Introduzione

Questo libro esamina vari modi per strutturare, progettare e costruire il software, in particolare i sistemi distribuiti, per resistere a tutti i problemi e le "paludi" che si troverà a incontrare nel mondo reale. Lo preparerete per eserciti di utenti scriteriati che escogiteranno atti folli e imprevedibili. Il vostro software sarà sotto attacco fin dal momento in cui lo rilascerete. Dovrà resistere ai forti venti dei Flash Mob e alla pressione di un attacco DDoS, sferrato da un esercito di tostapane IoT non abbastanza sicuri. Osserverete con uno sguardo severo il software che non è riuscito a superare questo test e troverete il modo per assicurarvi che il software sopravviva a contatto con il mondo reale.

Chi dovrebbe leggere questo libro

Ho indirizzato questo libro agli architetti, progettisti e sviluppatori di sistemi software distribuiti, come siti web, servizi web e progetti EAI, tra gli altri. Si tratta di sistemi che devono essere sempre disponibili, o l'azienda perderà del denaro. Magari si tratta di sistemi di commercio elettronico, che generano utili direttamente attraverso le vendite o di sistemi interni critici, utilizzati dai dipendenti per svolgere il loro lavoro. Se qualcuno rischia di essere licenziato perché il software ha smesso di funzionare, allora questo libro è per voi.

Come è organizzato questo libro

Il libro è suddiviso in quattro parti, ognuna delle quali è introdotta da un caso di studio. La Parte I, *Creare stabilità*, mostra come mantenere attivi i sistemi, curando l'uptime del sistema. Nonostante le promesse di affidabilità dovuta alla ridondanza, i sistemi distribuiti esibiscono una disponibilità più simile a un "doppio otto" che a un "cinque nove". La stabilità è un prerequisito necessario, prima di ogni altra preoccupazione. Se il sistema va giù e si impianta ogni giorno, questa diventa la prima priorità per tutti: a nessuno interessa nient'altro. Le correzioni a breve termine – e i ragionamenti a breve termine – saranno predominanti in un ambiente di quel tipo. Non si può pensare a un futuro senza stabilità, quindi inizieremo cercando il modo di realizzare una base stabile.
Dopo la stabilità, la preoccupazione successiva riguarda le operazioni in corso. Nella Parte II: *Progettare per la Produzione*, vedrete che cosa significa vivere in Produzione.

Avrete a che fare con la complessità degli attuali ambienti di produzione in tutti i loro più complessi dettagli di virtualizzazione, containerizzazione, bilanciamento del carico, individuazione dei servizi. Questa parte illustra i migliori modelli per assicurare il controllo, la trasparenza e la disponibilità nei data center fisici e negli ambienti cloud.
Nella Parte III: *Consegnare il sistema*, parlerò della fase di deployment. Oggi abbiamo a disposizione ottimi strumenti per riversare bit sui server, ma in realtà questa si rivela essere la parte più facile del problema. È molto più difficile introdurre frequenti e piccoli cambiamenti senza disturbare i consumatori. Vi parlerò della progettazione per il deployment e di deployment senza tempi di inattività; poi affronterò il controllo di versione su più servizi differenti, che è sempre un argomento complesso!
Nella Parte IV: *Risolvere i problemi sistemici*, esaminerete il corso di vita del sistema nell'ambito dell'ecosistema globale delle informazioni. Se la release 1.0 rappresenta la nascita del sistema, allora avrete bisogno di pensare anche alla sua crescita e al suo sviluppo successivo. In questa parte, imparerete a costruire sistemi in grado di crescere, piegarsi e adattarsi nel tempo. Parlerò di architettura evolutiva e di "conoscenza" condivisa fra più sistemi. Infine, imparerete a costruire sistemi "robusti" attraverso la disciplina emergente di "chaos engineering", che applica a un sistema la casualità e uno stress deliberato, con lo scopo di migliorarlo.

Informazioni sui casi di studio

Ho incluso diversi e ampi casi di studio per illustrare i temi principali di questo libro. Questi casi di studio sono tratti da eventi reali e da insuccessi di veri sistemi, che ho osservato personalmente. Questi insuccessi sono stati molto costosi e imbarazzanti per coloro che ne sono stati vittima. Pertanto, ho celato alcune informazioni per proteggere l'identità delle aziende e delle persone coinvolte. Ho anche cambiato i nomi dei sistemi, delle classi e dei metodi. Soltanto questi particolari inessenziali sono stati cambiati. In ogni caso, ho mantenuto lo stesso settore operativo, la stessa sequenza di eventi, le stesse modalità di errore, la stessa propagazione degli errori e lo stesso risultato. Non ho taciuto i costi di questi insuccessi. Si è trattato di aziende reali e di denaro reale. Ho conservato quelle cifre per sottolineare la gravità del danno e quindi l'importanza di queste pagine. Quando i sistemi "vanno giù", quelli che si perdono sono soldi veri.

Codice degli esempi

Sul sito web dell'editore originale inglese, all'indirizzo `https://pragprog.com/titles/mnee2/46`, è possibile scaricare il codice degli esempi.
Ora, cominciamo a parlare di cosa significhi *vivere in Produzione*.