MEGA Raccolta Quiz - Metodi e Tecnologie per lo Sviluppo Software

Corso di Bertazzo e Guzzo

---

1. ISSUE TRACKING SYSTEM (ITS)

Domande Teoriche

Q1.1: Definisci cos'è un Issue Tracking System e quali sono i suoi obiettivi principali.

Q1.2: Illustra il concetto di "workflow" in un ITS e spiega la sua importanza nella gestione dei work item.

Q1.3: Elenca e spiega almeno 5 funzionalità essenziali di un ITS moderno.

Q1.4: Cosa sono i "collegamenti" in un ITS e perché sono fondamentali per la tracciabilità?

Q1.5: Descrivi il processo di configurazione e utilizzo di un ITS, specificando i ruoli coinvolti.

Domande Pratiche

Q1.6: Un cliente riporta un bug critico. Descrivi il workflow completo che il work item dovrebbe attraversare dall'apertura alla risoluzione.

Q1.7: Come configureresti un ITS per un progetto Scrum? Quali tipi di work item definiresti?

Q1.8: Spiega come utilizzare board/bacheche per visualizzare lo stato di avanzamento del progetto.

Vantaggi e Benefici

Q1.9: Elenca i 7 principali vantaggi dell'utilizzo di un ITS nell'ambito dello sviluppo software.

Q1.10: Come un ITS può contribuire alla misurazione della qualità del progetto?

---

2. SOURCE CODE MANAGEMENT / VCS

Concetti Fondamentali

Q2.1: Definisci Source Code Management (SCM) e spiega i vantaggi rispetto alla gestione manuale del codice.

Q2.2: Illustra i diversi tipi di VCS e le loro caratteristiche distintive.

Q2.3: Confronta sistemi VCS centralizzati e distribuiti con esempi specifici.

Git Specifico

Q2.4: Descrivi le tre aree locali di Git e gli stati che un file può assumere.

Q2.5: Spiega il comando di configurazione base di Git per un nuovo sviluppatore.

Q2.6: Illustra i comandi Git essenziali per un workflow base (init, add, commit, push, pull).

Confronti e Workflow

Q2.7: Confronta Git e SVN evidenziando almeno 5 differenze significative.

Q2.8: Descrivi il GitFlow workflow e quando è appropriato utilizzarlo.

Q2.9: Spiega il Fork Workflow e i suoi vantaggi nei progetti open source.

Q2.10: Come si integra un VCS con un ITS per la tracciabilità delle modifiche?

---

3. FRAMEWORK SCRUM

Definizioni e Caratteristiche

Q3.1: Definisci Scrum e spiega perché è considerato un framework agile.

Q3.2: Elenca e descrivi le caratteristiche distintive del framework Scrum.

Q3.3: Spiega il concetto di "timeboxing" in Scrum e la sua importanza.

Ruoli

Q3.4: Descrivi dettagliatamente i tre ruoli principali in Scrum e le loro responsabilità.

Q3.5: Qual è la differenza tra Product Owner e Scrum Master? Possono essere la stessa persona?

Q3.6: Cosa significa che il Development Team è "auto-organizzato" e "cross-funzionale"?

Eventi

Q3.7: Illustra tutti gli eventi Scrum specificando durata, partecipanti e obiettivi.

Q3.8: Spiega la differenza tra Sprint Planning e Sprint Review.

Q3.9: Cos'è la Daily Scrum e quali sono le tre domande fondamentali?

Q3.10: Descrivi lo Sprint Retrospective e il suo valore per il miglioramento continuo.

Artefatti

Q3.11: Elenca e descrivi i tre artefatti principali di Scrum.

Q3.12: Cos'è il Product Backlog e chi ne è responsabile?

Q3.13: Spiega la differenza tra Product Backlog e Sprint Backlog.

Q3.14: Come si calcola e interpreta un burndown chart?

---

4. BUILD AUTOMATION E MAVEN

Concetti Generali

Q4.1: Definisci Build Automation e spiega la necessità storica di tali strumenti.

Q4.2: Illustra le caratteristiche CRISP dei build tools.

Q4.3: Confronta "scripting tools" e "artifact oriented tools" con esempi.

Q4.4: Cosa si intende per "processo di build" e quali elementi lo compongono?

Maven Specifico

Q4.5: Definisci Apache Maven e elenca le sue caratteristiche principali.

Q4.6: Spiega i tre Build Lifecycle di Maven e il loro scopo.

Q4.7: Descrivi le fasi del Default Build Lifecycle specificando l'ordine di esecuzione.

Q4.8: Cos'è il POM (Project Object Model) e quali informazioni contiene?

Q4.9: Spiega il concetto di "Convention over Configuration" in Maven.

Q4.10: Cosa sono gli Archetype in Maven e come si utilizzano?

Domande Pratiche Maven

Q4.11: Scrivi il comando Maven per creare un nuovo progetto Java usando quickstart archetype.

Q4.12: Cosa succede quando esegui `mvn install`? Elenca tutte le fasi eseguite.

Q4.13: Come si specificano le dipendenze in un progetto Maven?

Q4.14: Spiega il ruolo dei plugin Maven e cosa sono i "mojo".

---

5. SOFTWARE TESTING

Definizioni e Principi

Q5.1: Definisci Software Testing e i suoi obiettivi principali.

Q5.2: Illustra le categorie di testing (statico vs dinamico).

Q5.3: Elenca e spiega i "Sette Principi del Testing" secondo ISTQB.

Q5.4: Descrivi il V-Model e come si relaziona alle fasi di sviluppo.

Processo di Test

Q5.5: Illustra le fasi del processo di test dalla pianificazione alla chiusura.

Q5.6: Spiega la differenza tra errore (mistake), difetto (bug) e fallimento (failure).

Q5.7: Chi sono le figure principali che introducono errori nel software e in che percentuale?

Tipi di Test

Q5.8: Confronta test funzionali e non-funzionali con esempi.

Q5.9: Spiega la differenza tra test di unità, integrazione e sistema.

Q5.10: Cosa sono i test di non-regressione e quando vengono utilizzati?

---

6. UNIT TESTING

Definizioni e Concetti

Q6.1: Definisci Unit Testing specificando cosa si intende per "unità" nei diversi paradigmi.

Q6.2: Illustra le proprietà A-TRIP degli unit test.

Q6.3: Spiega perché gli unit test devono essere "Automatic" e cosa comporta.

Q6.4: Cosa significa che un test deve essere "Repeatable" e "Independent"?

Q6.5: Perché gli unit test devono essere scritti con approccio "Professional"?

Framework e Strumenti

Q6.6: Elenca i componenti necessari per un framework di unit testing.

Q6.7: Cosa distingue JUnit 4 come framework per unit testing?

Q6.8: Spiega il concetto di Mock Object e quando utilizzarli.

Tecniche di Verifica

Q6.9: Illustra la tecnica "Check inverse relationship" con esempi pratici.

Q6.10: Spiega "Cross-check Using Other Means" (uso di oracoli).

Q6.11: Cosa significa "Force error conditions" nei test di unità?

Q6.12: Quali sono le caratteristiche di performance desiderabili per gli unit test?

---

7. TEST DRIVEN DEVELOPMENT (TDD)

Concetti Fondamentali

Q7.1: Definisci Test Driven Development e il suo approccio metodologico.

Q7.2: Illustra il ciclo TDD nelle tre fasi (Rosso-Verde-Grigio).

Q7.3: Spiega cosa avviene nella "fase rossa" del ciclo TDD.

Q7.4: Descrivi gli obiettivi della "fase verde" e cosa significa "quantità minima di codice".

Q7.5: Cos'è il refactoring nella "fase grigia" e perché è fondamentale?

Vantaggi e Applicazione

Q7.6: Elenca i principali vantaggi dell'approccio TDD.

Q7.7: Come TDD influenza la progettazione del software?

Q7.8: Quali sfide può presentare l'adozione di TDD in un team?

Q7.9: In quali contesti TDD è più efficace?

Q7.10: Come si integra TDD con altri processi di sviluppo agile?

---

8. ANALISI STATICA DEL CODICE

Definizioni e Obiettivi

Q8.1: Definisci analisi statica del codice e confrontala con l'analisi dinamica.

Q8.2: Spiega la "Teoria delle finestre rotte" nel contesto dell'analisi statica.

Q8.3: Quali tipi di problemi può identificare l'analisi statica?

Q8.4: Come si integra l'analisi statica nel processo di sviluppo?

Strumenti e Implementazione

Q8.5: Elenca esempi di strumenti per l'analisi statica del codice.

Q8.6: Come si configurano i quality gate basati su analisi statica?

Q8.7: Quale ruolo gioca l'analisi statica nella Continuous Integration?

Q8.8: Come gestire i warning e le metriche di qualità del codice?

---

9. CONTINUOUS INTEGRATION (CI)

Concetti Fondamentali

Q9.1: Definisci Continuous Integration e i suoi principi base.

Q9.2: Illustra il processo CI dal commit all'artifact repository.

Q9.3: Spiega i 7 passi del processo di CI dallo sviluppatore.

Q9.4: Perché l'integrazione frequente è fondamentale nel CI?

Implementazione e Best Practices

Q9.5: Quali test possono essere eseguiti nel processo di CI?

Q9.6: Come deve essere strutturato un processo di build per CI?

Q9.7: Cosa comporta un processo di build lento nel CI?

Q9.8: Come si gestiscono le notifiche e il feedback nel CI?

Strumenti e Automazione

Q9.9: Illustra il ruolo di Jenkins nel processo di CI/CD.

Q9.10: Come si integrano VCS, build tools e testing nel pipeline CI?

Q9.11: Spiega l'uso di Docker nel contesto CI/CD.

Q9.12: Cosa sono i quality gate e come si implementano?

---

10. ARTIFACT REPOSITORY

Concetti e Definizioni

Q10.1: Definisci Artifact Repository e il suo ruolo nell'ecosistema DevOps.

Q10.2: Spiega la differenza tra repository locale, centrale e remoto in Maven.

Q10.3: Quali tipi di artefatti vengono gestiti in un repository?

Q10.4: Come si integra l'artifact repository con il processo di CI/CD?

Gestione e Configurazione

Q10.5: Illustra la configurazione di Nexus OSS come artifact repository.

Q10.6: Come si gestiscono versioni e dipendenze negli artifact repository?

Q10.7: Spiega le politiche di retention e cleanup degli artefatti.

Q10.8: Come si implementa la sicurezza negli artifact repository?

---

11. DOMANDE INTEGRATE E CASI PRATICI

Scenari Complessi

Q11.1: Descrivi un workflow completo dal commit del codice al deploy in produzione includendo tutti gli strumenti del corso.

Q11.2: Come implementeresti un processo DevOps completo per un progetto Java utilizzando Git, Maven, JUnit, SonarQube e Jenkins?

Q11.3: Progetta l'integrazione tra Jira (ITS), BitBucket (VCS) e Jenkins (CI) per un team Scrum.

Q11.4: Illustra come gestire la qualità del codice attraverso analisi statica, unit testing e CI.

Confronti e Scelte Tecnologiche

Q11.5: Confronta diversi approcci di branching (GitFlow vs GitHub Flow) per progetti diversi.

Q11.6: Quando scegliere tra diversi build tools (Maven vs Gradle vs Ant)?

Q11.7: Come valutare l'efficacia di un processo CI/CD implementato?

Q11.8: Progetta una strategia di testing completa che integri unit, integration e system testing.

Problem Solving

Q11.9: Un team lamenta build lente nel CI. Analizza le possibili cause e proponi soluzioni.

Q11.10: Come gestire dependency conflicts in un progetto Maven multimodulo?

Q11.11: Implementa una strategia di rollback per deployment falliti.

Q11.12: Come misurare e migliorare la code coverage in un progetto esistente?

---

12. QUIZ A RISPOSTA MULTIPLA

ITS e Project Management

Q12.1: Un workflow in un ITS rappresenta: a) Una sequenza di attività da completare b) Un insieme di stati e transizioni di un work item c) Un tipo di report automatico d) Una configurazione di permessi utente

Q12.2: Il principale vantaggio dei collegamenti bidirezionali in un ITS è: a) Velocizzare le query di ricerca b) Ridurre lo spazio di storage c) Garantire la tracciabilità tra work item d) Automatizzare le notifiche

VCS e Git

Q12.3: In Git, l'area "staging" serve per: a) Memorizzare i file modificati b) Preparare i file per il commit c) Sincronizzare con il repository remoto d) Creare backup automatici

Q12.4: Il comando `git merge` vs `git rebase`: a) Sono identici in funzionalità b) Merge preserva la cronologia, rebase la riscrive c) Rebase è più sicuro di merge d) Merge è deprecato in favore di rebase

Build Automation

Q12.5: In Maven, il comando `mvn clean install` esegue: a) Solo la pulizia e compilazione b) Tutte le fasi fino a install del default lifecycle c) Solo i test di unità d) Solo il packaging dell'artefatto

Q12.6: La convenzione "Convention over Configuration" significa: a) Evitare file di configurazione b) Usare configurazioni predefinite quando possibile c) Configurare tutto manualmente d) Utilizzare solo convenzioni di naming

Testing

Q12.7: Un test di unità "Independent" significa: a) Non richiede framework specifici b) Può essere eseguito senza connessione di rete c) Il risultato non dipende da altri test d) Non utilizza librerie esterne

Q12.8: Nel TDD, la "fase rossa" prevede: a) Scrivere codice che fallisce b) Scrivere un test che fallisce c) Correggere bug esistenti d) Eseguire analisi statica

CI/CD

Q12.9: Il principio fondamentale della Continuous Integration è: a) Deploy automatico in produzione b) Integrazione frequente del codice c) Testing automatizzato completo d) Monitoraggio in tempo reale

Q12.10: Un quality gate nel CI può bloccare la build se: a) Il codice non compila b) I test falliscono c) Le metriche di qualità non sono rispettate d) Tutte le precedenti

---

RISPOSTE RAPIDE - SEZIONE QUIZ MULTIPLA

Q12.1: b) Un insieme di stati e transizioni di un work item **Q12.2:** c) Garantire la tracciabilità tra work item
Q12.3: b) Preparare i file per il commit **Q12.4:** b) Merge preserva la cronologia, rebase la riscrive **Q12.5:** b) Tutte le fasi fino a install del default lifecycle **Q12.6:** b) Usare configurazioni predefinite quando possibile **Q12.7:** c) Il risultato non dipende da altri test **Q12.8:** b) Scrivere un test che fallisce **Q12.9:** b) Integrazione frequente del codice **Q12.10:** d) Tutte le precedenti

---

NOTE PER LO STUDIO

Argomenti Cruciali da Ripassare:

1. **Proprietà A-TRIP degli unit test** - Fondamentale
2. **Ciclo TDD (Rosso-Verde-Grigio)** - Spesso richiesto
3. **Build Lifecycle di Maven** - Ordine delle fasi
4. **Ruoli e eventi Scrum** - Definizioni precise
5. **Principi della Continuous Integration** - 7 passi del processo
6. **Workflow Git vs SVN** - Differenze architetturali

Integrazione tra Strumenti:

- Comprendere come ITS, VCS, Build Tools e CI/CD si integrano
- Saper progettare un workflow completo DevOps
- Conoscere i vantaggi di ogni strumento nel contesto generale

Casi Pratici Frequenti:

- Configurazione di un progetto Maven
- Implementazione di unit test con JUnit
- Setup di un processo CI con Jenkins
- Gestione di branch e merge conflicts in Git