VOLUME 1

La Gazzetta del Cloud

WE MAKE IT RUN

IL NOSTRO OROSCOPO

SCOPRI IL TUO OROSCOPO CLOUD DEL MESE

▶ *Pagina 14.*

PAROLA ALL'ESPERTO.

DA PRODOTTO A SERVIZIO: I MICROSERVIZI COME BUILDING BLOCK PER FACILITARE LA SERVITIZZAZIONE

di **Alessio Gandini**

ABBIAMO UN SIMULATORE IN UFFICIO!

INCIDENTE SUL SIMULATORE ALLE PRIME LUCI DELL'ALBA

Dopo una folle corsa in pista, un dipendente perde la vite che fissa il sedile.

di **Anonimo Sistemista**

▶ ARTICOLO A PAGINA 9.

Intervista esclusiva al noto line manager

ALESSIO "GANDO" GANDINI

I beSharper sono tornati con un nuovo appuntamento per la meravigliosa rubrica di "***Meet the experts***".
Oggi siamo felici di presentarvi Alessio Gandini, per tutti "Gando", Cloud-native Development Line Manager. Coordina tutto il team di sviluppo di beSharp.
Una bella responsabilità, non credete?
Abbiamo chiesto a Gando di raccontarci un po' di lui, tra lavoro e passioni personali.

di **Anonimo Sistemista**

▶ ARTICOLO A PAGINA 12.

Tagli sul personale. Vietata definitivamente la carta in ufficio.

Ha dell'inverosimile la vicenda di cui è protagonista beSharp da alcuni giorni. Strani **tagli** sono comparsi sulle dita dei **DevOps.**

Secondo le prime ricostruzioni, alcuni beSharper avrebbero improvvisamente cominciato ad accusare **malesseri** e bruciori alle **dita**.

Ciascuno di loro è stato ascoltato, ma le loro testimonianze sembrano non aver ancora fatto luce sulla vicenda. Ascoltati anche alcuni testimoni; un cassiere del vicino supermercato, dichiaratosi **estraneo ai fatti**, ha riferito di aver riconosciuto uno dei beSharper addetti al pronto soccorso mentre acquistava dei cerotti.

Dopo alcuni ritrovamenti sospetti di fogli sulle scrivanie, gli esperti hanno ipotizzato un **utilizzo improprio di alcune fotocopie** richieste da uffici locali. beSharp, da sempre esperta di Cloud e attenta alla sostenibilità, avrebbe negli anni perso dimestichezza con l'utilizzo di questo arcaico mezzo.

"Purtroppo" - Dichiara Il Facility Hero - "alcune operazioni quotidiane (come questo giornale) non possono ancora essere totalmente digitalizzate. Farò il possibile perché questo avvenga quanto prima".

Dall'amministrazione esortano alla **cautela**. L'HR invita i beSharper a mantenere la calma e a seguire con attenzione le indicazioni sull'utilizzo di tutto ciò che non è digitale.

Prosegue intanto lo sciopero dei collaboratori iniziato ieri mattina a favore della tutela dell'ambiente e del **paper-free.**

Da prodotto a servizio: i microservizi come building block per facilitare la servitizzazione

Nel mondo dinamico del cloud computing, dove agilità e scalabilità sono fondamentali, il disaccoppiamento dei servizi rappresenta una pietra miliare per la creazione di sistemi resilienti e adattabili. In questo blog, abbiamo approfondito vari aspetti tecnologici della servitizzazione, esplorando strategie per il disaccoppiamento dei microservizi, modelli di comunicazione asincroni e architetture disaccoppiate che sfruttano event busses.

Questo articolo mira a sintetizzare la principali informazioni, fornendo una prospettiva completa sull'importanza del disaccoppiamento dei servizi e sulla sua relazione con la servitizzazione.

Senza ulteriori indugi, cominciamo il nostro viaggio!

Servitizzazione e architetture basate su microservizi

La servitizzazione implica la trasformazione dei prodotti in servizi, consentendo alle organizzazioni di offrire valore ai clienti attraverso esperienze più flessibili e personalizzate.

La **servitizzazione riguarda più il business che la tecnologia**; tuttavia, è strettamente legato ai microservizi come modello architettonico principale nelle architetture cloud.

Infatti, costruire un servizio di successo sta diventando sempre più impegnativo a causa delle aspettative molto elevate dei clienti e degli standard di mercato di conseguenza elevati.
Costruire un servizio di qualità richiede, oggi più che mai, tempo e impegno ed è spesso complesso, soprattutto se la user base è ampia e geograficamente distribuita.

È qui che i microservizi tornano utili.

L'essenza del paradigma dei microservizi risiede nella **scomposizione di applicazioni complesse in componenti più piccoli e autonomi** (microservizi) che operano in modo indipendente ma collaborativo per soddisfare varie funzioni aziendali.
Questo approccio migliora la modularità e la flessibilità del sistema e facilita il perfetto adattamento alle mutevoli richieste del mercato e alle esigenze dei clienti.

Uno dei principali vantaggi derivanti dall'adozione di un'architettura basata su microservizi è la sua flessibilità intrinseca. Suddividendo le applicazioni in servizi distinti e autonomi, diventa possibile modificare, aggiornare e scalare i singoli componenti senza interrompere l'erogazione del servizio.
Questa modularità consente una rapida innovazione e iterazione, consentendo ai team di rispondere rapidamente alle dinamiche del mercato e al feedback dei clienti.

La scalabilità è un altro vantaggio significativo dell'approccio ai microservizi. A differenza delle architetture monolitiche, in cui la scalabilità implica spesso la replica dell'intero stack applicativo, i microservizi consentono una scalabilità granulare che ottimizza l'utilizzo delle risorse e migliora le prestazioni e la reattività del sistema in condizioni di carico di lavoro imprevedibile.

Inoltre, l'isolamento dei guasti insito nelle architetture dei microservizi migliora la resilienza del sistema e la tolleranza agli errori. In un sistema monolitico, un guasto in un componente può ripercuotersi sull'intera applicazione, determinando tempi di inattività diffusi o interruzioni del servizio.

Al contrario, con i microservizi, i guasti dovrebbero essere contenuti all'interno dei singoli servizi, impedendo loro di interrompere completamente l'esperienza dell'utente e minimizzando l'impatto complessivo sulla disponibilità del sistema.

Le architetture a microservizi facilitano inoltre le pratiche di sviluppo agile e il continuous deployment. Poiché ciascun servizio opera in modo indipendente e potenzialmente gestito da un team dedicato, è più semplice adottare la metodologia agile e le pratiche DevOps per accelerare i cicli di sviluppo e semplificare i processi di rilascio. Questa agilità consente di seguire più prontamente i feedback degli utenti e quindi fornisce un vantaggio competitivo. In aggiunta, permette di accelerare l'innovazione riducendo o addirittura annullando il costo del fallimento e quindi permettendo sperimentazioni.

Inoltre, la natura modulare dei microservizi semplifica l'integrazione con servizi di terze parti e sistemi esterni. Il disaccoppiamento dei servizi e la definizione di interfacce chiare facilitano l'integrazione di nuove funzionalità con un impatto minimo sui sistemi esistenti, promuovendo l'innovazione e la collaborazione in tutto l'ecosistema.

Naturalmente, la creazione di un'applicazione basata su un'architettura a microservizi può anche presentare sfide rispetto alle tradizionali applicazioni monolitiche.

La sfida più grande è la sicuramente rappresentata dalla **maggiore complessità nella gestione di più servizi e delle loro interazioni**. A differenza delle applicazioni monolitiche, in cui tutti i componenti sono strettamente integrati, in una buona applicazione a microservizi questi sono loosely coupled, il che richiede un'attenta orchestrazione e coordinamento.

Vi è poi un aumento dell'effort di operation legato al maggior numero di parti funzionanti. Ogni componente viene generalmente eseguito nel proprio container/istanza/FaaS, determinando una proliferazione di risorse da gestire e aumentando la complessità operativa in termini di distribuzione e monitoraggio.

Inoltre, il debug e la risoluzione dei problemi possono essere più impegnativi in un ambiente di microservizi. Con più servizi che interagiscono, identificare e diagnosticare i problemi può essere complesso, soprattutto quando gli errori si propagano oltre i confini del singolo servizio.

Nonostante queste sfide, i vantaggi dei microservizi spesso superano le difficoltà, rendendo questo paradigma sempre più rilevante.

Decoupling

Il disaccoppiamento (decoupling) gioca un ruolo fondamentale in una buona applicazione basata su microservizi. Significa progettare componenti all'interno di un sistema affinché funzionino in modo indipendente, con una dipendenza minima o nulla dal funzionamento interno degli altri. Questo principio consente un migliore isolamento degli errori e maggior scalabilità granulare.

Nei sistemi sincroni, dove i componenti interagiscono in tempo reale, strumenti come i bilanciatori di carico svolgono un ruolo cruciale nel disaccoppiamento. I sistemi di bilanciamento del carico distribuiscono il traffico in entrata su più istanze/container di un servizio, consentendo la scalabilità orizzontale e garantendo che nessun singolo componente diventi un collo di bottiglia. Questo meccanismo di disaccoppiamento sincrono consente alle organizzazioni di gestire carichi di lavoro variabili senza compromettere le prestazioni o l'affidabilità.

D'altro canto, i sistemi asincroni si affidano alla comunicazione basata su messaggi per ottenere il disaccoppiamento. Code, bus di eventi e sistemi di notifica sono componenti critici nelle architetture asincrone, poiché facilitano l'accoppiamento lasco e l'isolamento dei guasti.

Le code fungono da intermediari tra produttori e consumatori, memorizzando nel buffer i messaggi e consentendo-

ne l'elaborazione successiva. Disaccoppiando i tempi di produzione e consumo dei messaggi, le code consentono un utilizzo più efficiente delle risorse e un funzionamento del sistema più fluido.

I bus degli eventi rappresentano la spina dorsale delle architetture distribuite e consentono una comunicazione continua tra i servizi senza accoppiamento diretto. Forniscono un modello di pubblicazione-sottoscrizione in cui i produttori pubblicano eventi su un bus centrale e i consumatori si iscrivono a specifici tipi di eventi di interesse. Questo modello di comunicazione disaccoppiato consente maggiore flessibilità ed estensibilità, poiché è possibile aggiungere o rimuovere nuovi servizi senza interrompere il sistema complessivo.

Allo stesso modo, i sistemi di notifica come Amazon SNS (Simple Notification Service) consentono la comunicazione asincrona tra i componenti, consentendo aggiornamenti e avvisi in tempo reale. Disaccoppiando il mittente e il destinatario dei messaggi, i sistemi di notifica consentono architetture più resilienti e reattive, in cui i componenti possono reagire dinamicamente ai cambiamenti ambientali.

Serverless

Nel panorama in continua evoluzione del cloud computing, le architetture serverless rappresentano un punto di svolta, offrendo un cambio di paradigma nella creazione, implementazione e gestione dei sistemi distribuiti. In prima linea in questa rivoluzione ci sono servizi come AWS Lambda e Fargate che forniscono potenza di calcolo alle applicazioni basate sui microservizi con affidabilità e scalabilità integrata.

AWS Lambda consente agli sviluppatori di eseguire codice senza effettuare il provisioning o gestire i server. Invece di preoccuparsi dell'infrastruttura, gli sviluppatori possono concentrarsi sulla scrittura del codice e sulla definizione degli eventi che ne attivano l'esecuzione. Questo approccio serverless offre scalabilità e flessibilità senza pari, poiché Lambda si ridimensiona automaticamente per gestire (quasi) qualsiasi carico di lavoro.

Tuttavia, i servizi serverless possono essere utilizzati anche per creare il livello di disaccoppiamento delle moderne applicazioni a microservizi.

Amazon SQS (Simple Queue Service), Amazon SNS (Simple Notification Service) e Amazon EventBridge sono fondamentali per il disaccoppiamento.

SQS fornisce un servizio di accodamento dei messaggi affidabile e scalabile, consentendo la comunicazione asincrona tra le diverse parti del sistema.

D'altro canto, SNS abilita la messaggistica pub/sub, consentendo ai componenti di comunicare in tempo reale senza accoppiamento diretto.

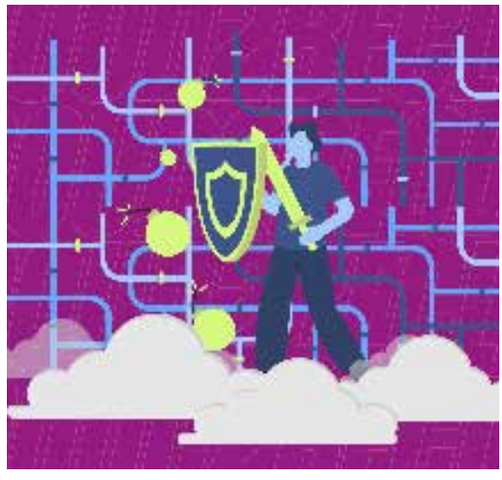
Amazon EventBridge funge da bus di eventi centrale, facilitando l'integrazione e l'orchestrazione di eventi tra diversi servizi e sistemi. EventBridge consente agli sviluppatori di definire regole e trigger per l'elaborazione degli eventi, consentendo una comunicazione e un coordinamento senza soluzione di continuità tra componenti disparati.

Oltre alla scalabilità e all'affidabilità, l'architettura serverless offre altri vantaggi, come facilitare cicli rapidi di sviluppo e CI/CD. Con Lambda, gli sviluppatori possono eseguire rapidamente l'iterazione de\l proprio codice, implementando le modifiche in pochi secondi anziché in giorni o settimane.

Questa agilità consente alle organizzazioni di rispondere rapidamente ai mutevoli requisiti aziendali e alle condizioni di mercato, offrendo loro un vantaggio competitivo nel frenetico panorama digitale di oggi.

Conclusioni

In uno scenario digitale che va sempre più nella direzione della servitizzazione, il successo di un business è sempre più influenzato dalle prestazioni dei servizi erogati, piuttosto che dalla vendita di prodotti. Fornire servizi digitali di alta qualità e rispettare gli standard e le aspettative dei clienti riguardo alla disponibilità, alla reattività e all'esperienza utente complessiva richiede soluzioni tecnologiche che consentano sviluppo rapido e resilienza del servizio.



L'adozione di tecniche di decoupling e di approcci asincroni, così come dei servizi serverless e delle tecnologie Cloud-native permette di raggiungere questo obiettivo.

Se da un lato dunque è indubbio il ruolo fondamentale dei microservizi in un percorso di servitizzazione di successo, è bene ricordare che non si può prescindere dalla capacità di controllare e gestire questo cambio di paradigma.

Speriamo che in nostro articolo abbia contribuito a chiarire le idee in questo senso.

di **Alessio Gandini**

FILIPPO ROSSI, IL NUMERO UNO DEL PADEL: “Giocare in beSharp è come progettare infrastrutture sul Cloud: una figata!”.

Lunedì 13 novembre si è svolta la finale dei Pavia Padel Open. L’evento ha visto sfidarsi post giornata lavorativa il campione Filippo Rossi con il compagno Mehmed Dourmouch e la coppia-rivelazione Alberto Casadei e Arnaldo Derosa.

6-4, 6-1 i due set che hanno consegnato nelle mani del team di Rossi il **terzo titolo consecutivo**.

“Giocare in beSharp è come progettare infrastrutture sul Cloud: Una figata!” ha affermato Rossi. “Ora si guarda al mondiale che si terrà a Las Vegas durante il re:Invent. Non saremo soli; **10 beSharpers verranno a supportarci**!

THE BESHARP BAND ON TOUR Audizioni aperte per un cantante.

La band Pavese si prepara a calcare i **palchi virtuali e non** delle principali fiere sul Cloud. In vista del tour 2024, i beSharps - in ritiro musicale nella sala-prove nella sede aziendale - hanno aperto le audizioni per un cantante aggiuntivo.

Via libera alle candidature.

Biglietti già disponibili su **TicketWAN**.

“IL MICROFONO NON FUNZIONA!“ Persi ieri i contatti con un beSharper in smart working.

Sono state ore di angoscia per la Business Unit di Cloud Infrastructure. Alle 9:30 circa di ieri mattina si sono persi i contatti con il beSharper M.G., 25 anni. Dopo svariati tentativi di configurazione, il giovane ha riferito **problemi al suo microfono** per poi non rispondere più ai ping dei suo colleghi.

Vani sono stati i tentativi di ricollegamento con la postazione in smart working di M.G. durati tutta la mattinata; mobilitate le Central Processing Unit della zona per rintracciare i suoi dispositivi. Realizzato anche uno **script in bash** in collaborazione con la Business Unit del generale Alessio Gandini.

Tutto si è fortunatamente risolto per il meglio quando, attorno all’ora di pranzo, il giovane si è presentato in ufficio. *“Sono uscito di casa per disperazione. Volevo solo tornare in sede”* ha dichiarato M.G.

Si è dunque trattato di allontanamento volontario. Il PC è stato comunque sottoposto a recovery per accertamenti.

“SONO USCITO DI CASA PER DISPERAZIONE. VOLEVO SOLO TORNARE IN SEDE”

*MG - **beSharper Traumatizzato.***

INCIDENTE SUL SIMULATORE ALLE PRIME LUCI DELL'ALBA.

Dopo una folle corsa in pista, un dipendente perde la vite che fissa il sedile.

Doveva essere una serata di divertimento come tante altre, invece si è trasformata in tragedia. È accaduto in beSharp ieri quando un beSharper, preso dall'entusiasmo per un **giro di pista record** grazie al sistema di **telemetria da lui stesso implementato**, ha esultato troppo vigorosamente sobbalzando sulla postazione e perdendo la vite di fissaggio del sedile.

Le ricerche sono ancora in corso: *"faremo il possibile per ripristinare il servizio nel minor tempo possibile"* ha dichiarato Nicola Ferrari, responsabile del servizio di supporto 24/7 di beSharp. *"Il sistema di ingestion e analisi dei dati in Cloud ci permetterà di fare luce su cause e responsabilità della vicenda"*.

Il Simulatore

Due anni fa, in un crescendo di smanettamento sempre più estremo, è stato assemblato il primo racing simulator by beSharp: volante **direct-drive da 32 Nm di coppia**, pedali con smorzatori idraulici progettati per resistere a oltre 140 kg di pressione, sedile racing con cinture a 6 punti e, soprattutto, una piattaforma di movimento a 4DOF per riprodurre la dinamica del veicolo e le asperità del tracciato.

"Un gran bel pezzo di hardware super-carrozzato equipaggiato con il pack *"Assetto Corsa Competizione"*.

Il team di beSharp è certo di tornare presto in pista.

L'ESTATE PIÙ CALDA DI SEMPRE. Il Mac del graphic designer dice basta.

Minuti concitati per l'azienda beSharp durante il salvataggio di alcuni asset critici. Dopo un'estate di **temperature ben sopra la media**, l'*Ente Aziendale per la Rilevazione delle Prestazioni delle Dotazioni Elettroniche* ha rilevato sul Mac del Graphic Designer la temperatura più alta rilevata negli ultimi 12 anni.

Il Mac è stato prontamente **sostituito** con uno di ultima generazione con le raccomandazioni del centro assistenza per il Graphic Designer di salvare almeno due volte al giorno, mangiare frutta e verdura e bere almeno due litri di acqua al giorno.

"Temevamo di averli persi" - ha dichiarato il designer- "invece abbiamo salvato tutti i dati".

La pronta e regolare sostituzione dei dispositivi è una pratica fondamentale, insieme alle dotazioni. Ciascun *beSharper* ha piena libertà di scegliere modello di Computer e strumenti di lavoro secondo le sue necessità e il proprio livello di *Nerditude.*

Microservizi, lambda e container: che differenza c'è? Parliamone!

Container, Lambda e microservizi sono concetti tanto popolari e importanti, quanto fraintesi dai tecnici nello sviluppo software moderno. A voltesi parla di architetture a microservizi quando, in realtà, si tratta solo di un gruppo di piccoli container con monoliti fortemente accoppiati. In questo articolo, faremo chiarezza sui microservizi, e sui servizi gestiti da AWS che ci faciliteranno nella loro implementazione.

"Usiamo i container, quindi il nostro sistema è a microservizi."

Sfortunatamente, le cose non sono così semplici: i microservizi devono aderire a regole rigorose e avere definizioni chiare. Entraimo nel vivo della discussione. I microservizi dovrebbero essere:

- **Strettamente definiti**: i microservizi devono implementare una singola funzione per la logica aziendale, come il logging, la registrazione dell'utente, il calcolo del prezzo.
- **Debolmente accoppiati**: un microservizio non deve dipendere da altri microservizi per implementare un'operazione, e la disponibilità di servizi esterni non deve influire sulla sua disponibilità.
- **Fortemente incapsulati**: la comunicazione tra microservizi deve essere standard, senza codice condiviso, e i dettagli sul funzionamento interno non devono essere accessibili, nemmeno per quanto riguarda la persistenza dei dati.
- **Indipendentemente distribuibili**: il rilascio di una nuova versione di un microservizio non deve influire sulla disponibilità e sull'implementazione di altri servizi.
- **Indipendentemente scalabili**: questo è una conseguenza dalle altre esigenze. Se un servizio ha bisogno di più risorse di calcolo, deve ottenerle senza dipendere da o influenzare altri servizi.

Vediamo ora i principi del design dei container:

- **Singola responsabilità**: un container si occupa di un singolo aspetto.
- **Alta osservabilità**: ogni container deve implementare API per osservare il suo stato e gestire l'applicazione.
- **Conformità al ciclo di vita**: un container dovrebbe essere in grado di leggere gli eventi riguardanti il ciclo di vita della piattaforma che lo esegue e reagire agli eventi.
- **Immagine immutabile**: un container è immutabile; il suo codice non deve essere specializzato e cambiare tra ambienti differenti.
- **Usa e getta**: un'applicazione containerizzata è effimera, quindi non devono essere memorizzati localmente dati e utilizzati per la persistenza dello stato
- **Auto-contenimento**: le librerie necessarie per l'applicazione sono disponibili all'interno del filesystem del container. L'unica dipendenza richiesta è il kernel.
- **Confinamento del runtime**: le risorse a disposizione del container sono definite e limitate. Il runtime applica i limiti per evitare che il sistema sia in sofferenza.
-

Come potete vedere, definizioni e concetti si sovrappongono in alcuni punti , ma il diavolo si nasconde nei dettagli: l'uso di una tecnologia specifica non significa che il design dell'architettura avrà automaticamente tutte le proprietà che ci aspettiamo (e non abbiamo nemmeno menzionato Lambda!).

I container, infatti, **offrono un modo per impacchettare in modo portabile ed eseguire il software in un ambiente isolato**. Allo stesso tempo, un approccio a microservizi aiuta a dividere un problema complesso in più servizi più semplici, fortemente disaccoppiati, che comunicano utilizzando API e possono essere gestiti da diversi team.

Benefici e sfide dell'uso di queste tecnologie

I container possono aiutare con la scalabilità, la portabilità e l'efficienza delle risorse, ma richiedono anche un'occhio attento a orchestrazione, monitoraggio e sicurezza.

Le lambda possono aiutare con l'agilità, la convenienza economica e la scalabilità, ma possono portare problemi di prestazioni e aumentare la complessità d'implementazione.

I microservizi possono aiutare con la modularità, la flessibilità e la tolleranza ai guasti, ma introducono anche latenza di rete, coerenza dei dati e aumentano la difficoltà nell'effettuare i test.

Ecco alcuni esempi di container e lambda che non possono essere identificati come microservizi.

Un WordPress containerizzato (non è strettamente defini-

to).
Una lambda che si basa interamente sull'output di un'altra lambda (non è fortemente disaccoppiata).
Due container che accedono alla stessa tabella del database (non c'è incapsulamento).
Ci sono molti esempi e sicuramente potete pensare a qualcosa nel vostro lavoro quotidiano.

Non ci concentreremo sul discutere se un'architettura orientata ai microservizi sia migliore e per quale caso d'uso perché miriamo a semplificare la nostra vita mentre gestiamo le infrastrutture che eseguiranno le nostre applicazioni.

Ricordate che quando usiamo i container e le lambda per implementare una strategia a microservizi, dobbiamo aggiungere almeno tre componenti alla nostra architettura: un runtime, un orchestrator e un discovery service.

Compute

Si tratta della parte più ovvia. Possiamo utilizzare EC2, Fargate (per ECS o EKS), Lambda. EC2 sembra controintuitivo, ma ricordiamo che i microservizi riguardano il design, non la tecnologia.

Lambda e Fargate offrono piattaforme di calcolo serverless, quindi la spesa si basa esclusivamente sull'uso. Sono la soluzione perfetta per un approccio a microservizi: se un microservizio basato su Lambda viene raramente invocato, il suo costo sarà marginale.

L'uso di Fargate e Lambda aiuta anche ad adattarsi ai modelli di utilizzo: la scalabilità orizzontale delle risorse orizzontali utilizzando piccole unità di calcolo ottimizza la scalabilità.

Orchestrazione

L'orchestrazione è un componente critico di un'architettura a microservizi perché aiuta ad automatizzare i rilasci, aggiunge scalabilità, consente il coordinamento e introduce la tolleranza ai guasti dei microservizi, gestendo anche il ciclo di vita del servizio e i flussi di esecuzione.

ECS e EKS (su EC2 e Fargate) offrono libertà di scelta per il mondo dei container e minimizzano l'effort di gestione; API Gateway e Step Functions aiutano con l'esecuzione e l'integrazione di Lambda e altri servizi AWS.

AWS AppMesh può anche essere utilizzato per gestire il routing e l'autorizzazione tra i servizi, aggiungendo un ulteriore strato di osservabilità.

Discovery Service

In un ambiente dinamico, utilizzare le risorse senza fare affidamento a nomi o indirizzi IP fissi è obbligatorio (questo requisito in realtà si applica a ogni ambiente cloud).

I microservizi dovrebbero anche essere in grado di notificare automaticamente la loro presenza a un registro centrale, che tiene traccia di cosa si trova dove e facilita la comunicazione.

Il discovery service mantiene un registro dei servizi disponibili e fornisce un meccanismo per gli utilizzatori per l'interrogazione e la scoperta, gestendo i cambiamenti dovuti alla scalabilità, ai guasti o agli aggiornamenti.

EKS utilizza il DNS interno integrato di Kubernetes, mentre ECS può fare affidamento su Amazon ECS Service Connect (questi due servizi possono essere utilizzati insieme).

Altri aspetti da tenere in considerazione

Per distribuire i microservizi in modo indipendente, avremo bisogno di pipeline. Inoltre sarà necessario impostare un sistema di logging efficiente per monitorare i container. Non dimentichiamoci poi dei componenti di comunicazione (come code, bilanciatori di carico e trigger di eventi) per disaccoppiare l'applicazione.

Questi componenti aumenteranno sicuramente la complessità dell'architettura, quindi è fondamentale sceglierli con attenzione: a volte insistiamo nell'uso di una tecnologia specifica, piegando le esigenze aziendali e il design della soluzione alla nostra fantasia o curiosità. Scegliere il design in base alla tecnologia è pericoloso e può portare un progetto a lfallimento. design sbagliato influisce sulla disponibilità, sulla gestibilità e, quindi, sul costo della soluzione.

Per Concludere

Come sempre, non esiste una soluzione perfetta: ci sono casi d'uso che si adattano perfettamente ai microservizi, mentre, a volte, suddividere ogni componente in servizi diversi può solo aumentare la complessità del progetto: c'è davvero bisogno di un servizio di registrazione utente per l'e-commerce di un negozio di animali di paese?

Iniziare con un monolite per accelerare il tempo di sviluppo e rilasciare il prodotto più velocemente potrebbe essere un approccio migliore rispetto al perdere il momento giusto per mettere sul mercato la soluzione.

di **Damiano Giorgi**

Meet the Experts

ALESSIO "GANDO" GANDINI

Ciao Gando! Come stai? Questa è una domanda che facciamo davvero a tutti. Qual è il primo ricordo che hai di beSharp?

La stanza al piano terra, il tavolo grande al centro e il caffè della Pixie.

In effetti, il primo ricordo che ho di beSharp è l'ambiente quasi famigliare. È più un concetto, che un ricordo di qualcosa di materiale. Se poi vogliamo scegliere qualcosa di materiale, allora c'è lui: Leonardo. Un vecchio PC che macinava task con i suoi 86°C costanti.

Raccontaci del tuo ruolo in azienda. Di cosa ti occupi? Quali sono i pro? Quali i contro?

Sono il responsabile del team di sviluppo applicazioni cloud-native e coordino il team data. Mi occupo della gestione dei progetti e del design ad alto livello delle soluzioni. Lo scopo del team è quello di progettare e realizzare applicazioni perfettamente integrate nel cloud di AWS, che sfruttano al 100% tutto il potenziale delle primitive a disposizione. I pro? Certamente la variabilità e la possibilità di restare sempre all'avanguardia nelle soluzioni proposte. I contro sono una maggiore difficoltà nella stima dell'effort, e le relative compilcazioni di management.

So che sei un appassionato di DIY. Quale è il progetto più folle che tu abbia mai affrontato?

Più folle non lo so, quello più curioso, utile e assurdo è un circuito per misurare il livello del liquido del radiatore di una vecchia automobile diesel. La spia in dotazione si accendeva solo quando già si vedeva il fumo dal cofano; così ho realizzato un sensore di livello costruito con due armature di rame da applicare all'esterno della tanica, di fatto un condensatore. Ho impiegato un oscillatore RC con il sensore di livello artigianale, ed usato il segnale come input per un timer/counter di un microcontrollore PIC al fine di determinare la soglia più adeguata ed avvisare il conducente prima che il livello sia critico, e quindi quando è ancora possibile rabboccare.

Cosa ci dici degli animali? Sappiamo che hai un gatto e hai avuto due.. oche..
Mi piacciono gli animali e li rispetto. Attualmente ho solo un gatto, ma ne ho avuti molti altri avendo vissuto in campagna da piccolo. Ho avuto anche una ventina di galline e due anatre vinte in fiera. Nessuno avrebbe scommesso nulla sulla loro sopravvivenza, invece sono diventate dei compagni veramente cari della mia infanzia. Ho poi adottato un corvo caduto dal nido, uno storno, una tartaruga e dei più comuni pappagalli.

Il tuo gioco da tavolo preferito?
Certamente Pandemic, anche se non disdegno generi completamente diversi, come ad esempio Talisman

Svelaci il tuo ideale di serata perfetta.
Sono un tipo molto classico per le serate. Cena di qualità e cinema. Serata che, tra l'altro, si adatta perfettamente sia da un'uscita con gli amici, sia ad una romantica.

Se dovessi consigliare un solo brano dalla tua playlist quale sarebbe?
Hotel California, degli Eagles

Quale è il tuo sogno nel cassetto che speri un giorno di poter realizzare?
Ho sempre sognato di produrre, realizzare e pubblicare un videogioco. Ho anche progetti abbastanza definiti, purtroppo attualmente mi mancano le risorse ed il tempo.

Il mondo sta per finire e tu, assieme alla tua personalissima squadra di cinque "Avengers" siete incaricati di salvare il mondo. Potendo scegliere tra qualunque personaggio, reale, di fantasia, di qualsiasi epoca e spazio, chi porteresti con te in questa missione?
Beh, se la sorte del mondo fosse in mano a 5 individui, a prescindere da chi essi siano, saremmo tutti davvero spacciati. Quindi sceglierò il team di cloud-native development di beSharp. Troveremmo sicuramente una soluzione… o capiremmo rapidamente che non c'è nulla da fare, in quel caso passeremmo i nostri ultimi giorni in birreria.

Se potessi reincarnarti in un personaggio storico o di fantasia di qualsiasi epoca, chi saresti? E perché?
Tony Stark. Perché è Iron Man, ha un sacco di soldi e tra le altre cose potrei anche produrre il mio videogioco. Poi ho già detto che è Iron Man?

Descrivi la tua vita con il nome di un film famoso!
Shutter Island

Per concludere, raccontaci la cosa più assurda che ti è mai successa!
Le cose più assurde che mi sono successe non si possono raccontare :D

"UNA RETE PER TROVARLI, UN IP PER NATTARLI E IN IPTABLES BANNARLI"

EC2 Classic
2006 - 2023

Il CEO di beSharp e i beSharpers tutti ne piangono la triste scomparsa con immenso dolore.

Ariete

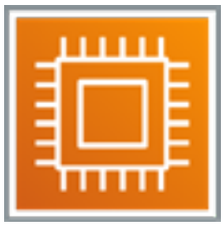
Saturno contro vi rende un po' **stanchi**. Scriptate il più possibile e affidatevi ai template!

Toro

Siete **up&running** e pieni di energie. Attenzione però al macho-programming: non esagerate.

Gemelli

Favoriti gli incontri per voi in questa settimana. È il momento giusto per il **Networking.**



Cancro

Prendetevi il tempo per apprezzare i piccoli dettagli. Le **ottimizzazioni** fatte vi faranno raggiungere performance senza pari.

Leone

Alti e bassi per voi del leone, ma rimarrete in un equilibrio stabile. Ora che **siete in Cloud**, i picchi non vi spaventano più!

Vergine

Il sole illumina i vostri progetti. Avete il potere di **deployare** qualsiasi cosa! Occhio allo spending però; non fatevi prendere troppo la mano.

Bilancia

Venere nel segno vi rende affascinanti, ma lo stesso non vale per le vostre applicazioni. È il momento di fare un po' di **code-review**!

Scorpione

Per voi questa settimana non sarà tutta Dev e Prod. Una **buona dose di test** sarà necessaria per uscire da qualche richiesta assurda dei clienti.

Sagittario

Con la Luna in aspetto dissonante, dovreste allontanarvi temporaneamente dall'ambiente **on-prem**. Avete bisogno di riposo!

Capricorno

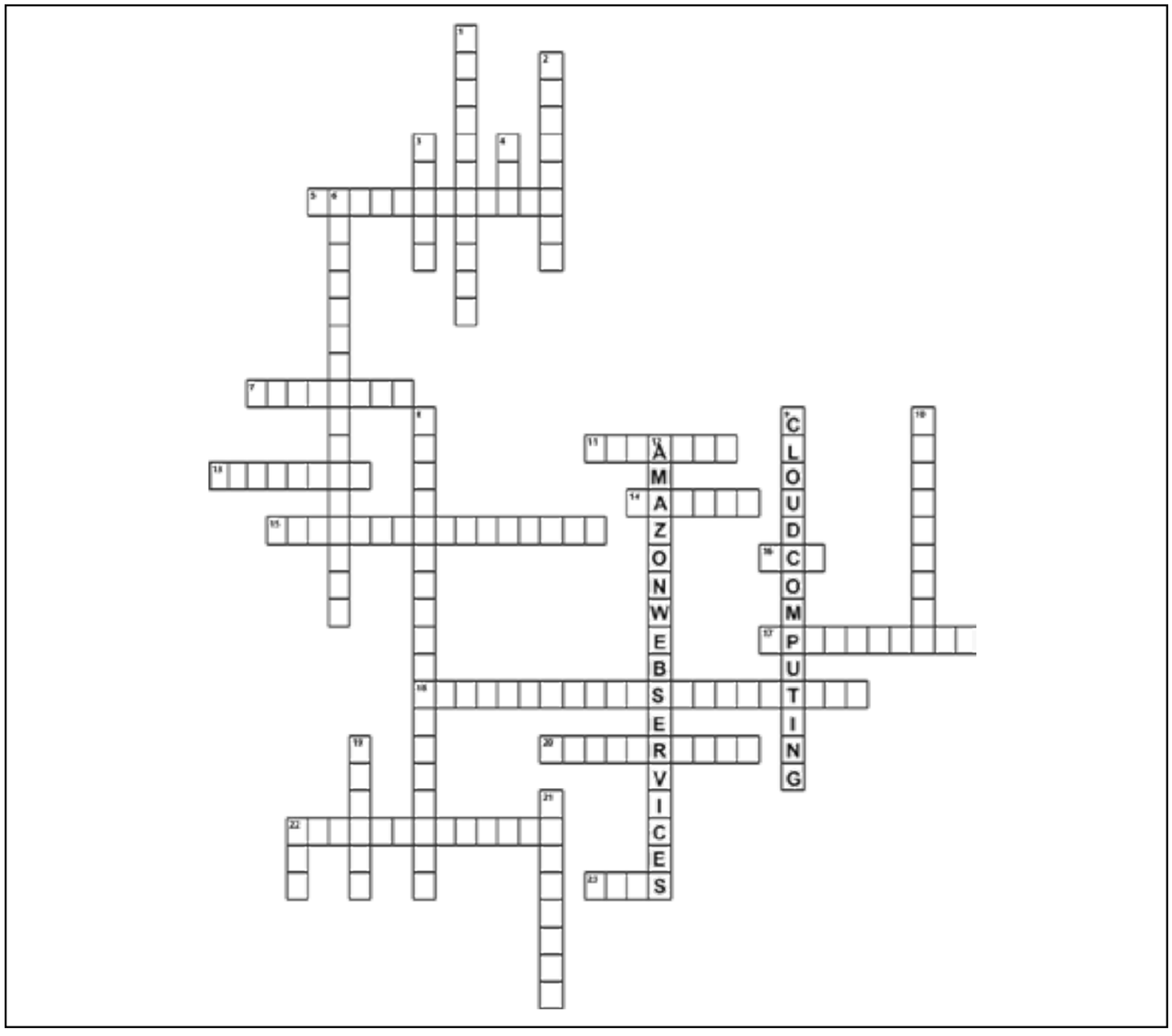
Sul lavoro siete ancora troppo diffidenti. **Uscite dai soliti server!**

Acquario

Avete dato i giusti consigli, ma non sono stati ascoltati? **Attendete con pazienza** la fine del deploy. Vi riscatterete.

Pesci

Avete una bella capacità di azione in questo periodo. È il momento giusto per una **nuova sfida**. Modernizzare monoliti non è mai stato così semplice!



Orizzontali:

5. Sistema di autenticazione che permette l'accesso a più applicazioni utilizzando un solo set di credenziali.
7. La parte di un sistema software che gestisce l'interazione con l'utente.
11. Server virtuale nel Cloud.
13. 1 TB equivale a 1000...?
14. Duplicazione di contenuti su supporti esterni al sistema principale finalizzato alla creazione di una copia di riserva.
15. Datacenter o raggruppamento di più datacenter all'interno di una AWS Region.
16. Information and Communication Technology.
17. Codice accessibile pubblicamente e modificabile.
18. Evoluzione correlata all'insieme di cambiamenti tecnologici e culturali associati all'applicazione di tecnologie digitali.
20. Paradigma Cloud che permette di eseguire codice senza dover gestire server.
22. L'insieme di pratiche e tecnologie volte al contrasto e alla prevenzione di attacchi informatici.
23. Attacco informatico basato sull'invio di un numero eccessivo di richieste al server finalizzato a renderlo indisponibile.

Verticali:

1. Capacità di una infrastruttura di aumentare o diminuire di scala e di adattarsi alle richieste in funzione del traffico.
2. Servizio AWS di object storage famoso per la sua "11 9s durability".
3. Assistente vocale di Amazon basato sul cloud che consente il funzionamento di Amazon Echo.
4. Rete per la distribuzione di contenuti web a bassa latenza (sigla).
6. Tecnologia che permette alle "cose" di interagire con altre "cose" sfruttando la connessione ad internet.
8. Metodologia di sviluppo software in cui sviluppo, testing e rilascio avvengono automaticamente tramite la definizione di Pipeline.
9. Paradigma di erogazione di servizi e risorse (compute, storage, database, networking,...) on-demand acquistabili secondo il modello di pricing pay-as-you-go.
10. Servizio AWS per la creazione e la distribuzione di modelli di Machine Learning.
12. Cloud provider mondiale con sede a Seattle. Adam Selipsky ne è il CEO.
19. AWS IAM: Identity and _____ Management.
21. Database NoSQL di AWS di tipo chiave-valore ad alte prestazioni.
22. Command Line Interface (Sigla).

ASCOLTA SU Spotify®

BESHARP RADIO. UNA PLAYLIST PER OGNI MOOD.

ASCOLTA SU Spotify®

PREZZO DEGLI ALIMENTARI ALLE STELLE. In beSharp ci pensa Planeat.eco

Aumenti record per i prezzi degli alimentari: secondo l'Istat, l'incremento medio è dell'11,2%. Il ministro delle Imprese lancia l'allarme: ***"Ora si deve intervenire"***

Inflazione, quanto è costata ai beSharper?

A finire sotto la lente di ingrandimento sono soprattutto i prezzi dei pranzi settimanali con un **prezzo medio** di 12€ a **schiscetta**. beSharp però non ci sta e aggiunge al **welfare aziendale** e ai **benefit** sugli alimentari già esistenti la possibilità di appoggiarsi a **Planeat**, un nuovo modo di fare la spesa a costo zero per i beSharper, sostenibile per l'ambiente, per l'economia del territorio e per la salute. *Patrizia beSharp*, addetta al controllo-qualità dei piatti, si dichiara estremamente soddisfatta.

"Piatti ben eseguiti, proprio come li cucinerei io!

Riso e Pollo
70% DI GRADIMENTO

Lasagna di mare
65% DI GRADIMENTO

Crostatina
81% DI GRADIMENTO

Carbonara
85% DI GRADIMENTO

planeat.eco

SCANSIONA IL QR CODE PER MAGGIORI INFORMAZIONI!

UOVA DI DRAGO SU MINECRAFT: C'È UNA NUOVA CONFERMA

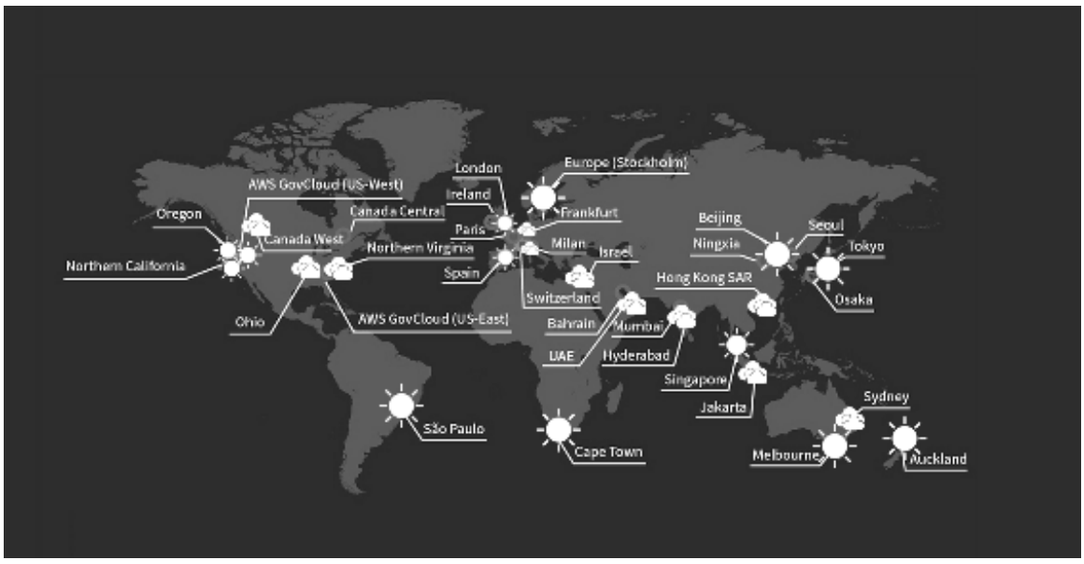
Grazie alla neo-nata collaborazione tra il team **Cloud Infrastructure** e il **team Cloud-native development** che aveva guidato la scoperta iniziale e con il supporto di alcuni ricercatori della ***Noovolari University***, è stato possibile condurre nuovi esperimenti e simulazioni che hanno dimostrato in modo definitivo la presenza di **uova di drago** su Minecraft.

Le prime scoperte

La prima scoperta di uova di drago sul server di Minecraft risale al **luglio 2015**, quando un team di beSharper individuò la presenza del raro oggetto. L'esistenza dell'uovo era poi stata messa in discussione da nuovi tentativi di ricerca, tutti improduttivi.

I beSharper non si sono arresi fino a quando lo scorso mese, aiutati dal Cloud, non hanno avuto la **certezza** dell'effettiva **esistenza** grazie a un nuovo avvistamento.

"Si tratta di un importante risultato scientifico raggiunto con i dati e con il grande lavoro svolto dal team di gaming di beSharp. Una scoperta di grande rilievo per il futuro dell'esplorazione di Minecraft", conclude **Daniele Papa**, responsabile dei videogames.



Il Meteo

BEL TEMPO SU QUASI TUTTE LE REGION AWS. NIENTE DISASTRI PREVISTI

beSharp non arresta la sua crescita. Nuovi talenti per unirsi alla band cercasi!

Siamo beSharper, e no, **non siamo i nerd a cui i film e gli stereotipi ti hanno abituato.**

Non ci importa da quale universo arrivi, se preferisci la **DC** alla **Marvel**, se il tuo sistema preferito è **Mac OS** o **Linux**, o se per te **Java** è meglio di **Python**. Quello che per noi è fondamentale è che tu abbia voglia, in un modo o nell'altro, come dice il nostro payoff, di far funzionare le cose, mettendo le mani in pasta in p**rogetti ad alto impatto tecnologico.**

beSharp (b#) come il ***Sì diesis***. Se sei un chitarrista pronto a pizzicare le corde giuste dei clienti, oppure un pianista che con la sua tastiera meccanica (e non solo) compone codice bello come uno spartito, **aspettiamo anche te**.

Niente bodyshaming in azienda, ma ti preferiamo senza peli sulla lingua: meglio un conflitto oggi che un *"merdone"* domani.

Se ti ritrovi in quello che abbiamo scritto, **c'è una felpa viola che ti aspetta.**

Cloud-native Back-End Developer

Il candidato ideale per questa posizione ha maturato almeno **3 anni di esperienza** come Back-end Developer in team di piccole o medie dimensioni. Il ruolo prevede l'interazione con i nostri clienti per la raccolta dei **requisiti funzionali** e la progettazione delle soluzioni tecniche necessarie a soddisfare le esigenze degli stessi.

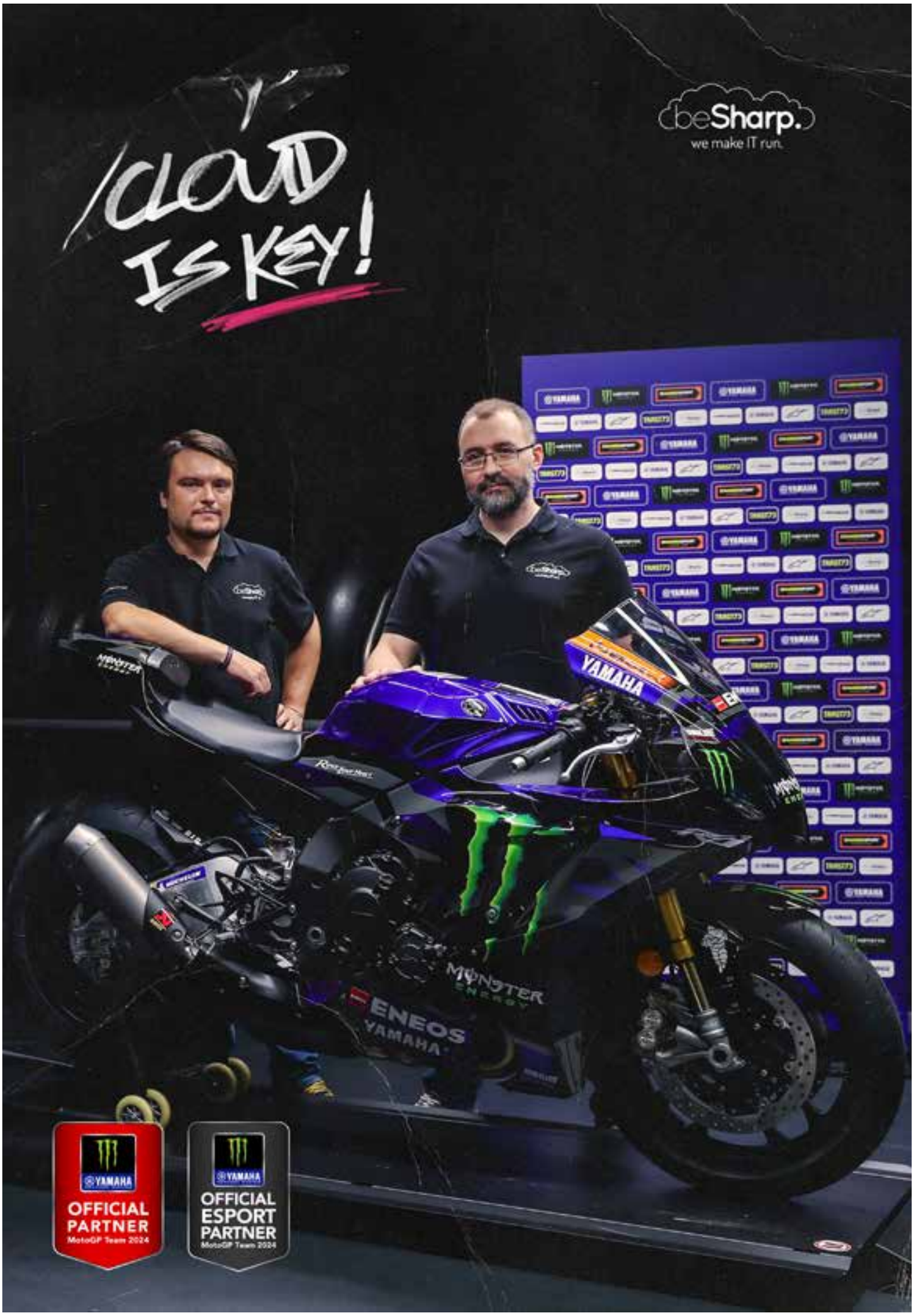
Senior Cloud-native Front-end Developer

Il candidato ideale per questa posizione ha maturato almeno **7 anni di esperienza** come Front-end Developer in team di piccole o medie dimensioni. Il ruolo prevede l'interazione con il cliente per la raccolta dei requisiti funzionali e la progettazione delle soluzioni tecniche necessarie a soddisfare le esigenze dello stesso. La persona selezionata diventerà inoltre il **riferimento tecnico** interno sullo stack tecnologico utilizzato, così da permettere al team di produrre soluzioni software resilienti ed aderenti alle best practice. Svilupperà e manterrà infine tutte le librerie ed i framework interni.

PENSI DI ESSERE LA PERSONA GIUSTA PER NOI?

SCOPRI DI PIÙ SU

www.besharp.it/careers


beSharp.
we make IT run.
CLOUD IS KEY!
YAMAHA
MONSTER ENERGY
ENEOS YAMAHA
OFFICIAL PARTNER
MotoGP Team 2024
OFFICIAL ESPORT PARTNER
MotoGP Team 2024