RED HAT®
CONSULTING

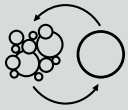
redhat.

WHITEPAPER

RINNOVARE LE VECCHIE APPLICAZIONI

Schemi e processi di sviluppo dei software per una modernizzazione continua delle applicazioni

Zohaib Khan



"Diciamocelo, tutto quello che facciamo è scrivere oggi i software di domani."

MARTIN FOWLER
THOUGHTWORKS CHIEF ARCHITECT

RIEPILOGO ESECUTIVO

Guardando ai passati decenni dell'informatica e dell'ingegneria dei software, una cosa appare chiara: tutto cambia. In tutti i periodi, la graduale trasformazione di hardware, linguaggi, infrastrutture e metodologie è stata affiancata da innovazioni sorprendenti.

Questa evoluzione ha consentito al settore IT di mantenersi al passo con le richieste in continua evoluzione, ma questo sforzo non è stato né facile né economico. Molti dei budget destinati al settore IT sono consumati da interventi di manutenzione sulle vecchie tecnologie, e restare al passo con aggiornamenti e migrazioni spesso logora tutte le risorse prima ancora che si veda l'ombra di qualche vantaggio economico.

Con il giusto approccio, però, è possibile modernizzare un portafoglio di applicazioni in maniera tale da ottenere un rendimento in meno tempo e a più basso costo, rendendo più facile ed economico lo sforzo di tenersi al passo con prodotti e tecnologie in continua evoluzione.

In questo whitepaper, Red Hat esamina tre specifici schemi di sviluppo software che consentono di modernizzare le applicazioni attualmente esistenti. Questi approcci alla modernizzazione consentono il passaggio dalle applicazioni esistenti ad architetture e infrastrutture più moderne, per renderle accessibili alle nuove applicazioni. Vengono inoltre esaminate le condizioni che portano alla riscrittura di un software quando questa è l'unica strada percorribile. Gli approcci descritti aiutano le imprese a capire come trarre il massimo dalle applicazioni esistenti e a stabilire delle buone pratiche per la modernizzazione continua che aiuteranno l'azienda oggi e nel futuro.

SVILUPPO E DISTRIBUZIONE DELLE APPLICAZIONI

Non molto tempo fa, le applicazioni erano codificate in linguaggi di programmazione e compilate in formati unici per un processore e un sistema operativo specifici. Erano generalmente complete, tendenzialmente grandi e ospitate su datacenter privati. Tutti pensavano che avrebbero avuto una lunga vita. Erano state realizzate utilizzando approcci per lo sviluppo di software basati su requisiti formali e rigidi e con tempi di sviluppo molto lunghi.

Adesso è tutto cambiato. Oggi, quelle applicazioni vengono definite monolitiche tradizionali, e sono i dinosauri delle applicazioni aziendali. Sebbene abbiano onorato lo scopo per cui erano state create, la velocità dell'innovazione tecnica e aziendale le ha rese un peso per le aziende.

L'innovazione ha portato al modello di sviluppo e distribuzione delle applicazioni oggi comunemente in uso, il processo DevOps, che guida la creazione di microservizi distribuiti su container ospitati nel cloud. Consideriamo per un attimo il progresso da quattro punti di vista diversi: metodologia, architettura, distribuzione e infrastruttura.

Il processo di sviluppo è diventato molto più rapido. Si è evoluto da una metodologia a cascata con requisiti fissi e tempi di realizzazione molto lunghi a una metodologia iterativa con release molto frequenti di funzionalità incrementali e a pratiche DevOps altamente collaborative con integrazione e risposta automatizzate e continue.

facebook.com/redhatinc
@redhatnews
linkedin.com/company/red-hat

redhat.com

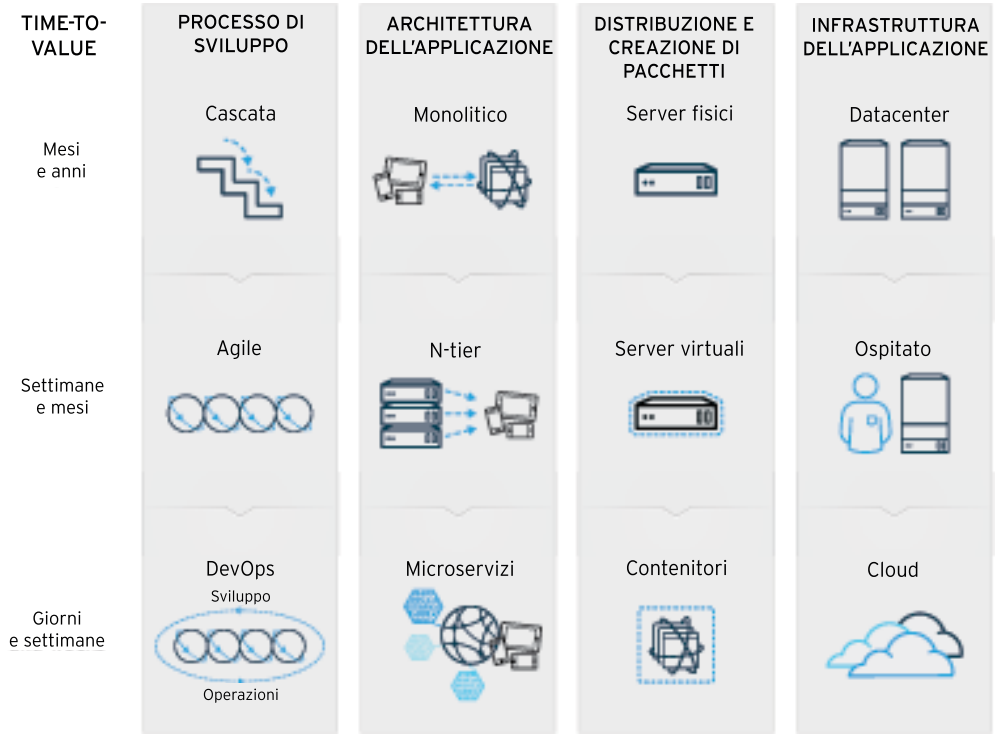


Figura 1: l'evoluzione delle fasi di sviluppo e distribuzione di un'applicazione.

Uno stack applicativo completo comprende tutti i componenti richiesti per sviluppare e distribuire un'applicazione, la metodologia usata per svilupparla e l'hardware sul quale funziona.

La tendenza nell'architettura delle applicazioni è la separazione delle funzionalità in componenti. Le applicazioni "tutto in uno" sono state suddivise in livelli diversi di interfaccia, logica e dati. L'architettura orientata ai servizi ha permesso di creare applicazioni utilizzando servizi dedicati su un comune Enterprise Service Bus. Questo, a sua volta, ha portato all'attuale modello basato su microservizi e API, dove altri servizi e applicazioni possono accedere a componenti altamente specializzati. I microservizi e le API partono dagli standard precedenti dei servizi Web e separano l'interfaccia (il modo in cui si accede) dall'implementazione (il modo in cui le applicazioni fanno quello che fanno), possono essere implementati in diversi linguaggi e vengono distribuiti su diversi sistemi.

La distribuzione delle applicazioni è ora diventata decisamente più flessibile. Le applicazioni, inoltre, non sono più strettamente associate all'hardware. Vengono scritte secondo degli standard, ad esempio Enterprise Java™, in maniera tale da poter essere distribuite su molte combinazioni diverse di hardware e sistemi operativi. Le macchine e i contenitori virtuali consentono loro di essere suddivise in pacchetti e distribuite più facilmente e per diversi host.

L'infrastruttura delle applicazioni si è evoluta dai grandi server specifici per una applicazione a server su scala orizzontale che supportano molte applicazioni. Oggi è normale per le applicazioni essere distribuite su più server in molti datacenter, cloud privati e cloud pubblici. In questo modo, la distribuzione è più veloce e le prestazioni e la disponibilità sono migliori.

L'evoluzione nello sviluppo e nella distribuzione delle applicazioni in queste quattro aree ha prodotto uno sviluppo iniziale più veloce, aggiornamenti più frequenti, una qualità più elevata, una maggiore capacità di far fronte alle esigenze aziendali, una maggiore flessibilità nelle operazioni e una riduzione dei costi.

La modernizzazione non comporta solo l'adozione di nuove tecnologie e pratiche, ma riguarda anche la gestione delle tecnologie più datate.

MODERNIZZAZIONE

Oggi ci riferiamo ai componenti software che supportano un'applicazione come uno stack. Per estensione, uno stack completo comprende tutti i componenti richiesti per sviluppare e distribuire un'applicazione, la metodologia usata per svilupparla e l'hardware sul quale funziona.

Le aziende hanno iniziato a usare nuovi prodotti e nuove tecnologie per nuovi progetti, ma gli stack tradizionali sono ancora in uso presso moltissime aziende.

Ad esempio, molte aziende che forniscono servizi finanziari hanno sviluppato applicazioni personalizzate già decine di anni fa. In quell'occasione, hanno utilizzato i terminali Bloomberg e le postazioni di lavoro degli intermediari, sistemi client-server e applicazioni web n-tier. Oggi, quelle stesse aziende stanno sviluppando altre applicazioni per clienti e dipendenti. Come risultato di questa crescita, oggi dispongono di decine di stack moderni e tradizionali in uso nel loro portafoglio di applicazioni.

La modernizzazione non comporta solo l'adozione di nuove tecnologie e pratiche, ma riguarda anche la gestione delle tecnologie più datate. Immaginiamo una vecchia casa riscaldata a carbone nella quale sono presenti delle stanze nuove riscaldate a gas. Iniziare a riscaldare tutta la casa con l'energia solare sarebbe costoso e non porterebbe grandi ritorni, ma avrebbe senso usare l'energia solare nella parte più nuova, tentando magari di far funzionare tutto sotto lo stesso tetto.

La modernizzazione delle applicazioni persegue due obiettivi principali: usare il più possibile le funzionalità e i dati esistenti nelle nuove applicazioni (derivando quindi nuovo valore da applicazioni vecchie) e portare i vantaggi dei nuovi processi, prodotti e tecnologie nelle applicazioni più datate.

Con il giusto approccio alla modernizzazione, con la giusta metodologia e gli schemi più adeguati, le imprese possono guidare il cambiamento.

TRE SCHEMI DI SVILUPPO PER LA MODERNIZZAZIONE

I tre schemi di sviluppo presentati di seguito rappresentano un approccio unificato alla modernizzazione delle applicazioni. I primi due estendono la vita e l'utilità delle applicazioni esistenti, evitano cambiamenti radicali a tutto il portafoglio applicativo e pongono le basi per il terzo schema, che prevede il refactoring dell'applicazione e l'aggiornamento dell'architettura. In nessuno dei casi si è costretti a riscrivere un'applicazione ex novo per beneficiare dei vantaggi offerti dalla modernizzazione. Perfino le applicazioni monolitiche possono trarre i loro vantaggi senza essere riscritte.

LIFT-AND-SHIFT

L'approccio lift-and-shift permette di modernizzare il modo in cui le applicazioni esistenti sono suddivise in pacchetti e distribuite. I componenti esistenti vengono infatti distribuiti su una piattaforma moderna. Un esempio che ci è familiare è rappresentato dalla virtualizzazione delle applicazioni, dove un'applicazione viene organizzata in pacchetti con il sistema operativo ed eseguita come macchina virtuale invece che sull'hardware dedicato.

L'approccio lift-and-shift non è pensato per modernizzare l'architettura dell'applicazione. Al contrario, porta l'azienda su una piattaforma di distribuzione moderna lasciando agli sviluppatori il tempo di occuparsi del suo refactoring.

Questo approccio può essere utilizzato per migliorare le prestazioni delle applicazioni distribuendole su hardware più aggiornati e veloci. Le applicazioni diventano più flessibili grazie a semplici processi di distribuzione su piattaforme moderne. Questo può ridurre anche i costi operativi eliminando i server dedicati e centralizzando la gestione.

Ecco alcuni degli esempi più comuni di utilizzo dell'approccio lift-and-shift:

> In un'applicazione a tre livelli con presentazione, logica di business e servizi dati, ogni livello è ospitato su un diverso server Linux®. Ogni servizio viene riorganizzato sotto forma di un contenitore che include tutte le configurazioni e le dipendenze di runtime per il servizio in questione. I contenitori sono distribuiti in un ambiente Platform as a Service (PaaS) che si trova su un cloud pubblico.
>
> Un sistema di gestione dei contenuti (content management system, CMS) creato su J2EE e ospitato su quattro macchine virtuali viene riorganizzato come set di contenitori e distribuito in una PaaS. Oltre ai vantaggi offerti dai contenitori in ambiente PaaS, i team di sviluppo

beneficiano anche di un'esperienza di sviluppo integrata basata su integrazione continua e distribuzione continua.

In questi esempi, l'architettura rimane inalterata ma le applicazioni godono di una nuova vita su una moderna piattaforma di sviluppo. Questo offre agli sviluppatori più tempo per occuparsi del refactoring delle applicazioni e per modernizzare l'architettura. Ad esempio, quattro componenti CMS possono essere modificati in microservizi, e l'intero sistema CMS può essere incapsulato come API, divenendo in questo modo accessibile per tutte le applicazioni mobili, cloud e di altro tipo che necessitano di capacità di gestione dei contenuti.

L'approccio lift-and-shift non è adatto a tutte le applicazioni. Le applicazioni collegate a soluzioni specifiche per un venditore potrebbero infatti essere difficili da riorganizzare e ridistribuire. I sistemi operativi più vecchi potrebbero non essere supportati dalle più moderne piattaforme di sviluppo.

AGGIUNTA DI NUOVI LIVELLI

Molte aziende creano valore commerciale fornendo applicazioni attraverso nuovi canali, ad esempio quelli mobili o Salesforce, e integrandole con le applicazioni dei partner. Aggiungere nuovi livelli è uno schema di sviluppo software che può rendere le funzionalità delle applicazioni esistenti accessibili per le nuove applicazioni e i nuovi canali di distribuzione, riducendo i tempi e i costi di sviluppo perché le funzionalità non devono essere sviluppate nuovamente. Laddove esistono, è più conveniente utilizzare funzionalità complesse e critiche, il cui funzionamento è stato già testato accuratamente nel tempo.

Aggiungere nuovi livelli significa creare un nuovo livello del software applicativo che ingloba la funzionalità e i dati esistenti con un'interfaccia accessibile per le nuove applicazioni. Per evitare di introdurre una complessità eccessiva, il livello di solito non prevede logiche di business aggiuntive, ma funge semplicemente da adattatore tra il vecchio e il nuovo.

Di solito, non è necessario modificare l'applicazione esistente, il che rende questo approccio molto allettante quando il codice sorgente non è disponibile o quando è troppo rischioso modificare l'applicazione esistente.

Per quanto riguarda l'approccio lift-and-shift, l'architettura dell'applicazione esistente non subisce modifiche, ma l'aggiunta di nuovi livelli permette di utilizzare la funzionalità esistente per creare nuove applicazioni e nuovi servizi.

Questo approccio getta le basi per una modernizzazione graduale dell'architettura delle applicazioni. Nel tempo, le vecchie funzionalità possono essere riscritte e i vecchi stack possono essere ritirati. Questo approccio può essere utilizzato per migrare lentamente le applicazioni esistenti (conosciute come strangler application) ed evitare di riscriverle da zero.

Ecco alcuni esempi di aggiunta di nuovi livelli:

- Un'applicazione commerciale esistente si trova su un sistema operativo non supportato dalle moderne piattaforme di visualizzazione o container. Ad essa si può accedere solo attraverso un'API standard. Viene scritto un microservizio adattatore per accedere alla API. Questo microservizio viene poi utilizzato dalla nuova applicazione e da altri microservizi per accedere alle funzionalità dell'applicazione commerciale.

- Un'applicazione di ordinazione è scritta in Visual Basic e impiega procedure di archiviazione molto complesse nel sistema di gestione del database. Viene sviluppata una nuova applicazione di ordinazione per cellulari e tablet. Viene creato un livello adattatore con un'interfaccia a microservizi per la nuova applicazione. L'adattatore nasconde il database e le procedure archiviate. Tutte le nuove applicazioni accedono all'adattatore che converte le richieste in chiamate alle procedure archiviate.

In entrambi gli esempi, l'aggiunta di nuovi livelli rappresenta solo una soluzione temporanea in attesa di un progetto di sviluppo più completo e basato su un'architettura moderna. Questo approccio è compatibile con l'approccio lift-and-shift e può essere applicato congiuntamente ad esso. Infatti, mentre l'uno rende disponibili per le nuove applicazioni le funzionalità delle applicazioni esistenti, l'altro rende la distribuzione e la gestione delle applicazioni esistenti più facili ed economiche.

Ed entrambi concedono agli sviluppatori più tempo per occuparsi del refactoring e della riscrittura delle applicazioni esistenti utilizzando un'architettura moderna.

RISCRITTURA

Riscrivere un'applicazione è completamente diverso dal creare una nuova applicazione dal nulla: la riscrittura è il processo attraverso il quale vengono create nuove funzionalità che consentono di sostituire e ritirare le applicazioni esistenti. Come parte di una strategia complessiva di modernizzazione, la riscrittura può seguire la fase di lift-and-shift o di aggiunta di nuovi livelli, ed è in realtà l'unico modo a nostra disposizione per aggiornare un'architettura per uno stack totalmente moderno.

Riscrivere un'applicazione esistente è di solito l'opzione meno allettante delle tre. Infatti, è costosa, porta via molto tempo e il ritorno economico è spesso visibile dopo molti anni. Se l'applicazione non è in grado di creare nuovo valore, la sua riscrittura è difficile da giustificare per le aziende con un budget limitato.

Ciononostante, ci sono casi in cui questa è l'unica opzione disponibile. Ad esempio, potrebbero esistere applicazioni molto vecchie ospitate su sistemi operativi e hardware non più supportati dai venditori. Le aziende di solito non vogliono utilizzare i principali sistemi aziendali senza la rete di sicurezza costituita dal supporto dei propri venditori, e a volte non sono in possesso delle competenze necessarie a far funzionare i sistemi più datati.

Quando riscrivere è l'unica opzione, l'approccio migliore è migrare gradualmente le funzionalità dalle vecchie applicazioni aggiungendo nuovi livelli. Potrebbe anche essere una buona idea ritardare alquanto la riscrittura, dal momento che alcune funzionalità diventano obsolete e probabilmente non occorrerà migrarle. Bisogna resistere alla tentazione di migrare tutti i vecchi comportamenti, ma occorre pianificare e stabilire delle priorità come se si stesse sviluppando una nuova applicazione. Questo permetterà di creare delle applicazioni più flessibili e in grado di adattarsi ai cambiamenti futuri.

SCELTA DELL'APPROCCIO

Non esiste un approccio in assoluto migliore degli altri: tutto dipende dall'applicazione, dall'azienda e dal contesto. Questi approcci riguardano diverse aree dello sviluppo e della distribuzione delle applicazioni, per questo tutti potrebbero essere adatti o completamente inutili per un caso specifico. In generale, per modernizzare un portafoglio di applicazioni occorre fare ricorso a più di un approccio.

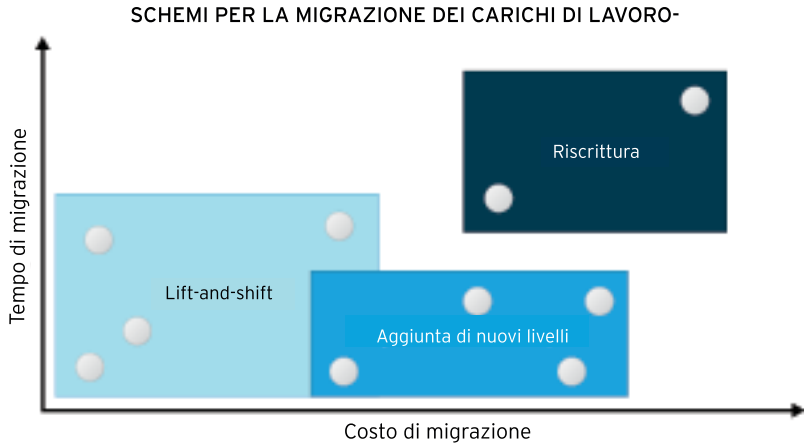


Figura 2: schemi per la migrazione dei carichi di lavoro in base a tempo e costi. Non sono inclusi i vantaggi perché dipendono dall'applicazione e dal suo stato attuale e futuro.

Il lift-and-shift è solitamente quello più economico. L'aggiunta di nuovi livelli può risultare più costosa poiché richiede uno sforzo di sviluppo, ma spesso può essere applicata più velocemente. La riscrittura è quasi sempre la soluzione più costosa e quella che richiede più tempo (Figura 2).

Questi schemi di sviluppo software aiutano le imprese a capire come trarre il massimo dalle applicazioni esistenti e stabilire delle buone pratiche per la modernizzazione continua, che aiuteranno l'azienda oggi e nel futuro.

UNA METODOLOGIA DI MODERNIZZAZIONE

I tre schemi di modernizzazione presentati aiutano gli architetti e i responsabili dello sviluppo a donare nuova vita alle applicazioni esistenti, ma di solito rappresentano solo una parte di una iniziativa più ampia. In questo caso, una metodologia formale aiuta a organizzare gli sforzi, a contenere i costi e a generare valore per le imprese.

Red Hat ha elaborato un processo a più fasi e iterativo per la modernizzazione delle applicazioni (Figura 3). In sostanza, si elabora un piano che viene ripetuto più volte, e ogni ripetizione determina un nuovo valore incrementale. Questo riduce i rischi legati a tagli e sostituzioni.

Le sessioni iterative durante la fase di scoperta consentono di comprendere lo stato e gli obiettivi di un'applicazione. Gli azionisti sono sempre alla ricerca di nuovi modi per modernizzare le proprie aziende e stabiliscono una serie di priorità allo scopo di ottenere dai propri collaboratori l'impegno di portare a termine l'impresa.

Figura 3: il processo di modernizzazione delle applicazioni di Red Hat.

La fase di progettazione è accompagnata da analisi, verifica dei concetti e elaborazione di progetti pilota. L'analisi automatizzata dei codici aiuta a individuare gli eventuali problemi e a valutare gli sforzi necessari, mentre gli utenti esaminano gli stack e le architetture per individuare lo stato delle applicazioni e stabilire quali sono gli elementi da modernizzare. In questa fase vengono stabiliti gli obiettivi da raggiungere, viene scelto l'approccio da adottare e viene elaborato un piano di distribuzione completo.

Questo piano di distribuzione viene messo in atto nella fase successiva. Organizzato in interazioni multiple al fine di ridurre i rischi e imparare dagli errori riscontrati nelle fasi precedenti, questo modello deve essere guidato da un centro di eccellenza composto da specialisti della migrazione. Sono loro, infatti, a sapere quali sono le migliori pratiche da seguire e ad assicurare la sostenibilità del progetto durante le sue fasi evolutive e nonostante ogni eventuale cambiamento dei vertici aziendali.

CONCLUSIONE

In presenza di un portafoglio di applicazioni molto assortito, la modernizzazione può sembrare un ostacolo insormontabile, soprattutto se si ha a che fare con vecchi stack e hardware obsoleto. Non è facile vedere al di là del tempo e degli sforzi richiesti, e spesso l'unico criterio importante sembra essere legato al contenimento dei costi operativi.

La metodologia elaborata da Red Hat semplifica il problema e consente di migrare solo alcune applicazioni alla volta. Le applicazioni vengono valutate in base al costo e al tempo necessari per la modernizzazione, ai costi operativi post-modernizzazione e al valore commerciale potenziale.

I tre approcci presentati offrono diverse possibilità di estensione della vita delle applicazioni esistenti e aiutano a determinare i casi in cui è necessario riscrivere un'applicazione da zero. Con il giusto approccio alla modernizzazione, con la giusta metodologia e gli schemi più adeguati, le imprese possono guidare il cambiamento.

Red Hat® Consulting impiega gli approcci descritti per aiutare le imprese a ottenere più valore dalle proprie applicazioni. Per maggiori informazioni visitare la pagina redhat.com/en/resources/application-migration-modernization-consulting-jboss-middleware-datasheet.

RED HAT®
CONSULTING

redhat.

INFORMAZIONI SULL'AUTORE

Zohaib Khan è attualmente Application Migration Practice Lead e uno dei responsabili della PaaS Community of Practice presso Red Hat. Ha presenziato a numerose conferenze, tra cui Red Hat Summit, StackWorld e Red Hat Technical Exchange. È autore di articoli sul processo DevOps e sull'innovazione tecnologica. Prima di unirsi a Red Hat, è stato senior manager nel settore dei servizi sanitari e finanziari e co-fondatore di una società di servizi tecnologici.

INFORMAZIONI SU RED HAT

Red Hat è il fornitore leader a livello mondiale di soluzioni software open source. Utilizza un approccio basato sulla collaborazione dell'intera comunità per fornire funzionalità affidabili e ad alte prestazioni per cloud, Linux, middleware, archiviazione e virtualizzazione. Offre inoltre servizi eccellenti per supporto tecnico, formazione e consulenza. Red Hat agisce come un hub, connettendo un network globale di imprese, partner e community open source, e aiutando a creare tecnologie fondamentali e innovative finalizzate a liberare risorse per la crescita e preparare i clienti al futuro dell'IT.

NORD AMERICA
1 888 REDHAT1

EUROPA, MEDIO ORIENTE E AFRICA
00800 7334 2835
europe@redhat.com

ASIA PACIFICO
+65 6490 4200
apac@redhat.com

AMERICA LATINA
+54 11 4329 7300
info-latam@redhat.com

facebook.com/redhatinc
@redhatnews
linkedin.com/company/red-hat

redhat.com
#046201_1216