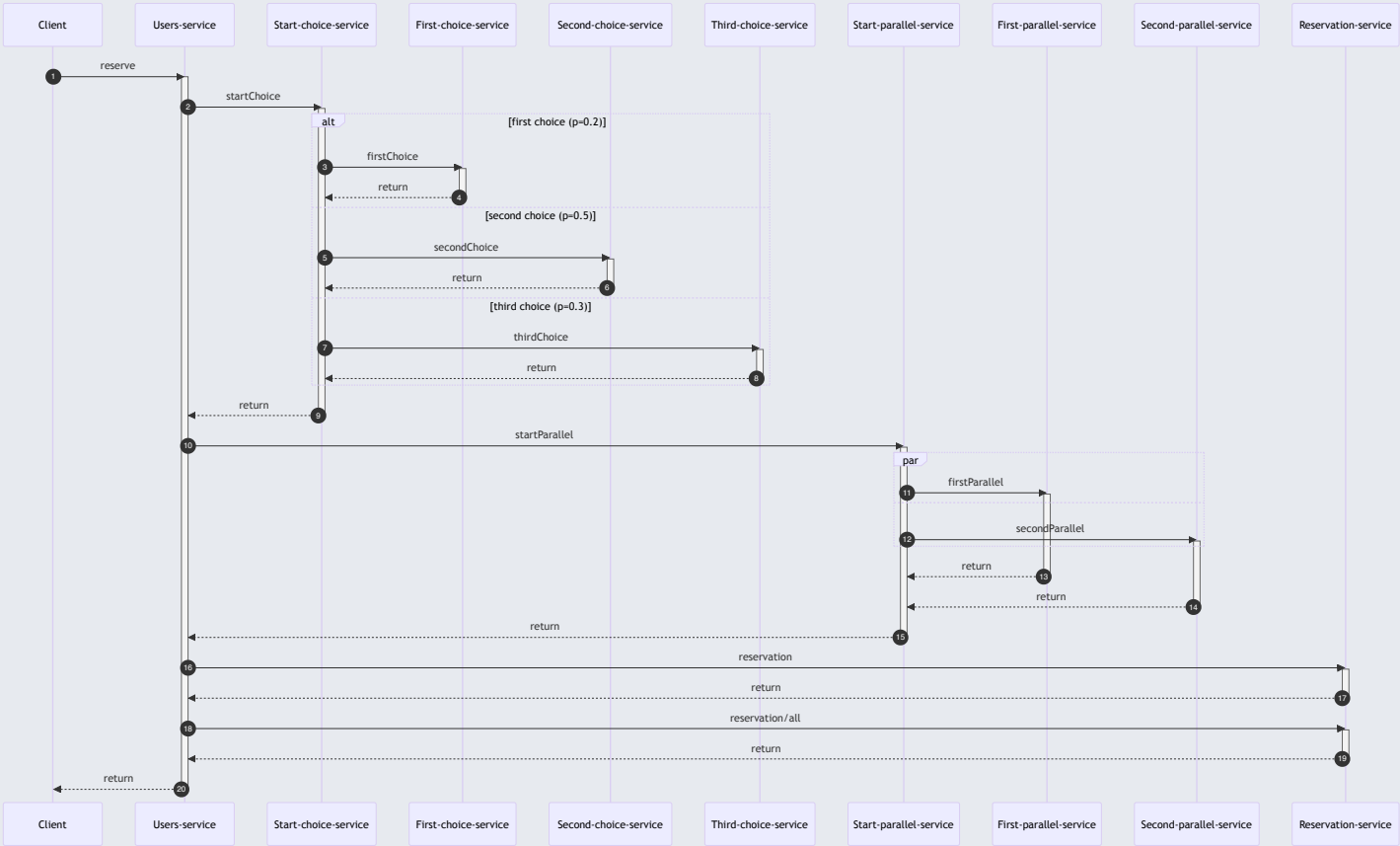
Workflow

Nuovo workflow - jaeger



Workflow

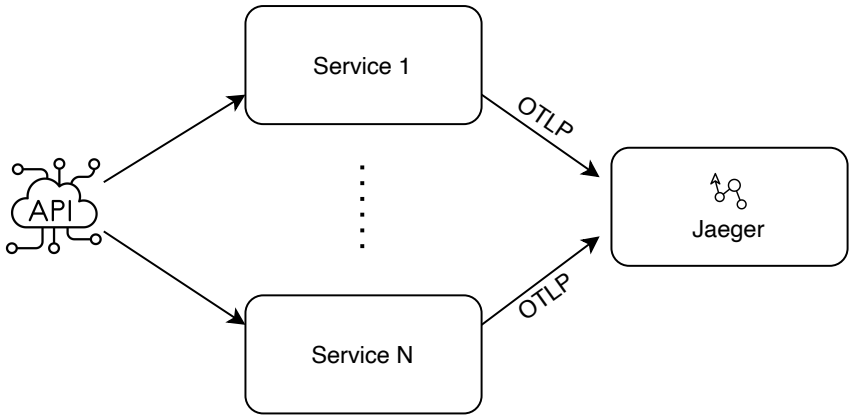
Nuovo workflow - sequence diagram



Tracing

Tracing precedente

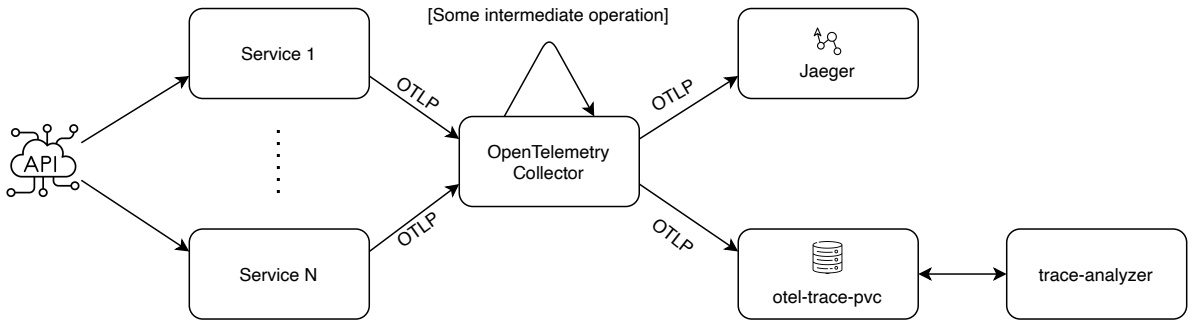
- I microservizi ricevono richieste.
- OpenTelemetry raccoglie i dati dai microservizi. In Quarkus questo è semplice tramite l'estensione *quarkus-opentelemetry*.
- OTel invia al backend (e.g., Jaeger). Serve configurazione in *application.properties*.



Tracing

Tracing attuale - Flusso

- I microservizi ricevono richieste.
- OpenTelemetry raccoglie i dati dai microservizi. In Quarkus questo è semplice tramite l'estensione *quarkus-opentelemetry*.
- OTel invia le statistiche all'Otel-collector, dove possiamo eseguire operazioni intermedie (e.g., filtraggio, aggregazione).
- Il collector invia i dati a vari back-end.



Tracing

Tracing attuale - Conseguenze

- Si aggiuge complessità, soprattutto rispetto alla versione con la sola estensione Quarkus.
- Per progetti semplici (o all'inizio di uno complesso) è meglio semplificare.
- Per progetti complessi, che devono scalare, o se è necessario eseguire operazioni intermedie, è necessario il collector.

Analisi E2E

Obiettivo

Derivare una distribuzione (CDF) del tempo E2E del workflow in modo data-driven.

Scenario

- Caso medio: Si usano le statistiche descrittive (i.e., media e varianza).
- Soft-real time: per garantire un dato Service Level Agreements (SLA), si deve lavorare su un upper-buond del tempo di esecuzione.

Analisi E2E

Possibilità di analisi

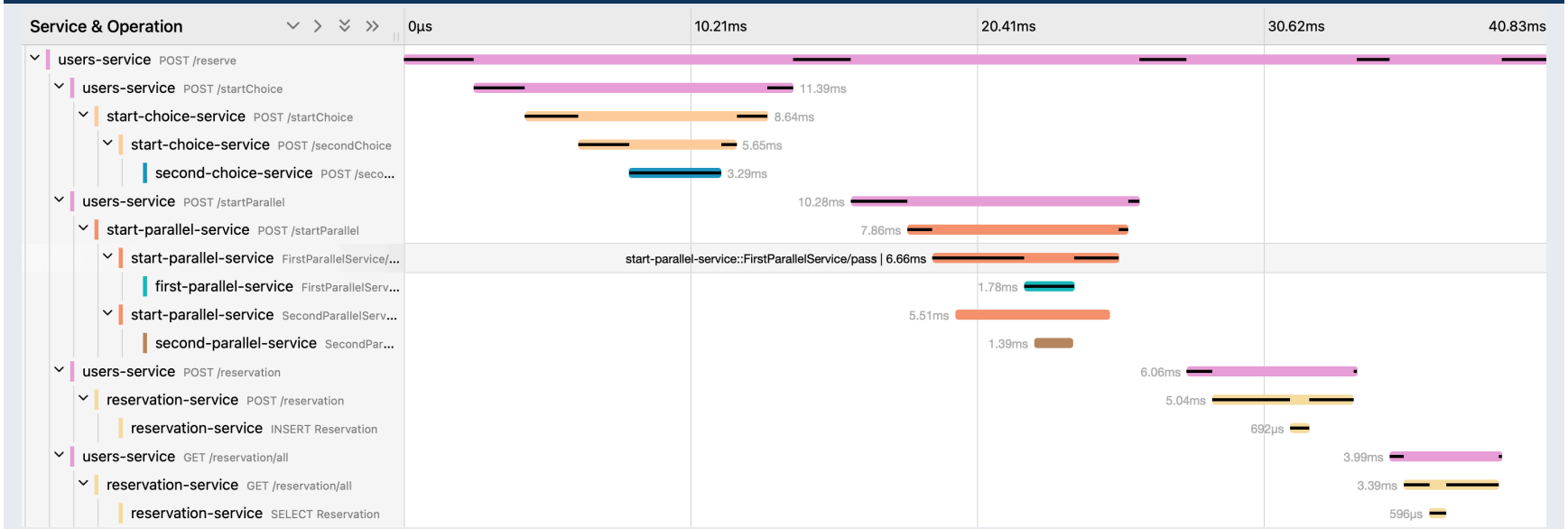
- Esatta: Fattibile per workflow non complessi e/o tempi di esecuzine Markoviani. In altri casi lo spazio degli stati esplode.
- Composizionale: Nello stato i-esimo si elimina il condizionamento sul tempo di esecuzione dello stato $i-1$-esimo. Si analizza in modo indipendente ogni blocco e si ricostruisce l'analisi E2E in modo bottom-up.

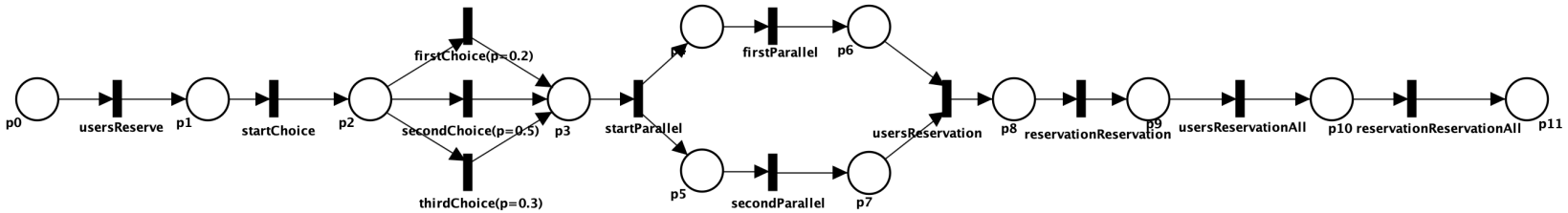
Dati necessari

- **Workflow** delle chiamate.
- **Distribuzioni marginali** dei tempi di esecuzione.

Analisi E2E

Workflow





Analisi E2E

Distribuzioni

L'obiettivo è derivare la distribuzione che meglio approssima il tempo di esecuzione di ogni microservizio. Serve:

- Generare molte tracce.
- Ottenere statistiche descrittive dei tempi di esecuzine.
- Fitting delle distribuzioni su tali dati.

Generazione

La generazione è stata eseguita con K6.

- Job rilasciato con Helm.
- Ciclo di 30 secondi che chiama iterativamente *users/reserve*.

Analisi E2E

Statistiche

A partire dalle tracce analizzate nel PVC collegato al pod *analyzer*.

Service	Operation	Min (ms)	Max (ms)	Mean (ms)	Variance	Std Dev	CV
users-service	POST /reserve	12.888	786.823	30.005	848.737	29.133	0.971
start-choice-service	POST /startChoice	1.735	680.265	7.623	358.586	18.936	2.484
third-choice-service	POST /thirdChoice	0.387	66.357	2.257	20.595	4.538	2.011
users-service	POST /startParallel	1.380	99.777	5.424	60.994	7.810	1.440
start-parallel-service	POST /startParallel	2.836	242.163	11.773	302.806	17.401	1.478
first-parallel-service	FirstParallelService/pass	0.177	76.629	2.053	22.980	4.794	2.335
second-parallel-service	SecondParallelService/pass	0.163	62.763	1.955	17.488	4.182	2.139
users-service	POST /reservation	1.281	90.195	3.892	47.608	6.900	1.773
reservation-service	POST /reservation	1.307	167.913	4.022	69.307	8.325	2.070
users-service	GET /reservation/all	1.422	86.108	5.220	44.638	6.681	1.280
reservation-service	GET /reservation/all	1.574	159.313	5.765	77.672	8.813	1.529
second-choice-service	POST /secondChoice	0.359	73.874	2.127	13.521	3.677	1.729
first-choice-service	POST /firstChoice	0.394	84.829	2.296	31.756	5.635	2.455

Analisi E2E

Statistiche

Si sono usato gli approssimanti di Whitt:

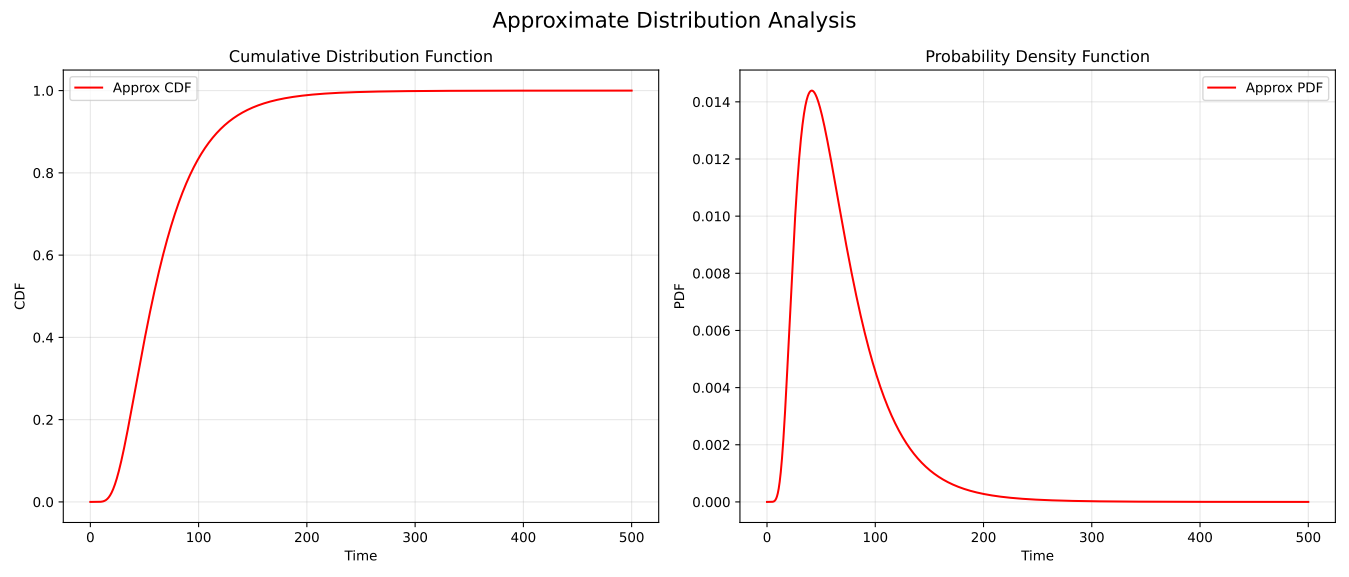
- $C_V = 1$: Exp.
- $C_V > 1$: HyperExp, XOR di due esponenziali.
- $\frac{1}{\sqrt{2}} \leq C_V < 1$: HypoExp: AND di due esponenziali.

Service	Operation	distribution	p1	Lambda 1	p2	Lambda 2
users-service	POST /reserve	hypoExp	N/A	0.0343	N/A	1.1294
start-choice-service	POST /startChoice	hyperExp	0.9246	0.2426	0.0754	0.0198
third-choice-service	POST /thirdChoice	hyperExp	0.8884	0.7874	0.1116	0.0989
users-service	POST /startParallel	hyperExp	0.7955	0.2933	0.2045	0.0754
start-parallel-service	POST /startParallel	hyperExp	0.8050	0.1368	0.1950	0.0331
first-parallel-service	FirstParallelService/pass	hyperExp	0.9153	0.8916	0.0847	0.0825
second-parallel-service	SecondParallelService/pass	hyperExp	0.9004	0.9210	0.0996	0.1019
users-service	POST /reservation	hyperExp	0.8596	0.4417	0.1404	0.0721
reservation-service	POST /reservation	hyperExp	0.8942	0.4447	0.1058	0.0526
users-service	GET /reservation/all	hyperExp	0.7459	0.2858	0.2541	0.0974
reservation-service	GET /reservation/all	hyperExp	0.8165	0.2833	0.1835	0.0637
second-choice-service	POST /secondChoice	hyperExp	0.8530	0.8021	0.1470	0.1382
first-choice-service	POST /firstChoice	hyperExp	0.9229	0.8041	0.0771	0.0672

Analisi E2E

Distribuzione E2E (Eulero)

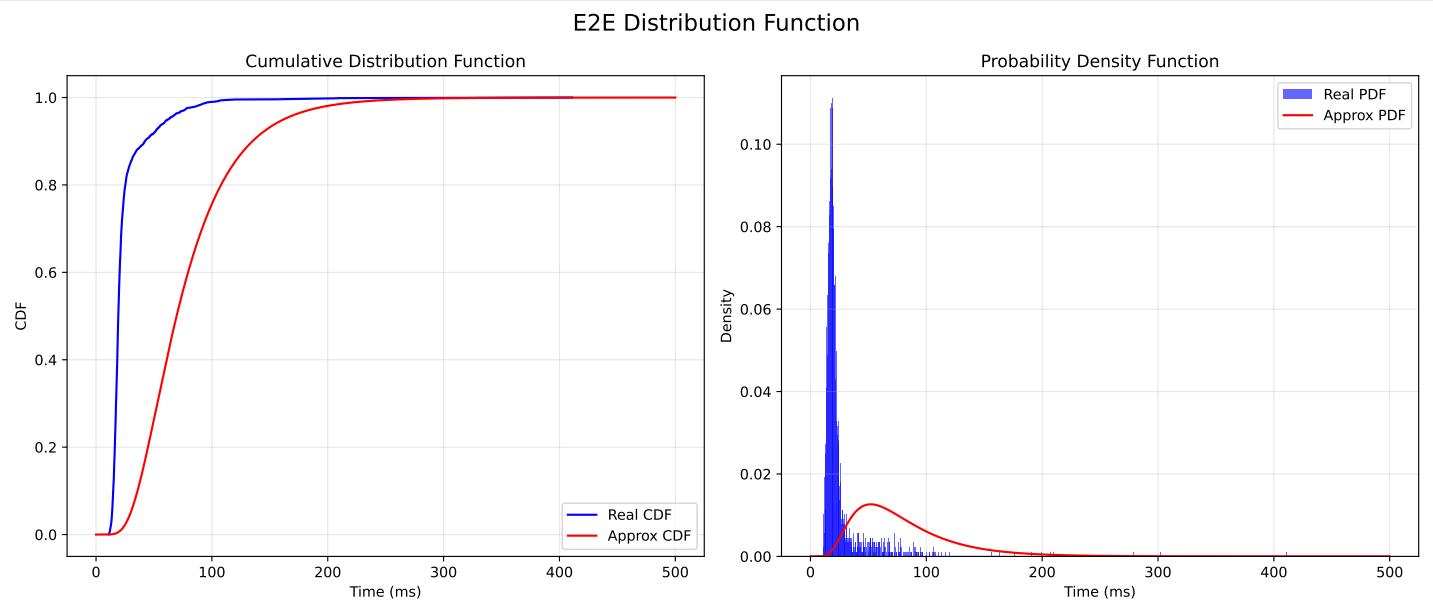
Di seguito la CDF e PDF del tempo di esecuzione E2E.



Analisi E2E

Distribuzione E2E reali vs approssimate

- Questo se nono conoscessimo la vera distribuzione del tempo di esevuzione E2E.
- Confrontando la reale distribuzione con quella approssimata:



Quarkus car-rental and E2E workflow execution time analysis

Edoardo Sarri

Grazie