INTRO CON CONTINUITA’ CON AGILE

Molti credono erroneamente che Agile e DevOps siano due metodologie contrapposte ma in realtà quest’ultimo non è altro che un miglioramento dell’Agile avvenuto nel tempo.  
La grande innovazione è infatti quella di proporre una collaborazione tra Developer, orientati alla creazione di cambiamenti e all'aggiunta o modifica di funzionalità, e Operation, tendenti alla creazione di stabilità e al miglioramento dei servizi, in tutte le fasi del ciclo di vita del prodotto, con la sicura convinzione che questa sinergia possa generare un enorme valore sia per la qualità del prodotto finito, sia per la velocità con cui viene realizzato.  
Devops riesce quindi ad estendere gli ideali dell’Agile oltre il codice, concentrandosi sull’intero servizio.  
Vediamo insieme una bellissima interpretazione di come il DevOps possa essere considerato una conseguenza dell’Agile.

Mentre la metodologia Agile si fonda su principi che riguardano Valori, Principi, Metodi e Pratiche, DevOps aggiunge anche gli Strumenti. Secondo il Manifesto Agile infatti, gli strumenti assumono una minima importanza, ma questo ha portato nel tempo a trascurarli notevolmente con conseguenze spesso non in linea con l’obiettivo del cliente.  
È bello pensare alla differenza tra i due metodi in questo modo: mentre Agile risolve i problemi della tecnologia, DevOps cerca di risolvere più ampiamente un intero problema di business attraverso un insieme di pratiche, cultura e valori che guidano il cambiamento verso software più solido, che raggiunge più presto la produzione e con meno intralci durante il suo ciclo di vita.  
Infatti DevOps rende chiaro l’obiettivo comune di Developers e Operators per puntare all’ottimizzazione dell’intero sistema invece che alle ottimizzazioni locali, cioè vuole raggiungere gli obiettivi del business risolvendo il problema del cliente, che ci sia da agire su ambiti del team dei Developers che Operators.

INTRO DEVOPS

Il DevOps, infatti, è un set di pratiche e di cambiamenti culturali supportati da strumenti automatici e processi di Lean Management, che consente di automatizzare il rilascio del software rispetto alla sua catena di produzione, permettendo alle organizzazioni di poter contare su un software e applicazioni di qualità superiore e sicura in modo estremamente più rapido, per accontentare i clienti nel modo migliore e più rapidamente.

na metodologia di sviluppo del software che punta alla comunicazione, collaborazione e integrazione tra sviluppatori e addetti alle operations dell'information technology (IT).[1] DevOps vuole rispondere all'interdipendenza tra sviluppo software e IT operations, puntando ad aiutare un'organizzazione a sviluppare in modo più rapido ed efficiente prodotti e servizi software.

AGILE DENTRO DEVOPS

Il DevOps è una medaglia a due facce. Sul fronte dello sviluppo in un ambiente DevOps la base è la metodologia di lavoro prescelta. Detto questo, un ambiente DevOps non può non fare riferimento alla metodologia Agile, e cioé a quella summa di principi derivati dal “Manifesto Agile” che nel 2001 ha definito un modello di sviluppo focalizzato sull’obiettivo di consegnare al cliente, in tempi brevi e frequentemente (early delivery – frequent delivery), software funzionante e di qualità.  
Rispetto ai metodi tradizionali a cascata o ad altri processi software, le pratiche Agile presuppongono la formazione di team di sviluppo piccoli, cross-funzionali e auto-organizzati, lo sviluppo iterativo e incrementale, la pianificazione adattiva e il coinvolgimento diretto e continuo del cliente nel processo di sviluppo.  
Se si vuole mettere in piedi un IT shop di successo, il primo passo è dunque quello di scegliere una metodologia Agile intuitiva, con dei framework di sviluppo come Scrum (che enfatizza tutti gli aspetti di gestione di progetto legati a contesti in cui è difficile pianificare in anticipo) o KanBan (che aiuta a visualizzare e rendere esplicito il flusso di lavoro per riconoscere prima le opportunità di miglioramento, limitando il work in progress). Questi framework sono importanti perché aiutano i team di sviluppo a definire rapidamente gli obiettivi e le priorità, ad assegnare i compiti e a identificare dove possono verificarsi problemi nello sviluppo di un processo.

CI E CD

Un’altra caratteristica fondamentale di un sistema di sviluppo DevOps sono l’integrazione continua (CI) e l’erogazione continua e/o la distribuzione continua (CD). CI significa che nel processo di sviluppo i test su una porzione di codice sono continui e automatici, mentre CD significa che il processo di messa in produzione del codice validato dopo il dovuto collaudo diventa automatica. È così che si accelerano i tempi di rilascio.  
In passato, ad esempio, molte aziende metteva in produzione il nuovo codice ad orari prestabiliti. Ma la velocità del business ha reso questo modello per cicli di rilascio piuttosto obsoleto e in antitesi al DevOps che, invece, punta proprio ad automatizzare il ciclo di rilascio per renderlo il più possibile immediato.  
Il codice, infatti, con il Devops è conservato in un repository di sorgenti che viene utilizzato come archivio ma anche come fonte per controllare le varie versioni.

IAC

Il codice del software non è l’unico elemento archiviato nel repository di Git. Sempre più spesso, infatti, vengono memorizzati anche gli script che contengono tutti i dettagli delle configurazioni e i modelli creati con gli strumenti di gestione della configurazioni come, ad esempio, Puppet e Chef, che sono due tra i linguaggi più popolari.  
La generazione di metodi automatici per configurare e implementare l’infrastruttura ha dato origine al concetto di infrastructure as a code (IaC).  
Nello specifico, avere un’infrastruttura come codice significa che questa può essere incorporata in altri processi DevOps, sia per le fasi di test che per la messa in produzione. Rally Software memorizza tutte le sue configurazioni infrastrutturali nel repository GitHub, testate e distribuite in modo continuo esattamente come il resto dei processi di integrazione e implementazione continui processati decine di volte al giorno. Che si tratti di codice del software o di infrastrutture, tutto passa dalla stessa metodologia automatica di processo, Indipendentemente da ciò che viene modificato.  
“Un altro vantaggio delle IaC è che si può contare su soluzioni ever green – ha aggiunto Alain Gaeremynck, manager DevOps nonché enterprise architect presso la canadese Yellow Pages Group – dal momento che è possibile mantenere sempre aggiornati alle ultime versioni sia i sistemi che i pacchetti. Noi abbiamo sposato la causa IaC. Una volta impiegavamo tempo e risorse nel costruire infrastrutture da monitorare costantemente e da aggiornare. Oggi, invece come parte del processo di generazione lavoriamo così: distruggiamo e ricostruiamo ex novo, il che ci permette di introdurre aggiornamenti Java o sviluppare per l’ultimo sistema operativo immesso sul mercato, ad esempio, perché in ogni caso passiamo da un ciclo QA”.

INTRO SUL PERCHE È NATO DEVOPS

Development da un lato e Operations dall’altro, il DevOps è una metodologia di sviluppo del software che sfrutta le nuove logiche della condivisione e della collaboration ma anche di un crowdsourcing più verticale. Ecco cosa significa DevOps ed è possibile portando nuovi livelli di integrazione tra gli sviluppatori e gli addetti alle operations per accelerare i tempi di progettazione, testing e di rilascio delle soluzioni applicative aziendali sia in ambienti tradizionali che in ambienti cloud. Il tutto assicurando la qualità e la sicurezza del software sviluppato.  
La complessità è infatti spesso il riflesso della crescita incontrastata delle infrastrutture It; un debito non risolto nemmeno con l’introduzione di tecnologie quali Soa, virtualizzazione, cloud computing che dovrebbero, al contrario, garantire proprio la semplificazione dell’It. La complessità, dunque, regna sovrana all’interno dei data center delle aziende: da un sondaggio condotto da Gartner, le infrastrutture It esistenti (comprese le architetture applicative) risultano essere ‘very complex’ nel 54% dei casi.  
Analizzando un po’ più in dettaglio le ‘cause’ di tale complessità, Gartner evidenzia come le problematiche maggiori siano da rilevare nell’ambito dei processi It, da un lato, e nel proliferare di tool tecnologici, dall’altro. Situazioni che rendono difficoltoso sia il rilascio sia il mantenimento di servizi It efficienti (ed efficaci sul piano del business).  
E dato che uno dei servizi It più impattante sul piano del business è quello applicativo, Gartner ha voluto indagare per capire meglio quali possano essere le ‘disfunzioni’ all’interno della famiglia It che si riflettono poi sull’azienda. Una delle aree maggiormente critiche risulta essere la ‘relazione’ tra l’application development e le It operation. Ben il 47% degli intervistati ha dichiarato che i rapporti tra questi differenti team è del tutto ‘un-collaborative’. Le lamentele più comuni delle operation derivano dal fatto che spesso queste vengono ‘lasciate fuori’ dalle decisioni sia quelle inerenti le architetture It sia quelle di business, con il risultato di un minor controllo sui service level (compresi i costi).

DATI SU APPLICAZIONE DEVOPS NELLE AZIENDE

Gli analisti raccontano uno scenario di indubbia e prosperosa crescita della diffusione e dell’adozione della metodologia DevOps nelle aziende. Tuttavia, andando ad analizzare in dettaglio alcuni dati di un’indagine compiuta da Freeform Dynamics l’anno scorso appariva evidente la difficoltà di tale percorso: solamente il 40% dei manager interpellati (su un panel di intervistati di 1.442 tra business manager e responsabili It, tra i quali 506 in Europa) dichiarava un’adozione diffusa della metodologia DevOps lungo tutta la propria azienda; il 32% la stava ‘sperimentando’ solo in alcune aree mentre il restante 28% era ancora fermo a guardare, anche questione Return on investment – Roi del DevOps non è infatti da sottovalutare.  
Per quanto riguarda l’Italia, sempre Freeform Dynamics ha rilevato che – in uno scenario in cui l’81% degli intervistati italiani riconosce l’importanza di garantire un’adeguata conoscenza delle priorità di business nell’ambito dell’It, ma solo il 38% mette in atto questo principio e sempre l’81% considera importante disporre dell’infrastruttura e degli strumenti giusti, ma solo il 26% ha già raggiunto questo risultato – comunque l’84% dei partecipanti al sondaggio ritiene fondamentale acquisire le competenze informatiche necessarie per mettere in pratica la metodologia DevOps, ma solo il 35% le ha già adottate.  
Eppure, le imprese che mostrano un elevato livello di adozione del DevOps nella realizzazione delle loro iniziative digitali sono anche quelle che hanno raggiunto i migliori risultati aziendali. In tutta Europa, tra il 70 e l’80% di queste realtà ha ottenuto ‘significativi benefici misurabili’ in termini di fidelizzazione dei clienti e acquisizione di nuovi e ha conseguito risultati concreti nei nuovi flussi di ricavi (ossia con una crescita di fatturato generato direttamente da nuovi servizi digitali).  
In generale, ciò che appare evidente nelle analisi delle società di ricerca è che la ‘filosofia’ DevOps offre miglioramenti che variano a seconda della posizione in cui si trova l’azienda nel percorso di implementazione: ad esempio, il 46% dei decision maker It intervistati da Vanson Bourne afferma di aver già visto aumentare la frequenza del deployment dei propri software e servizi, mentre un altro 44% prevede di ottenere gli stessi risultati nel breve periodo; circa il 39% registra un maggior numero di clienti e/o utenti finali che utilizza i software e i servizi aziendali, e il 39% conferma un incremento della collaborazione tra reparti; oltre un terzo (36%) afferma di aver visto aumentare la qualità e le performance dei software e il 34% di aver ridotto il tempo dedicato alla correzione e alla manutenzione delle applicazioni.  
In sintesi, alcune delle ragioni che rendono questa metodologia interessante per le aziende si evincono dalle risposte collezionate sia da Vanson Bourne sia da Freeform Dynamics: secondo quasi la metà degli intervistati in entrambi i panel “nell’economia delle applicazioni la qualità del software deve migliorare”. Anche le performance applicative sono citate come uno dei driver di DevOps, poiché le aziende sanno che rischiano di perdere i propri clienti qualora l’interfaccia utente non sia intuitiva o l’app troppo lenta rispetto alle esigenze e alle aspettative. Uno dei fattori che spinge particolarmente la crescita del DevOps è infatti la maggiore attenzione all’esperienza del cliente.  
Di contro, le ragioni a testimonianza di coloro che si sono avvicinati al DevOps solo in via sperimentale o di coloro che ancora non sanno come ‘muoversi’ sono prevalentemente da ricondurre alla misurazione del valore, dell’efficacia e del Roi del DevOps; malgrado quest’ultimo stia conquistando popolarità tra chi per primo l’ha adottato, mancano infatti ancora sistemi di misurazione adeguati a comunicare il successo di un’implementazione.

DEFINIZIONE CONTRASTO TRA DEV E OPS

Invece, in molte organizzazioni, lo sviluppo del software e la gestione dei sistemi sono in divisioni differenti e poiché lo sviluppo è generalmente guidato dalle necessità dell'utente, per continue modifiche e conseguenti rilasci, i gruppi operativi sono concentrati sulla disponibilità e affidabilità dei servizi, nonché sulla gestione dei costi. Ciò produce un "gap" tra sviluppo e gestione dei servizi che rallenta il passaggio in produzione.

DATI SU OPINIONI DEVOPS

<https://www.01net.it/agile-devops-insieme/>

GIT DEVOPS

Il codice, infatti, con il Devops è conservato in un repository di sorgenti che viene utilizzato come archivio ma anche come fonte per controllare le varie versioni. Quello di Git (un sistema di controllo versione che serve a registrare i cambiamenti che si fanno su un file o su una serie di file nel tempo, così da poter richiamare una versione specifica di quei dati in qualsiasi momento) è superiore anche a molti repository opensource, avendo sistemi di controllo più evoluti rispetto, ad esempio, ai meno recenti Concurrent Versions System (CVS) o Apache Subversion.  
Scritto nientemeno che da Linus Torvalds, il padre di Linux, Git è nato in risposta a un bisogno della comunità opensource che chiedeva un sistema decentrato a cui gli sviluppatori di tutto il mondo avrebbero potuto comodamente accedere. Come sottolieano gli esperti, si tratta di uno strumento decentralizzato funziona bene anche in azienda, dove le squadre di sviluppo possono essere distribuite tra più divisioni e tra più sedi geografiche differenti.

CI

Cominciamo con questo articolo una serie che affronterà aspetti vari della Continuous Integration (CI): l’integrazione continua è ormai entrata a far parte delle pratiche seguite dagli sviluppatori in contesti diversi e ha ormai alle spalle una serie di esperienze, trattazioni, riflessioni che ne fanno uno degli argomenti più conosciuti.   
Il problema a cui si risponde con la CI è il la divergenza delle versioni del codice e quindi le conseguenti integrazioni bibliche che a volte portano addirittura a dover rinunciare a pezzi di software sviluppato.  
Per quanto oggi possa sembrarci “assurdo”, per molti anni la prassi dello sviluppo è stata quella di creare “indipendentemente” dei componenti software che poi andavano integrati in un secondo tempo. E quando arrivava questa seconda fase, spesso erano dolori. Ci si ritrovava con porzioni di software a volte decisamente divergenti perché sviluppate per lunghi periodi di tempo senza “agganciarle” alle altre: il termine coniato per definire tale situazione fu integration hell, un vero “inferno dell’integrazione”.  
ogni piccolo commit di codice che facciamo nel nostro sistema di versionamento (Version Control Systems, VCS) preferito (git, Subversion, Mercurial, etc.) deve essere integrato il prima possibile e automaticamente da una macchina appositamente configurata che eseguirà la compilazione del codice e lancerà la suite di test, il che significa: “compila tutto senza rompere niente”.

CI E COLLEGAMENTO A CD

Per integrazione continua si intende principalmente la fase di creazione di build o di integrazione del processo di rilascio del software e implica sia un componente di automazione (ad es. l'integrazione continua o un servizio di creazione di build) sia un componente culturale (ad es. la decisione di integrare più di frequente nuovo codice). Gli obiettivi principali dell'integrazione continua sono individuare e risolvere i bug con maggiore tempestività, migliorare la qualità del software e ridurre il tempo richiesto per convalidare e pubblicare nuovi aggiornamenti.  
Mediante l'integrazione continua, gli sviluppatori eseguono i commit in modo più frequente in un repository condiviso e impiegano un sistema di controllo della versione, ad esempio Git. Prima di applicare una modifica, possono eseguire unit test in locale, una verifica aggiuntiva importante prima dell'integrazione definitiva. Un servizio di integrazione continua crea automaticamente una build e avvia unit test sulle modifiche più recenti del codice per individuare immediatamente qualsiasi errore.  
Con la distribuzione continua, le modifiche al codice vengono applicate a una build, testate e preparate per il rilascio in produzione in modo automatico. La distribuzione continua estende l'integrazione continua distribuendo tutte le modifiche al codice all'ambiente di testing e/o di produzione dopo la fase di creazione di build.