coders51

Business Case

orkestro

Orkestro: da Monolite a Microservizi

Orkestro (https://orkestro.com/) è una start-up tecnologica con sede a Londra che offre un servizio di last mile delivery collegando i rivenditori con un marketplace di corrieri specializzati.

Offre una soluzione di gestione delle consegne che aiuta le aziende ad individuare il corriere più conveniente per le loro esigenze e monitorare le consegne locali, comunicare con autisti e clienti e scalare rapidamente le operazioni di consegna.

Orkestro ha accesso ad un'ampia rete di corrieri e società di consegna e si integra con le piattaforme ePOS ed eCommerce.

> *Working with coders51 has been both a pleasure and great experience - they helped us refactor our platform allowing us to continue focusing on building out features for our clients. This replatforming was essential for both the growth of traffic as well as agility in designing new products - leveraging their knowledge in domain driven design amongst other expertise has helped us instill better engineering practices.*
>
> *As any developer will know, replatforming takes a long time, and having coders51 by our side through the whole journey has been a major advantage for us. I am very pleased with the overal result, we are now able to deploy new features faster and more reliably giving us the speed and agility to thrive in a fast moving industry.*
>
> **Daryl Rodrigo**
> CTO @ Orkestro

Le necessità del cliente

L'obiettivo era chiaro, ma non semplice: poter velocizzare gli sviluppi delle nuove funzionalità e semplificare il deploy.

Al momento i deployment erano complessi e poco sicuri vista l'architettura monolitica della piattaforma.

Le numerose interdipendenze del monolite inoltre rendevano complesso lo sviluppo di nuove funzionalità e rischiosa la loro messa in produzione.

Era inoltre necessario, rendere i processi più agili e la piattaforma più sicura e meno soggetta a malfunzionamenti, in particolare in alcune aree critiche del sistema.

Per fare questo si è deciso di intraprendere un percorso di migrazione e di evoluzione della piattaforma verso un'architettura a microservizi.

Era necessario che la transizione avvenisse in assoluta sicurezza in modo da non impattare minimamente sul lavoro e sui clienti già attivi della piattaforma che già stava producendo business.

La soluzione

Inizialmente il nostro lavoro è stato quello di intervenire sul monolite per renderlo capace di comunicare con i nuovi microservizi che andavamo a sviluppare in modo da farli operare in parallelo ad esso.

Quando eravamo certi che una nuova funzionalità fosse stata correttamente sviluppata nei nuovi microservizi "spegnevamo" la vecchia funzionalità del monolite.

Continuando ad applicare questo pattern abbiamo via via svuotato il monolite dalle funzionalità che abbiamo ritenuto più importanti da isolare e da far evolvere in futuro.

In parallelo con il lavoro di refactoring, nuove funzionalità venivano poi sviluppate immediatamente in nuovi microservizi e integrate con il vecchio monolite attraverso **RabbitMQ**.

L'utilizzo dello strangler pattern e di vari feature flag ci ha permesso di attivare o disattivare le varie funzionalità, sia nel monolite che e nei vari microservizi, a runtime senza avere mai nessun downtime.

La maggior parte della comunicazione tra i microservizi è basata su uno scambio asincrono di messaggi via RabbitMQ. Questo ci ha portato a ottenere una alta resilienza del sistema in quanto anche momentanei blocchi o rallentamenti di alcuni microservizi non hanno un impatto su tutto il sistema.

25+
Microservizi

> 15
Deploy per Day

99.99%
Availability

I vantaggi per il cliente

Il nostro intervento ha portato vantaggi in 2 aree principali:

- scalabilità e fault-tollerance
- facilità del deployment

L'impatto sui deploy è stato evidente. Con la migrazione ai microservizi i deployment sono passati da uno ogni due settimane a più di 15 deploy per giornata.

Questo perché la nuova architettura permette finalmente di fare deploy di singole funzionalità o correzioni senza impatti su tutta la piattaforma, ma solo su servizio relativo.

Abbiamo quindi abbattuto l'annosa paura dei crash della piattaforma legata ai deploy non andati a buon fine, rendendo tutto il team di sviluppo più veloce, agile e sicuro nelle operazioni.

L'identificazione delle problematiche attuali e la loro risoluzione per step, senza mai interrompere il lavoro del team di sviluppo e del business aziendale, ci ha permesso di essere quasi trasparenti e rilasciare il nuovo sistema a microservizi senza intoppi o frizioni.

I prossimi passi

Le nuove funzionalità vengono naturalmente sviluppate in nuovi microservizi.

La migrazione di funzionalità dal monolite ai vari microservizi continuerà finché il monolite stesso non diventerà più gestibile.

L'obiettivo finale di questo lavoro non è mai stato l'uccisione del monolite ma il miglioramento del processo di sviluppo e di implementazione di nuove business feature.