CENTRO REGIONALE
INFORMATION COMMUNICATION TECHNOLOGY
CERICT SCRL

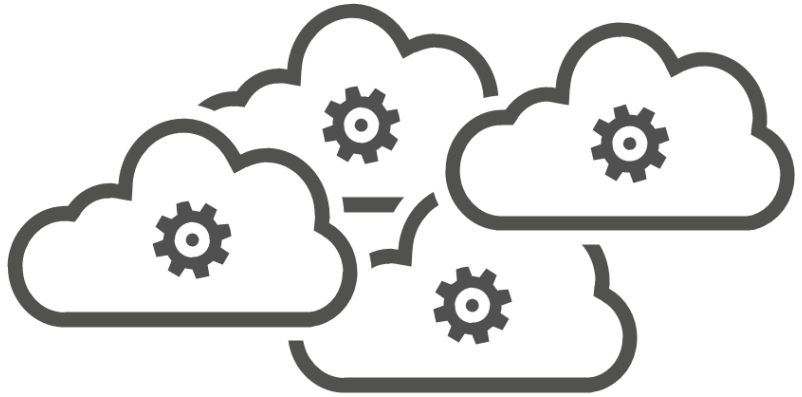
myway2go

Soluzione architetturale adottata per il progetto MyWay2Go di R&D

Architettura a Micoservizi

Argomenti trattati:

1. Cosa sono i **microservizi** e le **caratteristiche**
2. **Componenti** architettura
3. I microservizi nel contesto delle **smart city**
4. Il progetto **MyWay2Go**
5. Le **componenti** di MyWay2Go
6. Benefici nel **contesto aziendale**
7. **Esperimenti** e **prospettive** future
8. **Conclusioni**

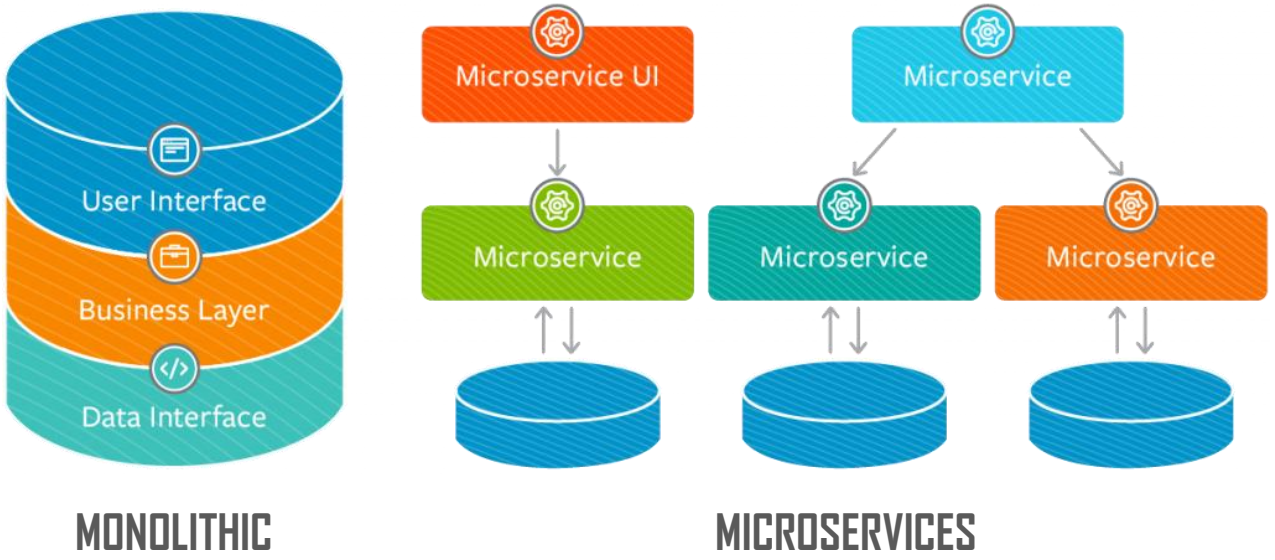


Cosa sono i Microservizi?

Un'architettura a *microservizi* è un approccio allo sviluppo di una singola applicazione come un insieme di piccoli servizi.

Ogni (*micro*)servizio viene progettato e sviluppato a seconda delle business capability del dominio, viene eseguito indipendentemente e può essere *deployato* singolarmente e in automatico.

La logica dei *microservizi* può essere implementata con diversi linguaggi di programmazione e diverse tecnologie di storage.



Si distingue da una soluzione monolitica poiché suddivide il sistema nelle sue funzioni di base.

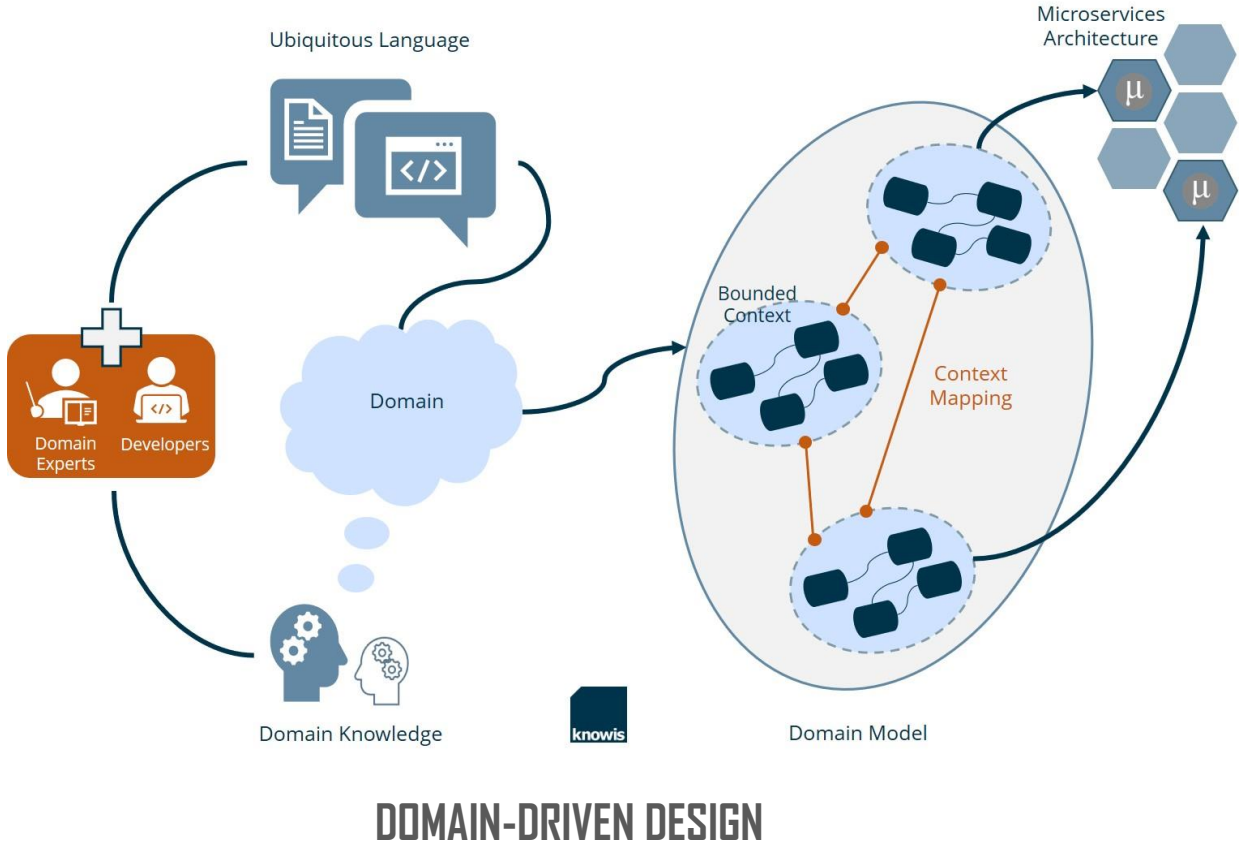
Caratteristiche dei microservizi

Il concetto di microservizi deriva dallo schema Bounded Context definito dal Domain-Driven Design (DDD) che, gestisce modelli di grandi dimensioni, suddividendoli in più contesti delimitati. Ogni contesto deve avere il proprio modello e il proprio database, con un linguaggio comune per le comunicazioni tra sviluppatori software ed esperti di dominio.

Questo ci permette di definire 3 caratteristiche fondamentali:

- **Basso Accoppiamento (Loose Coupling)**
- **Alta Coesione**
- **Buonded Context**

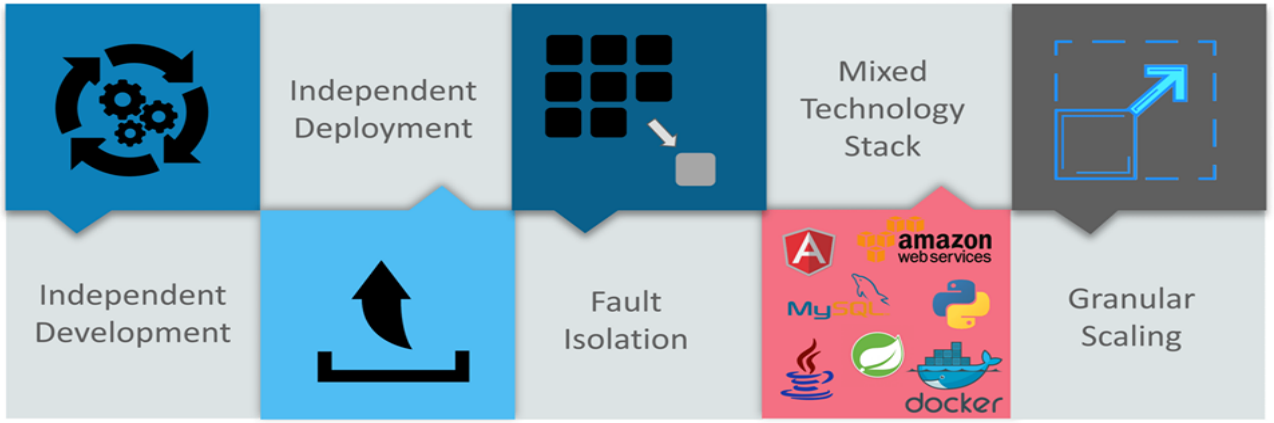
Esse permettono di apportare continui cambiamenti e aggiornamenti al singolo microservizio, nonchè di circoscriverne le anomalie, senza necessità di intervenire sulle altre componenti.



Caratteristiche dei microservizi

Altre caratteristiche fondamentali di un'architettura a microservizi, sono:

- **Resilienza:** un problema su un microservizio non pregiudica il funzionamento degli altri
- **Scalabilità:** facilità di distribuire i microservizi su più server, a seconda delle esigenze
- **Deploy *fast,* continuo e automatizzato (CI/CD):** possibilità di velocizzare ed automatizzare i processi di deployement ed integrazione
- **Eterogeneità tecnologica:** ogni microservizio può essere sviluppato in qualsiasi linguaggio di programmazione
- **Accessibilità:** facilità per gli sviluppatori di accedere alle componenti e quindi accelerare i cicli di sviluppo, anche in ambito Agile



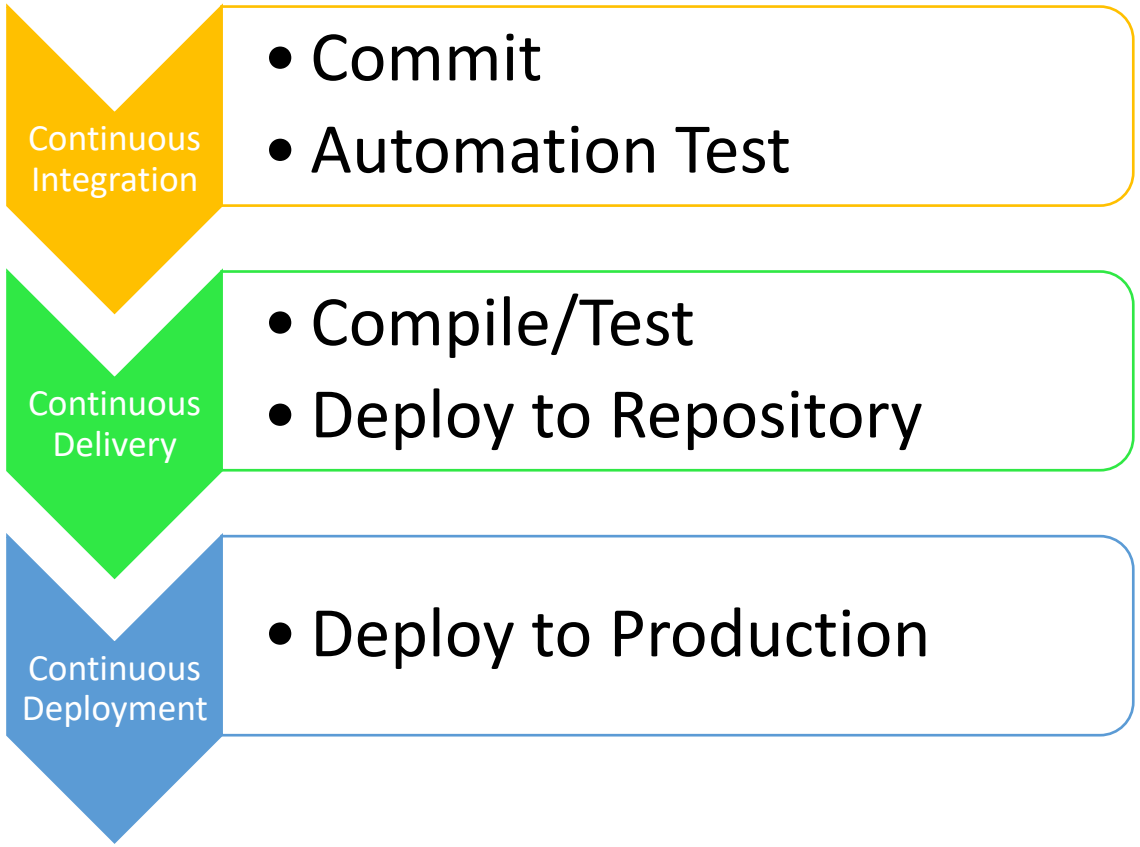
Continuous: Integration, Delivery, Deployment

Tra i maggiori vantaggi che si ottengono tramite l'utilizzo di un'architettura a microservizi, è quella di poter applicare a pieno i processi di Continuous Integration, Delivery e Deployment.

Continuous Integration: l'integrazione continua fa riferimento ad un processo di automazione per gli sviluppatori, i quali vanno ad effettuare continui (e piccoli) rilasci regolarmente.

Continuous Delivery: la *Delivery* invece si occupa delle distribuzione del software, infatti compila le nuove modifiche, avvia la fase di test, e unisce tutto in un repository.

Coninuous Deployment: è la fase finale del processo, in cui il codice compilato e testato viene rilasciato in produzione.



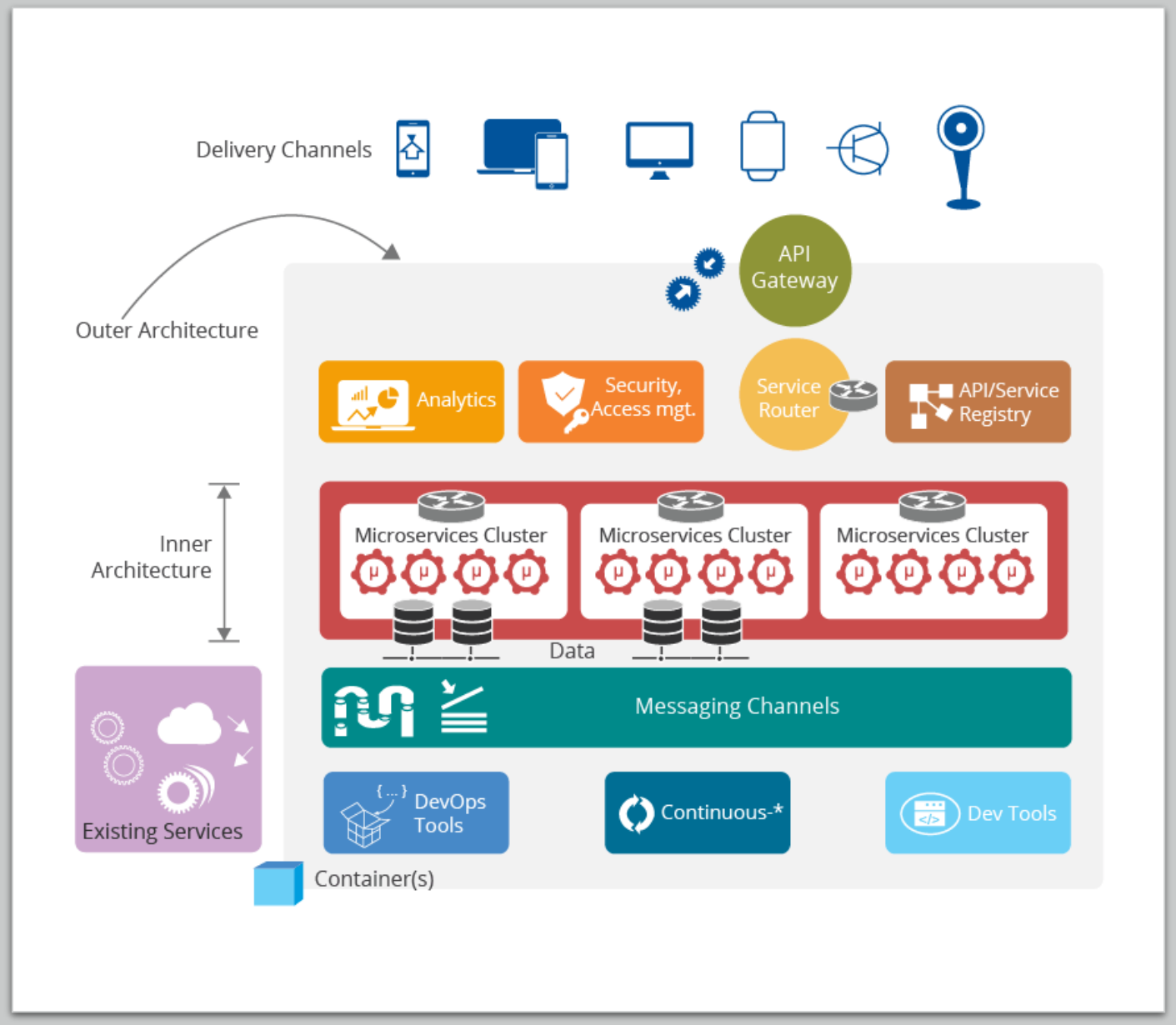
Casi di successo

- Netflix
- Amazon
- Ebay
- Spotify
- Uber
- Zalando
- Groupon
- Zoom

Componenti Architettura

Le principali componenti su cui si basa un'architettura a microservizi sono:

- **API Gateway**
- **Security Access Management**
- **API Service Registry**
- **Analytics**
- **Log Monitor**
- **Messaging Channel**



Componenti Architettura

API GATEWAY: il gateway e la componente che si occupa di ricevere le richieste dall'esterno dell'architettura (es. Webapp, Mobile, ecc.) che poi smista verso il destinatario.

API SERVICE REGISTRY: Gestisce un «registro dei servizi» in modo che ogni modulo può registrarvisi e rendersi disponibile per ricevere le richieste. In questo modo il gateway ha a disposizione le informazioni centralizzate per comunicare con tutti i microservizi della piattaforma.

MESSAGGING CHANNEL: Il messaging channel è la componente basata su coda che consente lo scambio di messaggi tra i microservizi.

I microservizi nel contesto delle Smart City: quali vantaggi?

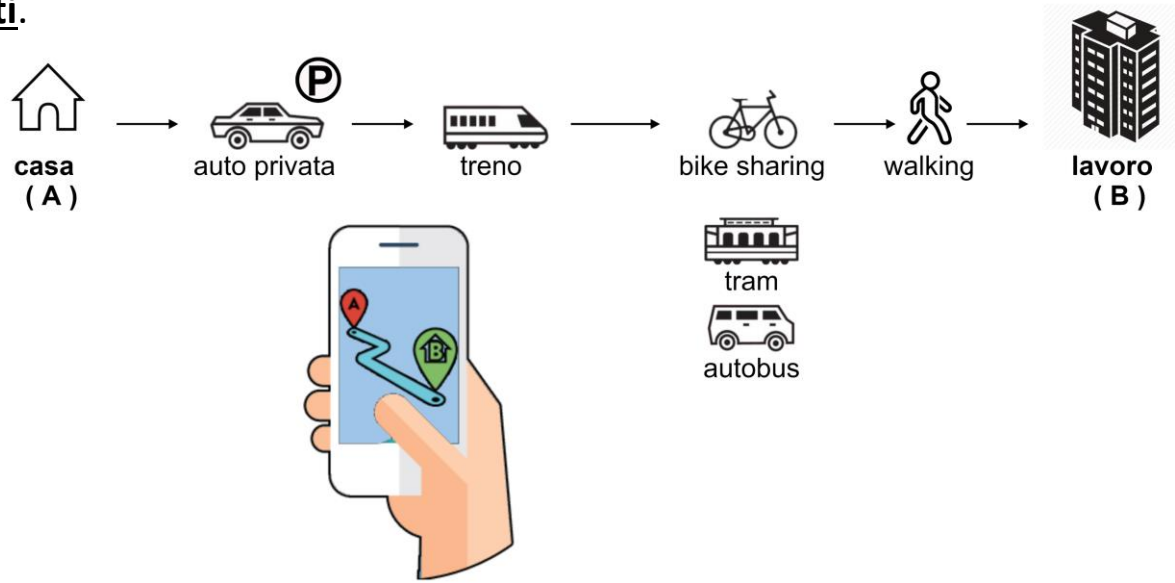
Sappiamo che il principio su cui si basano le Smart City sono la mole di da assorbire e l'integrazione delle diverse componenti che ne consentono il suo funzionamento. In questa prospettiva, l'adozione di un'architettura evoluta come quella a microservizi consente di ricoprire a pieno queste esigenze. Riprendendo i vantaggi dei microservizi, avremo:

- **Scalabilità e resilienza**: ogni componente può essere scalalata su diversi server e allo stesso tempo mantenere la resilienza rispetto agli altri
- **Dati**: la quantità di dati può essere distribuita tra i vari i microservizi, accessibile in ogni momento e con maggiore velocità. Oltretutto non si ha la necessità di legarsi ad un'unica soluzione software per lo storage.
- **Prestazioni**: la necessità di potenza di calcolo, di rete o altra risorsa, varia a seconda della funzione della componente che, in un'architettura a microservizi può essere facilmente scalata su più server.
- **Tecnology-free**: ogni applicativo, sensore o componente integrato nell'ecosistema della Smart City può adottare qualsiasi tipo di tecnologia
- **Sicurezza**: gli strati di sicurezza applicabili sono vari e particolarmente robusti. In questo modo la problematica può essere demandata direttamente ai moduli dedicati dell'architettura, svincolando così il singolo componente.

Il progetto MyWay2Go dell'aria R&D

MyWay2Go è una piattaforma tecnologica che affronta le tematiche dell'ambito **Smart Mobility** e che implementa un modello di tipo **MaaS (Mobility as a Service)** impiegando differenti servizi di mobilità offerti da differenti operatori.

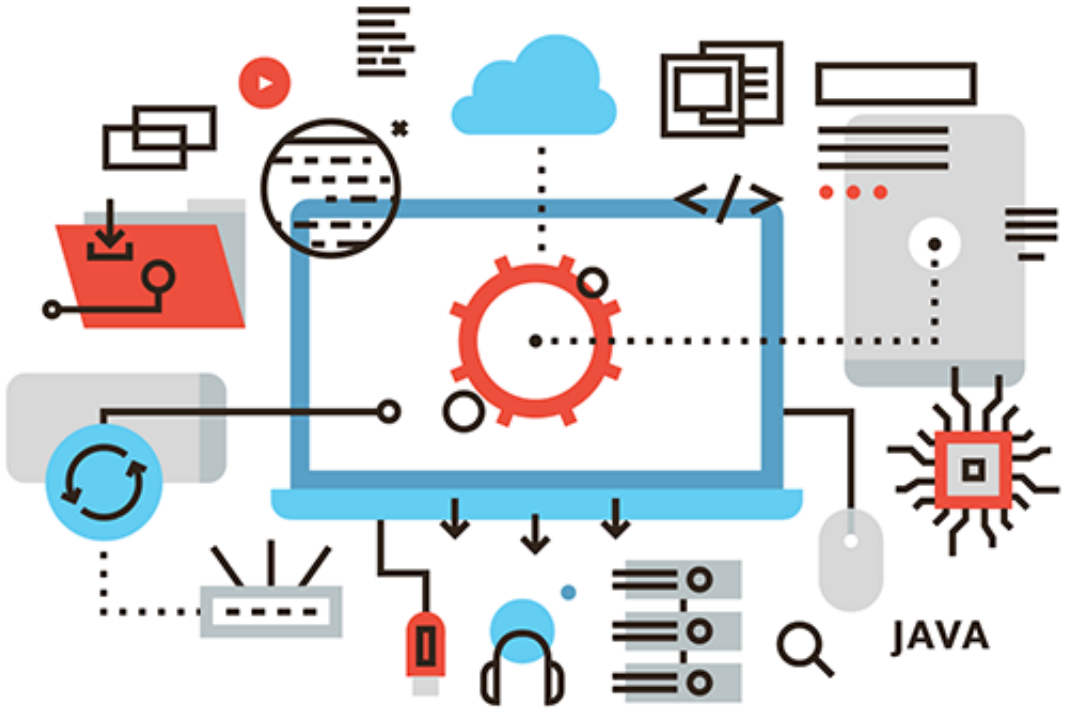
Il cittadino mediante l'utilizzo di **un'unica App mobile**, richiede alla piattaforma MyWay2Go di restituire un insieme di soluzioni intermodali per potersi spostare da un punto dipartenza ad un punto di destinazione. Il cittadino **fruisce dei servizi di mobilità che compongono il proprio itinerario e paga i titoli di viaggio corrispondenti** utilizzando sempre e solo l'applicazione mobile del sistema MyWay2Go. Tali dinamiche generano **movimenti finanziari** e la necessità di rendicontazioni tra le parti interessata, nonché la necessità per tutti gli attori coinvolti di avere la **certificazione degli eventi (transazioni) che generano tali movimenti**.



Architettura MyWay2Go

Le componenti di MyWay2Go:

- **Trt**Service
- **GTFS**Service
- **Credit**Service
- **GestioneServizio**Service
- **Parking**Service
- **Blockchain**Service



Le componenti MyWay2Go

- **GTFSService**: è la componente che si occupa della creazione, l'acquisizione e l'esportazione dei dati in formato GTFS.
- **TrtService**: raggruppa la configurazione relative al calcolo delle tariffe applicabili a viaggi e parcheggi. Per il calcolo dei costi, viene applica l'algoritmo di ottimizzazione Dijkstra.
- **GestioneServizioService**: riceve le richieste di percorsi effettuate tramite APP e controlla l'intero processo di esecuzione. Si interfaccia con OTP per ottenere i percorsi aggiornati in real-time.
- **ParkingService**: riceve le richieste di parcheggio dall'APP. Ottimizza i risultati in base alle preferenze specificate (distanza, costi, posizione).
- **CreditService**: è la componente di gestione del credito utente.
- **BlockchainService**: è lo strato intermedio che consente l'interconnessione con la piattaforma dedicata delle blockchain.

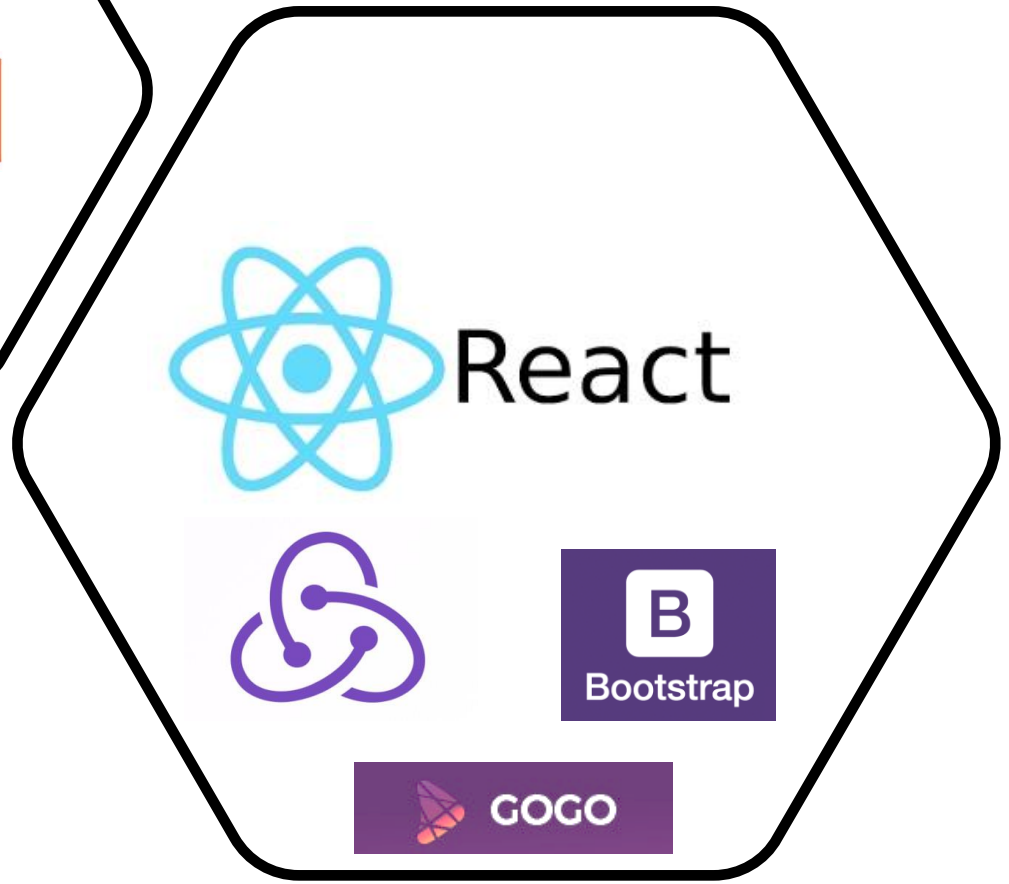
I Client di MyWay2Go

L’architettura a microservizi consente al sistema MyWay2Go di integrare delle soluzioni client indipendenti, ognuna con uno scopo preciso e con funzionalità dedicate.

Nello specifico, le applicazioni client che si collegano ai servizi delle piattaforma MyWay2Go sono:

- **MyWay2GoClient:** webapp per la gestione dei servizi offerti da MyWay2Go, dedicata al Maas ed ai Service Provider.
- **MyWay2GoApp:** applicazione mobile dedicata al Citizen per la fruizione dei servizi ed al Controllore per verificarne la validità in tempo reale.
- **CpAdmin:** webapp di gestione e controllo della piattaforma, abilitata all’amministratore di Sistema.

MyWay2GoClient



L'applicazione **MyWay2GoClient** consiste in una piattaforma di gestione dei servizi legati al settore del trasporto, siano essi erogati da soggetti pubblici o da privati, quindi: parcheggi, trasporti pubblici, bike sharing, noleggi con conducenti ecc.

La piattaforma viene utilizzata prevalentemente dal Maas e dai Service Provider, per la configurazione ed il controllo dei servizi.

Componenti esterne

La scalabilità dell'architettura ha consentito anche l'integrazione con altri servizi esterni che vanno a completare le funzionalità della piattaforma MyWay2Go:

- **OTP**: Open Trip Planner (OTP) è una piattaforma open source per la pianificazione dei viaggi e analisi multimodale della rete di trasporto.
- **GTFS-REALTIME**: tool per il controllo dei ritardi da parte dei mezzi pubblici, in grado di fornire in tempo reale le informazioni rispettando lo standard GTFS.
- **BIKE SHARING**: piattaforma dedicata ai servizi di Bike Sharing

I Benefici nel contesto aziendale

Tempistiche più rapide: Permette di gestire i cicli di sviluppo in un tempo più ridotto rispetto ai metodi tradizionali, grazie alla loro particolare struttura e all'agilità degli aggiornamenti;

Indipendenza a tutti gli effetti: In quanto componenti separati e autonomi, possono essere rimpiazzati, sviluppati e ridimensionati singolarmente;

Maggiore flessibilità e resilienza: Il difetto isolato non inficia il sistema. La sostituzione non ne modifica l'assetto e la funzionalità.

Tecnologia in evoluzione continua: Non esiste nessun vincolo allo sviluppo tecnologico a lungo termine. Con un approccio monolitico non sarebbe possibile, mentre un microservizio può essere rimodulato ad hoc.

Più intelligibilità, interventi mirati e gestione snella: Il team di riferimento è avvantaggiato dal più limitato raggio d'azione del servizio. Gli ambiti di intervento sono circoscritti e di più facile gestione.

Disegnati sulle potenzialità del business: I servizi non sono concepiti per un singolo settore o prodotto, ma sulle potenzialità globali di un'azienda.

Una risorsa riciclabile: Non essendo sviluppati per un prodotto o un progetto particolare ma sulle potenzialità produttive dell'azienda nel suo insieme, possono essere facilmente riutilizzati nel caso di variazioni nella strategia commerciale o anche nell'impostazione produttiva.

Gestione dati decentralizzata: Di solito, la gestione dati delle grandi imprese viene terziarizzata per una maggiore efficacia dell'elaborazione di ingenti quantità di dati. L'architettura a microservizi prevede la suddivisione in tanti singoli database.

Facile da sviluppare: Il margine di miglioramento diventa più facile da individuare e più immediato nello sviluppo. Si interviene solo ed esclusivamente laddove ce ne sia l'effettiva necessità;

Continua scalabilità: Più la domanda di servizi è alta, più i microservizi possono essere distribuiti su server e infrastrutture sulla base degli obiettivi aziendali.

Esperimenti e prospettive future

Il progetto di ricerca MyWay2Go ha consentito di sviluppare un'esperienza notevole per quanto riguarda l'architettura a microservizi. In particolare, vista la complessità delle sistema, la varietà dei moduli e la mole di dati da trattare ha dato l'opportunità a progettisti, analisti e sviluppatori di mettere in pratica le potenzialità dei microservizi, anche con esperimenti evoluti. Questa esperienza ci prospetta la possibilità di far diventare l'architettura una scelta prioritaria per i progetti futuri, con la possibilità di integrare ulteriori tipologie di tecnologie a seconda della necessità.

In base a questi presupposti, l'obiettivo è quello di portare a livello industriale la piattaforma MyWay2Go evolvendo ancora di più l'architettura a microservizi.

Conclusioni

La soluzione proposta ha permesso di sperimentare l'architettura sotto vari aspetti, permettendo così di fare un'esperienza notevole, tale da poter applicare le conoscenze acquisite su altri progetti futuri. In conclusione del progetto, si può affermare che una diversa scelta architetturale avrebbe fatto emergere problematiche difficili da affrontare e risolvere. Ad esempio, scegliendo una soluzione monolitica non si sarebbe potuto utilizzare uno stack tecnologico così ampio, con l'impossibilità di associare una tecnologia ad un singolo modulo. Altro aspetto importante riguarda la scalabilità e la resilienza dei microservizi che, in una piattaforma complessa come MyWay2Go, diventano indispensabili per un corretto funzionamento, e che, a contrario, sarebbero state affrontate singolarmente con un notevole dispendio di risorse.

Le varie modalità di comunicazione tra i microservizi inoltre, ci hanno dato la possibilità di sperimentare l'asincronismo delle operazioni che, essendo distribuite su diverse componenti, non per forza sono legate tra di loro. Su questo è stato molto importante l'ausilio della coda di messaggi.

In conclusione, si vuole sottolineare anche l'importanza delle fasi di *sviluppo*, *testing* e *deploy* del prototipo che, circoscrivendo le problematiche al singolo microservizio, non ci sono stati rallentamenti o attese tra i vari team. Il deploy automatizzato poi, ha permesso di eseguire continui aggiornamenti al sistema e allo stesso tempo avere sempre a disposizione una versione funzionante e aggiornata.