Unità di apprendimento 1

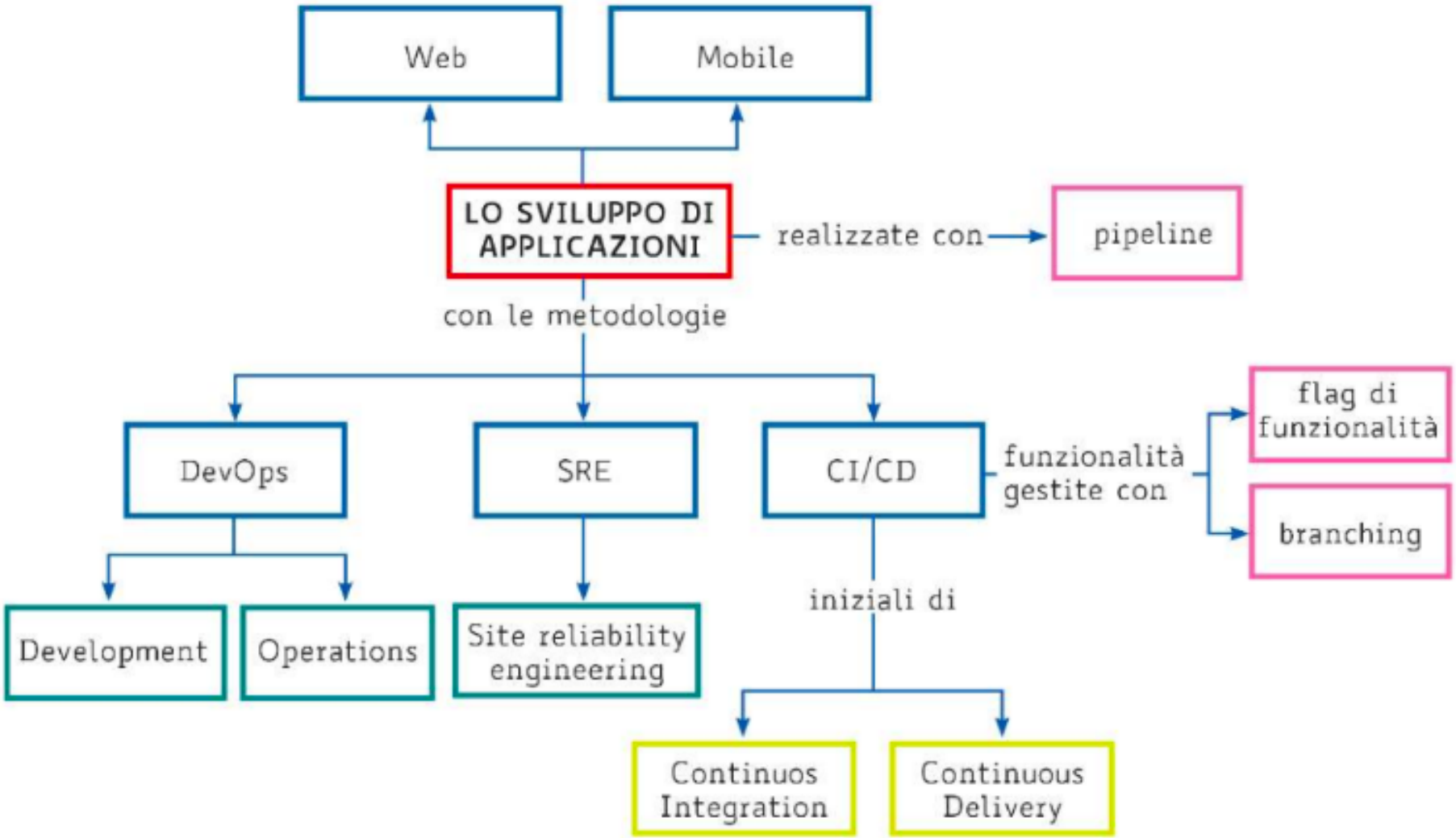
Architettura di rete e metodologia di sviluppo

Unità di apprendimento 1
Lezione 5

Lo sviluppo SW:
DevOps, SRE e CI/CD

In questa lezione impareremo:

- conoscere la metodologia DevOps
- elencare gli strumenti comuni utilizzati nell'ambito di DevOps
- definire il concetto di SRE
- conoscere la metodologia CI/CD


Web
Mobile
LO SVILUPPO DI APPLICAZIONI
realizzate con
pipeline
con le metodologie
DevOps
SRE
CI/CD
funzionalità gestite con
flag di funzionalità
branching
Development
Operations
Site reliability engineering
iniziali di
Continuos Integration
Continuous Delivery

La filosofia DevOps

- DevOps è una metodologia che promuove la collaborazione tra
  - team di sviluppo (*Development*)
  - operatori di sistemi (*Operations*, coloro che si occupano dell'IT, dell'infrastruttura)
- Il suo scopo è rimuovere le barriere tra queste due funzioni per:
  - migliorare la consegna del software in modo continuo, affidabile e veloce
  - consentire una produzione di software più veloce
  - garantire rilasci continui

La filosofia DevOps

- Non esistono regole codificate o manuali, ma solo linee guida
- Tutte le organizzazioni prevedono comunque:
  - la Continuous Integration (CI), che riguarda l'integrazione frequente dei cambiamenti/aggiornamenti del codice
  - il Continuous Development (CD), che riguarda il processo di rilascio automatico del software man mano che una nuova release è disponibile
  - inoltre anche monitoraggio e feedback costante delle applicazioni e dei sistemi di produzione sofware per raccogliere dati sulle performance, l'utilizzo del software e eventuali problemi

La filosofia DevOps

- Fasi e strumenti comuni nell'ambito di DevOps:
  - Plan, definizione degli obiettivi e dei requisiti (es. Trello)
  - Develop, sviluppo del codice (es. Visual Studio, Git)
  - Build, integrazione e creazione di una build (es. GitHub Actions)
  - Test, verifica della qualità del software (es. Selenium)
  - Release, preparazione per il rilascio (es. AWS CodeDeploy)
  - Deploy, distribuzione al pubblico (es. Kubernetes)
  - Operate, monitoraggio delle performance (es. Prometheus)
  - Report, raccolta e analisi dei dati d'utilizzo (es. Nagios)
  - Feedback, miglioramento basato sui riscontri

La collaborazione tra development e operations

- Il successo dell'interazione tra sviluppatori e operations si ottiene adottando pratiche collaborative e di condivisione di responsabilità
  - Si riescono ad identificare potenziali problemi in anticipo
  - Aiuta ad identificare e risolvere problemi più rapidamente
  - Migliora la qualità del software perché si agisce prima del rilascio all'utente
  - Si riduce il supporto post-vendita
- Non sempre questo approccio risulta di facile realizzazione per problemi relativi alla sicurezza, il rispetto di normative, best practice, standard

Site Reliability Engineering (SRE)

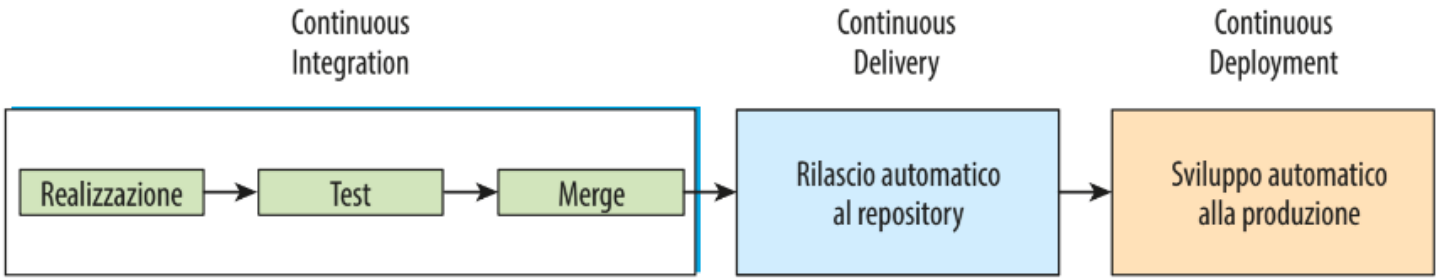
- La produzione veloce e i rilasci continui possono portare a downtime e conseguenti problemi di performance
- L'ingegneria dell'affidabilità del sito (SRE) mette l'accento sull'affidabilità dei sistemi informatici
- DevOps è una filosofia, SRE è una implementazione dei principi DevOps
- Si prefigge di garantire l'affidabilità dei servizi online, coniugandola con efficienza e scalabilità
- Funge da ponte tra development e operations occupandosi di affidabilità, automazione e monitoraggio, gestione dell'incidente, scalabilità e prestazioni, sicurezza

Introduzione alla CI/CD pipeline

- Per CI/CD si intende un insieme di pratiche che permettono il rilascio degli aggiornamenti del codice continuativamente e in modo sicuro
- Una pipeline consiste in una serie di passaggi che devono essere eseguiti per fornire una nuova versione del software
- Questa tecnica rientra nella metodologia Agile
- Ogni rilascio è costituito dallo sviluppo di piccoli cambiamenti
- Il codice viene versionato frequentemente in modo da integrare e validare i cambiamenti velocemente

Introduzione alla CI/CD pipeline

- CI (Continuous Integration) consente di riportare più facilmente e con maggiore frequenza le modifiche apportate al codice nell'applicazione che è stata rilasciata e che viene utilizzata dagli utenti
- CD (Continuous Delivery) consente di rilasciare correzioni e funzionalità rapidamente in ambienti e infrastrutture differenti



- È più facile identificare difetti e altri problemi di qualità del software su segmenti di codice più piccoli e appena sviluppati
- È meno probabile che più sviluppatori modifichino lo stesso codice

Integrazione continua

- In generale vi sono due modalità operative per la CI
  - rilascio periodico giornaliero, a determinate ore di ogni giorno viene effettuato il passaggio dal ramo di sviluppo a quello di master
  - rilascio in particolari giorni, possono comprendere modifiche fatte da più sviluppatori e quindi potrebbero verificarsi incompatibilità tra le modifiche fatte
- Successivamente al rilascio delle modifiche si passa alla convalida, con l'esecuzione automatica di batterie di test divisi per livelli

La distribuzione continua e/o il deployment continuo

- La CD assimila due concetti molto simili
  - La distribuzione continua, dove si automatizza il rilascio su un repository
  - Il deployment continuo, che è l'ultima fase di una pipeline CI/CD e si riferisce al rilascio automatico delle modifiche dal repository alla produzione
- Alla fine di queste due fasi l'applicazione aggiornata/corretta è accessibile e utilizzabile dai clienti

La gestione delle funzionalità

- Nella fase di CI si possono utilizzare due tecniche per l'integrazione delle funzionalità
  - il flag di funzionalità, per attivare o disattivare funzionalità del codice fino a quando non sono pronte all'uso
  - il branching del controllo di versione, dove si creano degli specifici branch basati sulle funzionalità e il nuovo codice viene unito ai branch standard una volta pronto all'uso

Il packaging e i test automatizzati

- Nella CI si automatizza anche il processo di generazione di un pacchetto (packaging) che contiene tutto il software realizzato, il database e gli altri componenti
- L'applicazione viene compilata, confezionata e configurata in un formato pronto per essere distribuito o installato in un ambiente di produzione
- Appositi tecnici sono incaricati del controllo di qualità
  - utilizzano framework automatizzati per definire ed eseguire vari tipi di test tra cui quello di regressione
  - eseguono l'analisi statica del codice e valutano le prestazioni
  - testano le API e la sicurezza

Riassumendo CI/CD

- CI/CD si ripete ogni volta che vengono apportate modifiche al codice

CI (CONTINUOUS INTEGRATION)	CD (CONTINUOUS DEPLOYMENT)
• Automazione dei processi di integrazione del codice • Integrazione frequente del codice nel repository condiviso • Esecuzione automatica dei test di integrazione • Rilevamento tempestivo di errori e conflitti nel codice • Garanzia di stabilità e qualità del codice prima del rilascio • Identificazione precoce di problemi di integrazione • Collaborazione continua tra membri del team di sviluppo • Riduzione del rischio di conflitti e problemi di integrazione del codice • Integrazione delle modifiche nel codice principale	• Automazione del rilascio e della distribuzione del software • Distribuzione continua del software ai server di produzione • Automazione dei processi di integrazione, configurazione e rilascio • Monitoraggio dell'infrastruttura e dei sistemi in produzione • Implementazione rapida e affidabile delle nuove versioni • Gestione del rollback e del ripristino in caso di errori • Feedback rapido e iterativo dagli utenti finali e correzioni • Miglioramento continuo del processo di sviluppo e distribuzione • Implementazione continua di piccole modifiche e aggiornamenti

SCELTA MULTIPLA

1. **Quale delle seguenti descrizioni è più adatta a DevOps?**
   a. Una metodologia di sviluppo del software.
   b. Un insieme di strumenti per l'automazione dei processi.
   c. Una filosofia che promuove la collaborazione tra sviluppatori e operatori di sistema.
   d. Un framework per la gestione dei progetti software.
2. **Qual è l'obiettivo principale di DevOps?**
   a. Migliorare l'efficienza delle operazioni IT.
   b. Ridurre i costi di sviluppo del software.
   c. Automatizzare completamente il processo di sviluppo del software.
   d. Favorire una consegna rapida e affidabile del software.
3. **Quali sono i principali benefici di DevOps?**
   a. Riduzione dei conflitti tra sviluppatori e operatori di sistema.
   b. Miglioramento della qualità del software.
   c. Aumento dei tempi di sviluppo del software.
   d. Limitazione delle responsabilità dei membri del team.
4. **Quali sono i pilastri fondamentali di DevOps?**
   a. Automazione, collaborazione e monitoraggio.
   b. Sviluppo, distribuzione e manutenzione.
   c. Sicurezza, scalabilità e flessibilità.
5. **Qual è l'obiettivo principale della CI?**
   a. Ridurre al minimo il codice sorgente.
   b. Automatizzare il processo di test.
   c. Garantire l'isolamento dei componenti software.
   d. Migliorare la comunicazione nel team.
6. **Come possono i team di sviluppo software beneficiare dell'implementazione di CI/CD?**
   a. Riducendo il numero di ore di lavoro.
   b. Aumentando la velocità di rilascio del software.
   c. Migliorando la qualità del codice.
   d. Semplificando la documentazione del progetto.
7. **Quali sono alcune delle principali pratiche associate alla Continuous Integration?**
   a. Code reviews e pair programming.
   b. Gestione dei requisiti e analisi dei bisogni.
   c. Manual testing e manual deployment.
   d. Versionamento del codice e automazione dei test.

VERO/FALSO

1.	DevOps si concentra solo sulla collaborazione tra sviluppatori e amministratori di sistema.	V	F
2.	DevOps promuove il rilascio frequente e continuo del software.	V	F
3.	DevOps si basa fortemente sull'automazione dei processi.	V	F
4.	DevOps promuove la cultura della responsabilità condivisa.	V	F
5.	DevOps si basa sull'approccio waterfall nello sviluppo del software.	V	F
6.	DevOps è solo un insieme di strumenti e tecnologie.	V	F
7.	CI si riferisce al processo di distribuzione continua del software.	V	F
8.	CD include la distribuzione continua del software ai server di produzione.	V	F
9.	CI/CD promuove il rilascio rapido e frequente del software.	V	F
10.	CI/CD riduce la necessità di eseguire test automatici sul software.	V	F
11.	CI/CD richiede l'uso di un sistema di controllo del codice sorgente come Git.	V	F
12.	CI/CD può contribuire a ridurre i rischi e gli errori associati al rilascio del software.	V	F
13.	CI/CD richiede una stretta collaborazione tra i team di sviluppo e operations.	V	F