Ebook

Red Hat

Agile integration

Un modello per l'architettura enterprise

Sommario

"La modernizzazione dell'IT rappresenta da sempre l'aspetto della trasformazione digitale più critico (e talvolta frainteso), perché consente alle aziende di accelerare l'innovazione e migliorare l'efficienza. Un altro aspetto non meno importante è la creazione di una cultura digitale."[1]

Michael Bender e Paul Wilmot
Digital McKinsey

Il ruolo dell'integrazione nell'innovazione

La capacità di reagire alle innovazioni è sempre più spesso la chiave del successo aziendale. Quando aziende innovative fanno ingresso nei mercati e nuove tecnologie superano le aspettative dei consumatori, nasce la necessità di evolversi per far fronte ai cambiamenti più rapidamente. Per reagire in modo efficiente alle innovazioni ed essere vincenti nei propri mercati di appartenenza, le aziende devono adottare processi e architetture software all'avanguardia.

La tecnologia ha trasformato interi settori. Oggi, la maggior parte delle aziende offre servizi di ecommerce e i consumatori interagiscono regolarmente con esse in una realtà digitale. L'innovazione porta le organizzazioni a trasformare radicalmente i loro ambienti IT per poter fornire i nuovi servizi digitali che i clienti chiedono, in modo migliore e più rapido rispetto ai loro concorrenti.

L'agile integration rappresenta quindi la soluzione fondamentale da adottare per collegare le applicazioni e i servizi enterprise. L'agile integration unisce tre funzionalità architetturali efficienti per migliorare l'agilità, rendere più efficaci i nuovi processi e fornire un vantaggio competitivo: l'integrazione distribuita, le interfacce di programmazione delle applicazioni (API) e i container.

In alcuni settori, come quello turistico e dell'accoglienza, la trasformazione è stata innescata da modi nuovi di fare impresa: oggi vengono erogati nuovi servizi con i quali i consumatori interagiscono in modo differente. Il trend si sta propagando verso altri settori strategici, dai servizi finanziari alla pubblica amministrazione, ed è supportato da nuove tecnologie e approcci moderni all'interazione tra aziende e clienti. Le nuove sfide impongono alle organizzazioni presenti sul mercato una trasformazione radicale delle tecnologie, al fine di poter erogare nuovi servizi.

L'integrazione al centro del digital business

Fornire un'esperienza eccellente ai clienti non costituisce più un valore aggiunto, ma un requisito fondamentale per restare sul mercato. Se l'esperienza e le interazioni individuali costituiscono la base sulla quale costruire un rapporto di fiducia con i clienti, il loro percorso complessivo è l'elemento che garantisce stabilità.

Per fare un esempio, in un hotel il soggiorno ha la stessa importanza del processo di prenotazione, della conversazione che l'ospite ha avuto in merito a problemi con il Wi-Fi, del programma fedeltà e così via. Nell'economia digitale, i clienti sono sempre più esigenti e si aspettano interazioni personalizzate e adattate al contesto. Fondamentalmente, il rapporto che si instaura con il cliente dipende dall'esperienza che gli si offre.

Per far fronte alle aspettative dei clienti, oggi un'organizzazione deve semplificare la condivisione dei dati su tutte le applicazioni con cui i clienti interagiscono. Una strategia di integrazione efficace consentirà alle aziende di comprendere meglio le diverse esigenze dei clienti, anticipando le loro necessità e riducendo il rischio che si rivolgano alla concorrenza.

Uber rappresenta l'esempio perfetto. Mentre molti citano Uber come un esempio di innovazione digitale, viene spesso tralasciato l'impatto che l'integrazione digitale ha avuto sul suo successo. Le società che offrono servizi taxi hanno avuto accesso agli stessi dati dei clienti per almeno 20 anni, ma non erano in grado di collegare le applicazioni e utilizzare i dati in modo da prevedere e cambiare le aspettative dei clienti. L'agile integration ha colmato questa lacuna, offrendo pressoché infinite opportunità d'innovazione nel mondo digitale odierno e trasformando il modo in cui le società competono.

1 *Michael Bender e Paul Wilmott, Digital McKinsey, "Digital reinvention: Unlocking the 'how." Gennaio 2018,* https://www.mckinsey.com/~/media/McKinsey/Business%20Functions/McKinsey%20Digital/Our%20Insights/Digital%20Reinvention%20Unlocking%20the%20how/Digital-Reinvention_Unlocking-the-how.ashx

"Chi non riesce a superare i propri concorrenti in termini di time to market e agilità, ha perso in partenza. Le nuove funzionalità sono sempre una scommessa. Se si è fortunati, il 10% darà i risultati desiderati. Quindi, prima si riesce a immetterle sul mercato e a testarle, maggiore sarà il vantaggio competitivo. Anche l'azienda ottiene più rapidamente un ritorno sugli investimenti, iniziando a guadagnare prima."[2]

Gene Kim
The Phoenix Project

Oggi i leader di mercato più innovativi puntano sull'integrazione, perché sanno che è cruciale per la trasformazione digitale e, quindi, per il successo.

Agile integration: ottenere flessibilità e velocità

La velocità è fondamentale nel mondo digitale. Per mantenere la posizione di mercato, le aziende devono poter pianificare e applicare in tempi rapidi le modifiche ai sistemi software. Un'organizzazione capace di immettere rapidamente sul mercato nuovi prodotti a prezzi migliori si trova enormemente avvantaggiata rispetto a un'azienda che impiega tre mesi per l'implementazione e adotta una serie di fasi di verifica manuali.

Per adeguarsi ai ritmi accelerati di erogazione dei software richiesti dall'economia digitale, le organizzazioni non possono prescindere da un'infrastruttura agile. In questo caso, "agile" non indica lo sviluppo rapido del software, ma assume un significato più tradizionale: elastico, scattante, svelto e capace di muoversi con facilità.[3]

Finora, le metodologie agili si sono concentrate sullo sviluppo del software, puntando a migliorare e a ottimizzare la creazione delle applicazioni. Con l'approccio DevOps[4] si è cercato di integrare tale metodologia anche nel deployment delle applicazioni. Di per sé, l'approccio DevOps in genere non va oltre questo aspetto, occupandosi principalmente delle nuove applicazioni software sviluppate all'interno dell'azienda.

Con un'infrastruttura agile è possibile guardare oltre e creare un ambiente che accolga tutti i sistemi IT, incluso il software esistente. Un'infrastruttura agile coglie la complessità dei sistemi esistenti, dei diversi tipi di dati, dei loro flussi e delle aspettative dei clienti e cerca un modo per unificare il tutto.

Red Hat chiama questo processo "agile integration". L'integrazione non è una componente dell'infrastruttura, ma un approccio concettuale all'infrastruttura che include dati e applicazioni, hardware e piattaforme. Dall'allineamento di tecnologie e metodologie DevOps e di agile integration, ha origine una piattaforma che offre ai team la capacità di reagire ai cambiamenti assecondando i ritmi imposti dal mercato.

La fine della pianificazione: organizzazioni agili

Secondo Jim Whitehurst, CEO di Red Hat, "La pianificazione come la intendiamo non esiste più. In un ambiente poco conosciuto, pianificare è una prassi ormai inefficiente".[5] L'accelerazione e il cambiamento dei contesti aziendali scuotono ogni scenario e costringono a mutare rapidamente i piani. Rimanere vincolati a un piano d'azione può avere un costo elevato.

In questo senso, i piani di un'azienda hanno meno valore quando le informazioni sono scarse o l'ambiente non è sufficientemente stabile.

La scarsa conoscenza è un limite

Pianificare un'infrastruttura è un'attività a lungo termine, frequentemente estesa su più anni. I piani pluriennali, però, possono rivelarsi controproducenti e limitare la capacità di innovare o di adattarsi alle variazioni del mercato. Per "agilità" si intende la capacità di pianificare ed eseguire i piani più rapidamente. In questo ambiente, i piani hanno un'aspettativa di vita più breve, mentre vengono coltivati continuamente nuovi piani.

---

2 *Gene Kim, Kevin Behr e George Spafford. The Phoenix Project: A Novel about IT, DevOps, and Helping Your Business Win. Portland, Oregon: IT Revolution Press, 2013.*

3 *Dizionario Treccani*

4 *I vantaggi dell'approccio DevOps,* https://www.redhat.com/it/topics/devops

5 *Keynote di Jim Whitehurst al Red Hat Summit 2017.* https://www.youtube.com/watch?v=8MCbJmZQM9c

Quando i team sono abituati a cicli di sviluppo compresi tra i 6 e i 24 mesi, adattarsi a questi cambiamenti rapidi è complicato. Il problema è ancora più sentito nelle organizzazioni con strutture tradizionali in competizione con le start up, che affrontano il mercato in modi totalmente nuovi. Netflix, Blockbuster o Uber sono esempi lampanti, ma l'effetto rivoluzionario delle start up risale ai primi tempi dell'Era dell'informazione, con Amazon nel 1993 o i personal computer degli anni Ottanta.

Tabella 1: Principali rivoluzioni per settore

Settore	Servizio tradizionale	Innovazione	Effetti
Trasporti	Taxi, trasporto pubblico	Uber, Lyft	Creazione di un'esperienza cliente uniforme, impossibile da replicare per le piccole imprese locali
Gestione patrimoniale	Società di investimenti	Fondi automatici	Riallocazione dei differenziatori per la gestione dei fondi dal personale agli algoritmi
Vendita al dettaglio	Shopping fisico	Amazon	Cambiamento delle abitudini di acquisto, con il passaggio dall'acquisto in un esercizio fisico a quello online
Motori di ricerca	Google, ricerca tramite browser	Ricerca vocale	Ha effetti sul principale canale di Google verso il mercato e consente l'ingresso di nuove aziende

Start up e aziende innovative hanno il vantaggio di poter strutturare liberamente l'infrastruttura, i team, le applicazioni, l'architettura e perfino i processi di deployment. Non si tratta solo di avere idee innovative. Significa metterle in pratica, perché se non sono vincolate a un'infrastruttura esistente – o, per dirla con le parole di Rachel Laycock, "ai team esistenti"[6], possono essere davvero agili.

Oltre a ideare qualcosa di nuovo, queste organizzazioni realizzano anche sistemi innovativi pronti al cambiamento. L'infrastruttura software è parte integrante della loro capacità di differenziarsi: quasi ogni componente del sistema può essere sostituito, aggiornato o rimosso, per adattarsi alle nuove realtà del mercato. Tuttavia, con il passare del tempo, alcune start up risentono di una ridotta capacità di adattamento, mentre le migliori provvedono a far sì che la propria adattabilità sia protetta a tutti i costi.

6 *Rachel Laycock, "Continuous Delivery". Sessione pomeridiana generale, Red Hat Summit – DevNation 2016. 1° luglio 2016, San Francisco, California. https://youtube.com/watch?v=y87SUSOfgTY*

Le basi dell'agilità

Negli ambienti in rapida evoluzione, la chiave del successo è il funzionamento agile dell'intera infrastruttura IT.

Il cambiamento deve avvenire su due livelli:

1. A livello organizzativo e culturale con il supporto dei processi agili, dal design dell'architettura alla comunicazione tra i team.
2. A livello tecnico, con un'infrastruttura che consenta di aggiornare, aggiungere e rimuovere rapidamente funzionalità.

Di per sé, i cambiamenti tecnici e culturali non creano l'agilità, ma ne costituiscono le fondamenta.

Marty Cagan, Product Manager di eBay, applica una sorta di "tassa" a ogni progetto: parte del tempo e delle risorse impiegate in ogni progetto ordinario vengono destinate a nuovi progetti infrastrutturali.[7] Questo approccio dà priorità ai nuovi progetti e all'innovazione.

L'infrastruttura dell'agilità

Adottare una serie di nuove tecnologie spesso non aiuta a creare un'infrastruttura agile, in quanto porta i diversi team a muoversi in diverse direzioni alla ricerca di possibilità di miglioramento. Senza un insieme coerente di obiettivi a cui puntare, stabilire quali nuove capacità possano davvero rendere l'azienda più efficiente può diventare complicato.

I tre requisiti fondamentali dell'agile integration

L'approccio all'agile integration si fonda su tre tecnologie principali:

1. **Integrazione distribuita:** si tratta di poche decine di modelli di integrazione generali in grado di riflettere il lavoro e il flusso di dati enterprise. Nei container, questi modelli possono essere distribuiti nella misura e negli ambienti richiesti da applicazioni e team specifici. È il concetto di architettura di integrazione distribuita, diversa dalla tradizionale architettura di integrazione centralizzata, che consente ai singoli team di definire e distribuire in modo agile i modelli di integrazione necessari.
2. **API:** API stabili e ben gestite sono fondamentali per l'integrazione di team e pratiche devops. Le API raggruppano le risorse chiave in interfacce stabili e riutilizzabili, che funzionano come moduli da riutilizzare nell'organizzazione oppure con i partner e i clienti. Il deployment può essere effettuato insieme ai container in ambienti diversi, consentendo agli utenti di interagire con diversi insiemi di API.
3. **Container:** i container rappresentano la piattaforma di deployment di base per le API e le tecnologie di integrazione distribuita. Consentono di erogare un determinato servizio in un ambiente specifico, semplificandone lo sviluppo, il test e la manutenzione. Piattaforma principale degli ambienti DevOps e dei microservizi, i container usati per l'integrazione consentono una relazione trasparente e collaborativa tra i team di sviluppo e quelli di infrastruttura.

---

7 *Cagan, Marty, "Inspired: How to Create Products Customers Love". Wiley Press, 2017*

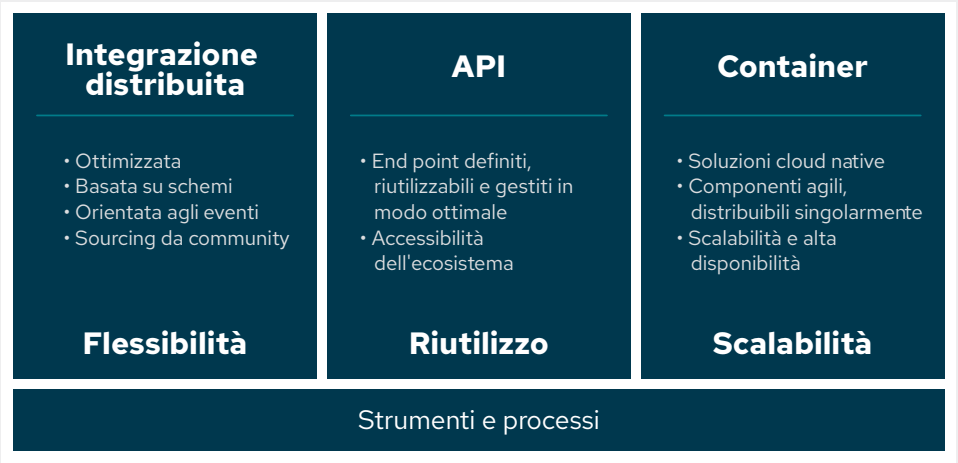

Figura 1. I tre requisiti fondamentali dell'agile integration

I tre requisiti dell'agile integration rendono l'infrastruttura IT più agile, poiché garantiscono un livello di astrazione elevato, al quale i diversi team possono collaborare. L'uso di una piattaforma per container con API e integrazioni distribuite consente di distinguere l'implementazione dell'integrazione dall'integrazione stessa. I team ottengono maggiore agilità perché sia le API che i modelli di integrazione distribuita possono contenere risorse chiave specifiche a un livello ampiamente comprensibile, senza che sia necessario comprendere o modificare l'architettura.

Ciascuna di queste tecnologie può fornire livelli di agilità importanti per specifiche sfide di integrazione. Nel loro insieme, avranno un effetto amplificato. È la cultura organizzativa a mettere in risalto la tecnologia: i vantaggi che essa offre si moltiplicano se combinati alle strategie DevOps, soprattutto alle procedure di automazione e deployment.

Requisito 1: integrazione distribuita

Una delle principali sfide dei sistemi IT odierni è data dall'esigenza di connettere applicazioni utilizzate da diversi team e al di fuori dell'organizzazione. In passato, questa necessità ha portato alla realizzazione di hub di integrazione centralizzati e molto complessi; spesso implementati come ESB (Enterprise Service Bus), hanno finito per diventare un ostacolo complesso e rigido al cambiamento rapido.

L'impiego di un ESB impone al team l'utilizzo degli strumenti dell'ESB per il suo intero ciclo di vita, in aggiunta a quelli già utilizzati negli ambienti di sviluppo e operativi. È una limitazione che complica le attività e le rende inefficienti e soggette a errori.

Con l'integrazione distribuita, si raggiungono molti degli obiettivi tecnici degli ESB di vecchia generazione, ma secondo modalità più adattabili ai vari team di un'organizzazione. Come gli ESB, la tecnologia di integrazione distribuita è dotata di capacità di trasformazione, routing, analisi, gestione degli errori e invio di notifiche.

"Nel software, quando qualcosa non funziona a dovere, la soluzione per uscire dall'impasse non è ignorare la situazione, ma affrontarla più spesso".[8]

David Farley
Continuous Delivery: Reliable Software Releases Through Build, Test, and Deployment Automation

La differenza sta nell'architettura. In un'architettura di integrazione distribuita, ogni punto di integrazione è considerato come un deployment unico e distinto e non come parte di un'applicazione di integrazione centralizzata di grandi dimensioni. L'integrazione può essere containerizzata e distribuita in locale per uno specifico progetto o team, senza che ciò influisca sulle altre integrazioni distribuite in azienda. Fondamentalmente, questo approccio considera l'integrazione come se fosse un microservizio[9] che accelera la velocità di sviluppo e il rilascio delle integrazioni.

Un approccio distribuito fornisce la massima flessibilità. Utilizza, inoltre, la stessa toolchain dei team agili o DevOps, sfruttando la piattaforma di base dei container e aumentando la capacità dei team di gestire le proprie integrazioni secondo strumenti e tempistiche proprie.

Strategico è anche l'allineamento con gli strumenti e i processi degli sviluppatori. L'integrazione distribuita non è un'infrastruttura software centralizzata, sviluppata e gestita da un gruppo di utenti specializzati e poi distribuita separatamente rispetto al processo di sviluppo del software. Quando l'architettura di integrazione viene distribuita mediante strumenti e piattaforme comuni, diventa accessibile a tutti gli sviluppatori a livello di progetto e agevola il deployment ottimizzato sempre e ovunque sia necessario.

Tabella 2. Confronto delle tecnologie di integrazione per ogni fase del ciclo di vita del software

Fase del ciclo di vita	ESB, gran parte delle iPaaS (Integration Platform-as-a-Service)	Supporto per tecnologie di integrazione distribuita
Controllo versione	Proprietario	Github e altri
Build	Proprietario	Maven e altri
Deployment	Proprietario	Container e altri strumenti DevOps
Gestione e scalabilità	Proprietario	Container e altri strumenti DevOps

Streaming su un backplane distribuito

La possibilità di adottare l'integrazione sincrona o asincrona, a seconda delle esigenze dell'applicazione, è una prerogativa dell'approccio che Red Hat definisce "agile integration". Un approccio comune di integrazione distribuita utilizza il metodo sincrono, eseguendo il deployment di container per condividere i dati tra i vari utenti, come detto in precedenza. Una seconda opzione di integrazione distribuita è lo streaming, un metodo asincrono, che consente agli sviluppatori di ricevere, rileggere e riprodurre eventi da altre sorgenti, a seconda delle necessità. Viene usata la messaggistica per replicare i dati in un archivio intermedio, così che i dati possano essere condivisi tra più team e le loro applicazioni. L'archivio di dati intermedio è vantaggioso per i team di microservizi, perché non sono costretti a cercare dati continuamente in modo sincrono da altre sorgenti.

8 *David Farley e Jez Humble, Continuous Delivery: Reliable Software Releases Through Build, Test, and Deployment Automation. Addison-Wesley Professional, 2010.*

9 *Vedere l'utilissima definizione di microservizi fornita Martin Fowler:* https://martinfowler.com/articles/microservices.html

Una piattaforma distribuita per lo streaming dei dati può pubblicare, sottoscrivere, archiviare ed elaborare flussi di archiviazione in tempo reale. È progettata per gestire flussi di dati provenienti da più sorgenti distribuendoli a più consumatori. Le piattaforme per lo streaming sono inoltre in grado di gestire milioni di punti dati al secondo, aspetto particolarmente vantaggioso negli scenari di utilizzo che richiedono una disponibilità dei dati in tempo reale, come operazioni IT e ecommerce. La figura che segue illustra un modello con più scenari di utilizzo di streaming. In seguito a una data operazione, ogni applicazione produce una modifica. Queste applicazioni possono propagare tali modifiche sotto forma di eventi a uno o più processori di streaming. A loro volta, tali processori applicano dei criteri di ricerca o di trasformazione e abbinano la situazione attuale ai dati storici spesso forniti dai sistemi di analisi.

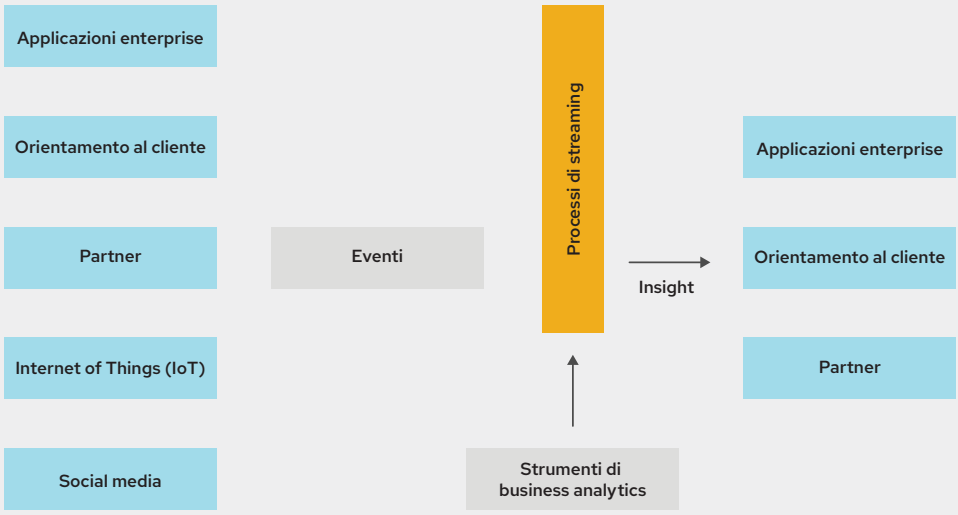

Figura 2. Flusso di eventi attraverso una piattaforma di elaborazione per l'analisi dei dati

In alcuni scenari di utilizzo, l'integrazione asincrona è più vantaggiosa rispetto a quella sincrona. L'integrazione sincrona presuppone la disponibilità di un ricevente perché la comunicazione possa avvenire correttamente, mentre l'integrazione asincrona basata sulla messaggistica consente a un messaggio di essere inviato a prescindere dalla disponibilità del ricevente in quel dato momento. Nella comunicazione sincrona, i timeout sono comuni, poiché l'applicazione deve aspettare una risposta per poter continuare a funzionare. Questo ritardo non avviene invece con la messaggistica asincrona, che continua l'elaborazione a prescindere dalla ricezione della risposta. Questa flessibilità rende l'integrazione asincrona, come lo streaming, ideale nell'ambito della gestione di grandi volumi di dati. L'integrazione asincrona fornisce inoltre disponibilità elevata e affidabilità, grazie ad un ridotto timeout del sistema.

L'utilizzo dell'integrazione asincrona guidata dagli eventi – come lo streaming – per incrementare integrazione sincrona e API, rafforza ulteriormente l'agile integration. La combinazione di integrazione e messaggistica migliora le prestazioni complessive dell'architettura di integrazione, grazie a un routing più efficiente, al supporto per più linguaggi e protocolli, a una maggiore produttività e a una migliore gestione dei dati.

Event mesh

L'event mesh rappresenta un'altra opzione asincrona di integrazione distribuita: un livello infrastrutturale dinamico e configurabile per la distribuzione di eventi su applicazioni disaccoppiate, dispositivi e cloud.

Una event mesh, composta da una rete di broker di eventi interconnessi, gestisce le interazioni asincrone basate sugli eventi. È indipendente dall'ambiente, quindi gli eventi di un'applicazione possono essere indirizzati ad altre applicazioni e ricevuti da altrettante applicazioni, senza dover configurare alcun routing e a prescindere dall'ambiente in cui è stato eseguito il deployment: on premise, cloud privato, pubblico, ibrido, Platform-as-a-Service (PaaS) o Internet of Things (IoT).

Inoltre, un'event mesh può collegare microservizi, servizi cloud native, applicazioni, dispositivi e database esistenti, fondamentalmente connettendo qualsiasi componente in grado di generare eventi verso qualsiasi utente. Un'event mesh è altamente efficace nella determinazione del percorso più veloce tra il produttore di un evento e il suo utente, per consentire interazioni in tempo reale.

Con l'evolversi degli ambienti IT distribuiti, l'event mesh si rivela una soluzione moderna, poiché rende la comunicazione flessibile, affidabile, veloce e più sicura.

Le interazioni asincrone e i modelli architetturali guidati dagli eventi non sono una novità, ma l'event mesh rappresenta un approccio all'avanguardia nel panorama dell'integrazione. Sebbene l'event mesh non venga ancora ampiamente adottata, Red Hat si aspetta che questa tecnologia guadagni popolarità nei prossimi anni, dato che la trasformazione digitale impone che le organizzazioni dipendano sempre più dalla flessibilità dell'integrazione basata sugli eventi.

Event mesh e service mesh non sono la stessa cosa. Mentre l'event mesh è asincrona, la service mesh supporta l'integrazione sincrona basata sulle API.

A livello di architettura, l'integrazione distribuita considera le integrazioni come microservizi. Possono infatti essere containerizzate, sono distribuibili in locale e in modo semplice e possono avere cicli di rilascio rapidi.

La tecnologia di integrazione deve poter supportare questo tipo di architettura ottimizzata e basata su microservizi. Con Red Hat® Fuse, gli utenti possono utilizzare le integrazioni come un codice eseguibile ovunque, anche in un container.

Fuse è offerto in bundle con Red Hat AMQ per erogare un'infrastruttura di messaggistica efficace in grado di garantire un efficace instradamento di dati ed eventi tra i sistemi. La messaggistica rappresenta un importante strumento per i microservizi, poiché la sua natura asincrona non richiede dipendenze.

Red Hat AMQ offre Apache Kafka, una piattaforma di streaming distribuito, tramite AMQ Streams su Red Hat OpenShift® Container Platform, la piattaforma Kubernetes utilizzata da quasi la metà delle aziende Fortune 100.[10] AMQ Streams è una funzionalità di data streaming dalle prestazioni elevate e altamente scalabile e distribuita, basata sul progetto Apache Kafka. Questa combinazione consente ai microservizi e ad altre applicazioni di condividere dati caratterizzati da produttività elevata e latenza ridotta.

Red Hat fornisce inoltre un'event mesh tramite AMQ Interconnect, un sistema globale peer to peer di erogazione eventi. AMQ Interconnect opera da gestore del traffico distribuito in grado di evitare tempi di fermo ed errori, al fine di raggiungere i consumatori in modo più semplice e affidabile. In questo modo, offre resilienza agli errori a livello di nodo e cloud.

Requisito 2: interfacce di programmazione delle applicazioni (API)

La maggior parte delle infrastrutture informatiche contiene centinaia, se non migliaia, di sistemi, applicazioni e risorse, che interagiscono con difficoltà, al punto che gli amministratori potrebbero non essere in grado di comprendere quali sistemi siano disponibili. Le API risolvono questa sfida fungendo da interfacce per tutte le risorse collegabili tramite tecnologie di integrazione.

A mano a mano che le organizzazioni passano da un approccio di integrazione centralizzato a uno distribuito, il self service diventa una priorità. Autorità e autonomia sono indispensabili per i team agili che devono ricercare, testare e usare i servizi sviluppati all'interno e all'esterno della propria azienda. Si tratta di caratteristiche che possono essere fornite con solide funzionalità API e che offrono ai team quell'integrazione flessibile di cui hanno bisogno mentre l'organizzazione continua a garantire la gestione e l'applicazione dei criteri di sicurezza, autorizzazione e utilizzo.

Le API forniscono le definizioni e le regole che stabiliscono il modo in cui le applicazioni comunicano tra di loro, fornendo agli sviluppatori un linguaggio comune per l'integrazione e un riferimento per progettare le integrazioni.

10 *Comunicato stampa di Red Hat, "More than 1,000 enterprises across the globe adopt Red Hat OpenShift Container Platform to power business applications" (Oltre 1.000 aziende al mondo adottano Red Hat OpenShift Container Platform per migliorare le loro applicazioni enterprise). 8 maggio 2019,* https://www.redhat.com/it/about/press-releases/more-1000-enterprises-across-globe-adopt-red-hat-openshift-container-platform-power-business-applications?extIdCarryOver=true&sc_cid=701f2000001OH74AAG

Gli sviluppatori mettono le API alla base dei loro progetti. Tuttavia, anche le API sono state concepite per essere condivise. È possibile mettere diverse API o sottoinsiemi di API a disposizione di destinatari diversi. Ad esempio, le necessità di un fornitore possono essere diverse rispetto a quelle dei team sviluppo interni o degli sviluppatori della community e le API appropriate possono così essere visualizzate da ogni gruppo, a seconda delle necessità.

Per utilizzare le API in modo corretto, l'organizzazione deve disporre di funzionalità di gestione delle API. La gestione delle API prevede la progettazione dell'API per l'applicazione e il gruppo di utenti, nonché la gestione dell'intero ciclo di vita dell'interfaccia. Poiché le interfacce di programmazione delle applicazioni sono spesso gestite come prodotti, ognuno affidato a un team diverso, emerge la necessità di garantire uniformità e semplicità d'uso delle diverse risorse.

Service mesh

Una service mesh migliora l'integrazione delle API fornendo un livello infrastrutturale configurabile per i microservizi, e rendendo la comunicazione flessibile, affidabile, veloce e gestibile.

L'uso dei microservizi prevede funzionalità di governance integrate che consentono di gestire la comunicazione service to service con un'interruzione minima delle operazioni. Tuttavia, aggiungere continuamente nuovi servizi, applicazioni e funzionalità sotto forma di microservizi, potrebbe aumentare la complessità operativa dell'architettura. Ogni nuovo servizio introduce un nuovo potenziale punto di errore. I servizi possono essere sovraccaricati dalle richieste. Centinaia di migliaia di microservizi che cercano di collegarsi costantemente l'uno con l'altro possono comportare latenze nelle comunicazioni e periodi di inattività delle applicazioni. Inoltre, in un'architettura a microservizi così complessa, diventa quasi impossibile identificare la causa di un problema. La service mesh è stata introdotta proprio per risolvere queste complicazioni.

Una service mesh è un livello infrastrutturale dedicato creato all'interno di un'app, che rimuove le logiche alla base della comunicazione service to service da servizi individuali e le astrae a livello di infrastruttura. Un sidecar proxy affianca un microservizio e indirizza le richieste ad altri proxy. L'insieme dei sidecar proxy forma una rete di mesh. In questo modo, una service mesh reindirizza le richieste da un servizio all'altro bilanciando il carico complessivo di tutti i microservizi.

Una service mesh consente quindi agli utenti di introdurre capacità aggiuntive ai microservizi, quali routing, tolleranza di errore e sicurezza, come anche visibilità, monitoraggio e test, senza modificare i componenti dei microservizi stessi. Questa funzionalità viene raggiunta dal sidecar proxy inserendo le informazioni e le funzionalità all'interno del microservizio.

La service mesh semplifica quindi la risoluzione degli errori di comunicazione perché la causa del problema non è nascosta all'interno dei microservizi, ma è visibile insieme ai servizi a livello di infrastruttura.

Inoltre, una service mesh rende le applicazioni più stabili e disponibili perché le richieste vengono reindirizzate escludendo i servizi che presentano errori.

Le service mesh raccolgono anche le metriche sulle prestazioni per fornire funzionalità diagnostiche maggiori in caso di errore di un servizio, riducendo in modo significativo i tempi di inattività. Gli stessi dati sulle prestazioni possono essere usati dagli sviluppatori per ottimizzare i rilasci futuri.

Senza una service mesh, ogni microservizio deve essere codificato mediante una logica che regoli la comunicazione service to service, ma questo impedisce agli sviluppatori di dedicarsi agli obiettivi aziendali. Eliminando questa codifica extra, la service mesh ottimizza il processo di sviluppo. Per questo, Red Hat si aspetta una rapida crescita dell'adozione di questa tecnologia nei prossimi anni.

Le API sono efficaci se possono essere utilizzate da sviluppatori interni e utenti esterni. Red Hat 3scale API Management offre strumenti per ogni tipo di utente. Fornisce un portale per sviluppatori che migliora la collaborazione ai fini di creare nuove API e un portale amministratori per pubblicarle. 3scale API Management aiuta a rendere disponibili le API agli utenti esterni, grazie alle funzioni di autenticazione, all'integrazione con i principali provider di servizi cloud e alla possibilità di esecuzione all'interno dei container.

Una strategia ottimale per le API prevede la progettazione dell'API e le modalità di distribuzione. 3scale API Management, soprattutto se implementata in una piattaforma per container, offre gli strumenti idonei per l'esecuzione di tale strategia.

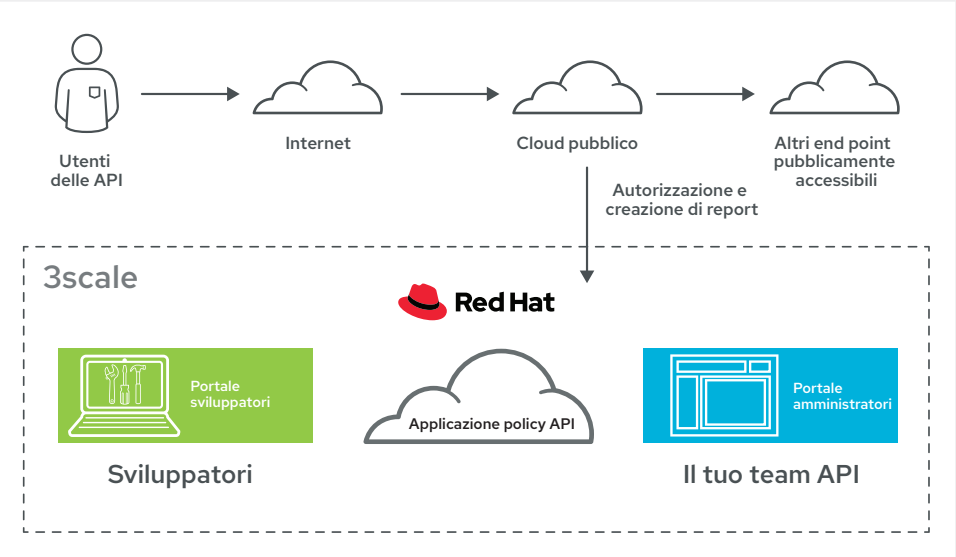

Figura 3. Panoramica su gestione delle API, end point e cloud pubblico

Red Hat offre inoltre Red Hat OpenShift service mesh, che unisce la diffusa service mesh Istio ad altri progetti chiave, come Jaeger per il tracciamento e Kiali per la visualizzazione, al fine di migliorare la gestione e la tracciabilità del deployment dei microservizi.

Inoltre, 3scale fornisce funzionalità complete di gestione delle API alla service mesh Istio in Kubernetes usando l'adattatore 3scale Istio.

"Implementare una soluzione multicloud comporta un'analisi accurata dei processi, delle competenze del personale e delle strutture organizzative, ma il settore concorda sul fatto che le piattaforme tecnologiche in grado di rendere il multicloud una realtà siano i container e Kubernetes".[11]

Suresh Vasudevan
Forbes

Requisito 3: container

Le tecnologie di virtualizzazione, cloud e i container hanno lo stesso compito: separano l'ambiente operativo software dall'hardware fisico, agevolando così la sovrapposizione di più istanze sull'hardware e rendendo più efficaci utilizzo, scalabilità e deployment. Tuttavia, affrontano la sfida in modi diversi. La virtualizzazione astrae il livello del sistema operativo; il cloud elimina il concetto di istanze del server distinte e permanenti, mentre i container definiscono l'ambiente operativo e le librerie essenziali per l'esecuzione di una singola applicazione.

È l'approccio regolato e ottimizzato che rende la tecnologia dei container lo strumento ideale per gli ambienti software moderni. Ogni istanza utilizza una definizione invariabile, dal sistema operativo alla versione esatta di ogni libreria. Per ogni istanza, quindi, l'ambiente è altamente ripetibile e coerente, idoneo anche ai flussi di integrazione continua/distribuzione continua (CI/CD). Ottimizzazione e ripetibilità fanno dei container la piattaforma tecnologica ideale per l'agile integration.

Poiché l'immagine del container definisce solo gli elementi necessari a una singola funzionalità, i container realizzano la vision dei microservizi e la loro orchestrazione è funzionale anche al deployment e alla gestione di infrastrutture a microservizi di grandi dimensioni.

I tradizionali approcci all'integrazione presentano una struttura fortemente centralizzata, con ESB posizionati nei punti focali dell'infrastruttura. L'integrazione distribuita e la gestione delle API hanno entrambe un'architettura decentralizzata, che distribuisce solo la funzionalità richiesta ove necessario, o a un team specifico. I container possono essere la piattaforma alla base di entrambi gli approcci, grazie alla loro naturale invariabilità, che assicura la coerenza delle immagini e dei deployment nei diversi ambienti: possono essere rapidamente distribuiti o sostituiti senza conflitti o dipendenze non trasparenti.

In un'architettura distribuita che si avvale di integrazioni o API, è fondamentale che la progettazione e la distribuzione di nuovi servizi non richiedano un processo di approvazione complesso.

I container consentono tanto alle API quanto alle integrazioni distribuite di essere utilizzate come microservizi. Forniscono strumenti comuni ai team di sviluppo e ai team operativi e consentono l'impiego di processi di sviluppo rapidi e procedure di rilascio gestite.

Ogni container rappresenta un singolo servizio o applicazione, così come un microservizio rappresenta una singola funzionalità distinta dalle altre. In un'architettura a microservizi possono coesistere decine o centinaia di servizi separati, che vengono duplicati negli ambienti di sviluppo, test e produzione, con un numero di istanze conseguentemente elevato. Affinché l'ambiente containerizzato sia efficace, la capacità di orchestrare e di eseguire attività di amministrazione avanzate è strategica.

11 *Vasudevan, Suresh, "Containers And Kubernetes Are Powering The Second Cloud Adoption Cycle", Forbes. 10 luglio 2019,* https://www.forbes.com/sites/forbestechcouncil/2019/07/10/containers-and-kubernetes-are-powering-the-second-cloud-adoption-cycle/#171ce5006929

Red Hat OpenShift unisce i container Linux® con il progetto di orchestrazione Kubernetes di Google. Prevede un'amministrazione centralizzata che include gestione delle istanze, monitoraggio, logging, gestione del traffico e automazione, attività non consentite da un ambiente di soli container.

Red Hat OpenShift include anche strumenti pensati per gli sviluppatori, come cataloghi self service, clustering delle istanze, persistenza delle applicazioni e isolamento a livello di progetto,

in una combinazione che soddisfa tanto i requisiti operativi, in particolare stabilità e test, quanto le esigenze degli sviluppatori in termini di facilità d'uso e rapidità di erogazione.

Implementazione dell'agile integration

Un lavoro di squadra

Se distribuite e rese disponibili ai team come capacità riutilizzabili, le tecnologie su cui si basa l'agile integration sono più efficaci. Con il termine "capacità" si intende tecnologie utilizzabili da gruppi di utenti autorizzati in modalità self service, per seguire con facilità le linee guida aziendali e accedere alle informazioni sulle migliori pratiche.

Spetta agli architetti o agli amministratori IT definire processi chiari per i singoli team, ad esempio:

- Fornire linee guida all'utilizzo ampiamente disponibili.
- Far applicare regole e pratiche di utilizzo quando necessario, ma concedere la libertà di sperimentare.
- Definire procedure idonee per ogni fase del ciclo di vita: prototipo, test, rilascio, aggiornamento e ritiro.
- Agevolare la condivisione delle informazioni per nuovi deployment e sviluppi.
- Consentire ai team dell'infrastruttura di abilitare e distribuire capacità self service, anziché coinvolgerli in ogni processo.

Un team dedicato al software deve, ad esempio, poter sviluppare, testare e preparare una nuova API per il rilascio self service, grazie a processi idonei ad aggiornare gli altri gruppi e a creare documentazione.

Prima della pubblicazione o dell'entrata in produzione possono essere attuate procedure o controlli incrociati con altri team, ma l'infrastruttura deve prevedere la massima automazione del processo.

Architettura dell'infrastruttura

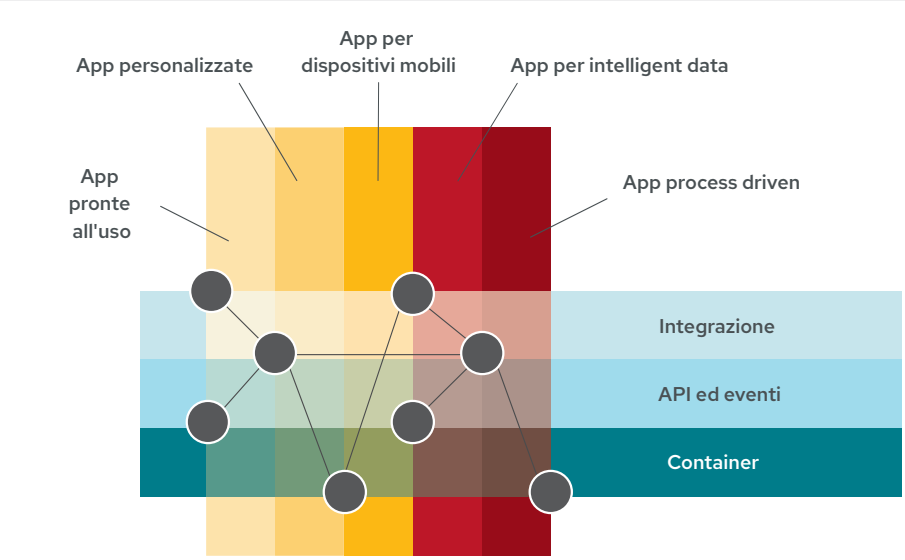

Figura 4. Intersezioni tra tecnologie di integrazione e stack applicativi

Un ecosistema software interno a un'organizzazione e, in molti casi, i punti di accesso alle integrazioni esterne, sono composti da due livelli, uno sincrono e uno asincrono. I container, le API e l'integrazione lavorano insieme a livello sincrono. Gli eventi, i container e l'integrazione lavorano insieme a livello asincrono. L'elaborazione di un evento include l'event mesh e le piattaforme di streaming distribuito.

Diversi tipi di sistema visualizzano diversi end point riutilizzabili, ciascuno mostrato come API riutilizzabile e molti eseguiti nei container, per garantire scalabilità e facilità di deployment. Le integrazioni offrono capacità di trasformazione, composizione e logiche di business assimilate, ovunque all'interno del sistema, integrando un gruppo di singoli servizi o raccogliendo i risultati dalle diverse unità organizzative.

È poi possibile aggregare ulteriormente le applicazioni integrate prima di destinarle all'utente finale.

Non è necessario che tutti i sistemi vengano destrutturati in componenti sempre più piccoli o che passino attraverso i molteplici livelli di astrazione delle API. Operazioni di questo tipo possono infatti ridurre l'efficienza, causare latenza o aggiungere complessità inutili. In alcuni ambiti, conservare la funzionalità ESB può essere la scelta giusta per mantenere le connessioni tra specifiche applicazioni. Occorre anche tener conto delle dipendenze tra sistemi distribuiti, utilizzando strumenti di monitoraggio e gestione idonei.

Per il sistema nel suo complesso, tuttavia, rimodulare l'architettura in termini di container, API e integrazione significa poter prendere le decisioni più giuste per ogni servizio, punto di integrazione e interazione con il cliente. Diventa possibile, ad esempio, verificare la sicurezza e reindirizzare direttamente al servizio di back end più appropriato un elevato volume di richieste in ingresso, evitando i cali di prestazioni dovuti all'utilizzo di un singolo ESB.

Negli ambienti cloud ibridi e distribuiti, molti dei sistemi back end in questione possono occupare posizioni fisiche diverse. In questi casi, si ottiene più efficienza e sicurezza integrando sistemi locali, anziché eseguendo il routing di ogni elemento attraverso un unico sistema di integrazione centralizzato che detiene la logica di business strategica.

Organizzazioni e cultura agile

Il ciclo di vita dell'infrastruttura è diverso da quello operativo o dello sviluppo del software. Il ciclo dello sviluppo del software prevede il completamento di un progetto prima di passare a quello successivo. In questo caso si è efficienti se si riducono i tempi di rilascio o se aumenta il numero di funzionalità prodotte in un tempo specificato. Per i team operativi, che si occupano di manutenzione e stabilità, è vantaggioso poter applicare correzioni e aggiornamenti di sicurezza, distribuire nuovi servizi o annullare modifiche in modo rapido ed efficiente.

Nel caso dell'infrastruttura, l'approccio è differente, essendo quest'ultima oggetto di un lavoro sul lungo periodo, con team eterogenei e altamente specializzati che operano in modo diverso dai team interfunzionali impegnati in progetti di software engineering specifici.

In linea di massima, i progetti di infrastruttura hanno un respiro più vasto rispetto ai progetti software. Pertanto, l'applicazione di cicli di rilascio brevi come quelli del software potrebbe portare a un mancato conseguimento degli obiettivi o a progetti incompleti. Su InformationWeek, Andrew Froehlich, professionista dell'IT enterprise, ha affermato che l'infrastruttura presenta un punto di non ritorno, soprattutto per quel che riguarda hardware e datacenter, ma anche per il cloud pubblico: superato questo punto non è più possibile annullare un progetto e cominciare da capo.[12] L'infrastruttura è permanente. Ciò non toglie che sia possibile conciliare metodologie e prestazioni dell'infrastruttura.

Evidenti per i team di sviluppo e operativi, i vantaggi dei processi reattivi e iterativi come quelli agili e DevOps lo sono meno per i team di infrastruttura. Tuttavia, è importante che i team dell'infrastruttura siano allineati con i team DevOps, per ottenere la massima efficienza. La società di consulenza globale McKinsey ha dichiarato: "Usare una trasformazione agile per rinnovare un'infrastruttura IT non è facile, ma ne vale la pena. Secondo la nostra esperienza, gli approcci agili possono consentire ai gruppi delle infrastrutture IT di aumentare la loro produttività del 25-30% in 6-18 mesi, a seconda della dimensione dell'organizzazione".[13]

Le tecnologie di agile integration sostengono un'infrastruttura più agile. API, immagini dei container e integrazioni diventano oggetto di attenzione quando si parla di infrastruttura software.

---

12 *Andrew Froehlich, "Should IT go agile? The pros and cons". 6 ottobre 2015,* http://www.informationweek.com/infrastructure/pc-and-servers/should-it-go-agile-the-pros-and-cons/d/d-id/1322448

13 *Cormella-Dorda, Santiago, et al. "Transforming IT infrastructure organizations using agile." McKinsey Digital, ottobre 2018,* https://www.mckinsey.com/business-functions/mckinsey-digital/our-insights/transforming-it-infrastructure-organizations-using-agile

Il Manifesto Agile definisce quattro principi chiave per lo sviluppo software.[14] In un'infrastruttura agile basata sull'integrazione, questi principi possono essere adottati nella strategia di integrazione.

1	**Individui e interazioni hanno la priorità su processi e strumenti.** Nell'ambito dell'infrastruttura, le interazioni tra i team assumono la forma di comunicazioni dirette - gestite da API, messaggistica e schemi di traffico, interdipendenze a livello di sistema, processi di test e rilascio, come nel caso dei flussi CI/CD.
2	**Un software efficiente prevale su una documentazione esaustiva.** Per sua natura, l'infrastruttura deve essere funzionale 24 ore su 24, 7 giorni su 7, con un adattamento graduale da preferire a cambiamenti repentini. Pertanto, l'efficienza è sempre un requisito implicito. Una strategia dell'infrastruttura è efficiente quando le prestazioni soddisfano le aspettative dell'utente finale.
3	**La collaborazione con il cliente prevale sulla negoziazione dei contratti.** Nei sistemi infrastrutturali, i contratti indicano ai team di infrastruttura come gestire le dipendenze di sistema, ovvero i criteri di sicurezza, gli accordi a livello di servizio e le API pubblicate. I clienti sono gli utenti, interni ed esterni, di tali sistemi. L'agilità consente loro di avere voce in capitolo e proporre eventuali modifiche da apportare ai criteri e alle interfacce associate ai sistemi e permette di applicarle in tempi rapidi. L'adozione delle integrazioni distribuite consente di ampliare la collaborazione offrendo direttamente ai team l'autonomia e il controllo necessari per sviluppare e distribuire le integrazioni.
4	**Rispondere al cambiamento ha la priorità sull'esecuzione di un piano.** Questo principio prevede che la tecnologia supporti i processi. I sistemi dell'infrastruttura devono rimanere stabili ma le tecnologie più recenti, come i container, offrono una piattaforma elastica, nella quale aggiungere e rimuovere dinamicamente le istanze in funzione delle richieste, automatizzare deployment e aggiornamenti, orchestrare i cambiamenti su più istanze. Le definizioni delle API pubblicate diventano strumenti riutilizzabili che rendono più coerenti le attività di sviluppo. Grazie a questo approccio, anche una piattaforma progettata per adattarsi ai cambiamenti acquisisce stabilità.

Figura 5. I principi chiave per lo sviluppo software secondo il Manifesto Agile

Nell'agile integration, la tecnologia sostiene il cambiamento culturale dei team di infrastruttura, funge da base per la strategia dell'infrastruttura, e allinea i team e le tecnologie dell'infrastruttura alle strategie aziendali e di sviluppo.

La metodologia agile identifica alcuni componenti chiave di un progetto software, come individui, build e dipendenze e quindi definisce le relazioni tra questi elementi. Se si interpreta l'infrastruttura di integrazione come un progetto agile, si possono identificare elementi e relazioni simili e paralleli: team, immagini di container, API e punti di integrazione. La Tabella 3 descrive alcuni di questi parallelismi.

14 *Agile Manifesto,* http://agilemanifesto.org/

Tabella 3. Confronto tra elementi del software agile e dell'infrastruttura agile

Progetto	Organizzazione	Dettagli
Individui	Team	I team sono responsabili di alcuni specifici aspetti dell'infrastruttura. Identificano le informazioni relative alle responsabilità dei team, ad esempio i sistemi o le API gestite dal team, i leader e gli obiettivi del team.
Moduli	API	Quando sono ben definite, le interfacce API sono stabili nel tempo, hanno roadmap proprie, sono eseguite da team specifici e forniscono una capacità specifica che offre valore all'organizzazione.
Build	Immagini dei container	I rilasci si basano su unità distribuibili testate e contrassegnate, il cui deployment può essere eseguito in modo affidabile da qualsiasi team autorizzato. Sostituiscono il codice monolitico basato sulle versioni.
Compilazione delle dipendenze	Integrazioni	Questo elemento identifica le integrazioni e le associazioni tra i diversi componenti dei sistemi distribuiti. I punti di integrazione possono essere gestiti, ordinati, ritirati, sottoposti a versioni e testati, come qualsiasi altro elemento del sistema.
Test delle build	Automazione dell'infrastruttura	Questo elemento include la gestione dell'intero ciclo: la capacità di testare le build software, le prestazioni e i requisiti degli utenti, gli aspetti operativi e il monitoraggio di più sistemi.

Applicazione dei principi Agile alla pianificazione dell'infrastruttura

La maggior parte degli approcci alla gestione dei cambiamenti esige una documentazione completa di tutti i sottosistemi. Tale documentazione deve illustrare dettagliatamente ogni aspetto del sistema, dal metodo di monitoraggio ai parametri delle prestazioni e specificare inoltre i team responsabili. L'adozione dei principi Agile richiede collaborazione e adattabilità, al contrario di una gestione del cambiamento basata sulla documentazione.

Invece di definire in modo prescrittivo tutte le potenziali parti coinvolte, le modifiche e i componenti di sistema, è più semplice definire un insieme di linee guida e di standard da applicare per valutare le richieste di modifica e la pianificazione. È necessario considerare i seguenti aspetti:

- Qual è l'esperienza complessiva che si desidera erogare all'utente?
- In che modo ogni elemento coinvolto - team, API, sistema - contribuisce a migliorare tale esperienza nel tempo?
- In che modo verranno definite le attività di monitoraggio e avviso necessarie per mantenere i livelli di servizio? Per quali parametri?
- Quali test automatici sono necessari per verificare il comportamento previsto?
- Qual è il flusso di rilascio che adottano i team per testare e distribuire le nuove versioni dei propri sottosistemi senza interferire con l'esperienza dell'utente?
- In che modo l'errore di un componente influisce sui livelli di servizio dell'intero sistema?

Nell'ambito dell'infrastruttura agile, la gestione dei cambiamenti va intesa più come una collaborazione continua che come un contratto.

Pronostici

Quante probabilità di riuscita ha il tuo progetto? Innanzitutto, occorre conoscere i criteri che definiscono la buona riuscita di un progetto: la soddisfazione delle specifiche, l'adozione da parte dei clienti o il rilascio. Una ricerca condotta da un istituto di Project Management ha rivelato che, oggi, un numero maggiore di progetti raggiunge gli obiettivi prefissati rispetto agli scorsi 5 anni. La ricerca attribuisce questo miglioramento all'allineamento sempre più forte tra IT e team aziendali, da cui deriva una migliore condivisione di informazioni su strategia ed esigenze dei clienti.[15]

Una delle ragioni alla base di un allineamento strategico è la creazione di team agili. La metodologia agile incoraggia di fatto la collaborazione e il feedback, una visione globale dei problemi e dei sistemi e approcci più creativi.

Disporre di uno stack tecnologico condiviso sposta l'attenzione dall'indipendenza del codice ai sistemi e alle rispettive interdipendenze. Significa pensare a livello di sistema, considerando tutti gli elementi dell'infrastruttura software, incluso il software sviluppato internamente, i sistemi dei fornitori e le loro connessioni, come un sistema unico. Le API e i sistemi di messaggistica possono coprire l'intera infrastruttura e cooperare per unificare i sistemi software.

La possibilità di sviluppare e comprendere le API e le integrazioni distribuite nei singoli team operativi o di sviluppo rende ancor più chiare le responsabilità dei team in merito all'integrazione. Le stesse integrazioni vengono comprese meglio, perché i team che si occupano di sviluppo e deployment riconoscono le interdipendenze tra sistemi e applicazioni.

L'adozione dell'integrazione come base dell'infrastruttura e la conseguente distribuzione delle responsabilità tra i team crea un ambiente infrastrutturale dove gli approcci agili assumono una maggiore rilevanza.

15 *Florentine, Sharon, "IT project success rates finally improving." 27 febbraio 2017.* https://www.cio.com/article/3174516/project-management/it-project-success-rates-finally-improving.html

Conclusioni: raggiungere l'agile integration

L'agilità è un processo e non un progetto.

Reagire ai cambiamenti del mercato non è mai stato così importante per le aziende. Rilasciare nuovi servizi o aggiornare quelli esistenti in tempi rapidi è una capacità conferita in larga parte dai sistemi IT. Anche reinventare l'infrastruttura IT, ovvero la base su cui poggiano i servizi digitali, non è mai stato così importante.

I team delle infrastrutture sono storicamente vincolati a lunghi processi modulati che hanno l'obiettivo di limitare i rischi e assicurare stabilità. È tuttavia giunto il momento di migrare a una tipologia di infrastruttura non più basata sull'hardware o sulla piattaforma, ma sull'integrazione. L'integrazione non è una componente dell'infrastruttura, ma un approccio concettuale all'infrastruttura che include dati e applicazioni, hardware e piattaforme.

Red Hat definisce questo approccio "agile integration", un modo di usare i tre fondamenti dell'agile integration (le integrazioni distribuite, le API e i container) per creare un'infrastruttura più agile e adattiva. La tecnologia è spesso usata a supporto del cambiamento culturale: in questo caso opera affinché siano agili i team dell'infrastruttura, non solo il software che realizzano. A mano a mano che i team di infrastruttura si allineano ai principi Agile, sarà possibile integrare gradualmente la tecnologia a supporto dei cambiamenti. Non esiste un unico progetto che possa rimodellare un'intera azienda per renderla agile. Può essere più efficace implementare una tecnologia di agile integration o cambiare un settore di attività e poi estendere in modo incrementale tali cambiamenti.

Migliorare la reattività dell'infrastruttura IT è un obiettivo a lungo termine. Per progredire, non è necessario apportare modifiche radicali, né tentare cambiamenti isolati per poi propagarli all'azienda intera.

L'agile integration offre un framework tecnico e organizzativo che può aiutare l'azienda a rimodellare l'infrastruttura IT.

ITALIA
it.redhat.com
italy@redhat.com

EUROPA, MEDIO ORIENTE, E AFRICA (EMEA)
00800 7334 2835
it.redhat.com
europe@redhat.com

facebook.com/RedHatItaly
twitter.com/RedHatItaly
linkedin.com/company/red-hat

it.redhat.com
#F20123_1119