Fondamentali di architetture software

L'architettura combina stili, caratteristiche, decisioni e principi per garantire disponibilità e scalabilità.
L'architetto software deve guidare le scelte tecnologiche, mantenendo la sinergia con i team di sviluppo e comprendendo il dominio aziendale.

- L'architetto software deve avere una vasta conoscenza tecnologica e rimanere aggiornato sulle ultime tendenze.
- È fondamentale che l'architetto verifichi che i team di sviluppo seguano le decisioni architettoniche per garantire il corretto funzionamento del sistema.
- Le competenze interpersonali e la capacità di navigare nella politica aziendale sono essenziali per ottenere approvazioni e supporto.
- L'architetto deve bilanciare attività architettoniche con la programmazione, dedicandosi a prove di concetto e revisione del codice.
  Caratteristiche architetturali:
  Questo testo discute le caratteristiche non funzionali dell'architettura software e la loro importanza nel design.
- Le caratteristiche architetturali non funzionali influenzano il design e sono cruciali per il successo dell'applicazione.
- È importante selezionare solo le tre caratteristiche più rilevanti con i portatori di interesse.
- Le caratteristiche includono disponibilità, performance, scalabilità, sicurezza e usabilità, tra le altre.
- Ogni caratteristica richiede sforzi di design e può avere impatti negativi su altre.
- L'approccio migliore è progettare un'architettura iterativa e facilmente modificabile.
  Measuring:
  È fondamentale stabilire un linguaggio comune per le caratteristiche architetturali, con metriche come la complessità cicomatica, dove un valore sotto 10 è accettabile, preferibilmente sotto 5. La copertura del codice è un'altra metrica importante per la deployabilità. Le funzioni di fitness possono garantire il rispetto dei principi architetturali.
  Modularity:
  La sezione discute la modularità, la coesione, il coupling e le metriche per valutare la stabilità del codice.
- La modularità è un principio organizzativo che aiuta a gestire sistemi complessi attraverso gruppi logici di codice.
- La coesione è misurata tramite il LCOM, che valuta la mancanza di coesione strutturale nei metodi.

- Il coupling inter-modulare è analizzato attraverso metriche come instabilità, astrazione e distanza dalla sequenza principale.
- L'instabilità è calcolata come il rapporto di coupling efferente rispetto alla somma di coupling afferente ed efferente.
- Le forme di connascenza variano in forza, località e grado, influenzando la facilità di refactoring e l'impatto sul codice.
- Connascenza più debole si verifica quando le connessioni tra i moduli sono più remote, migliorando la manutenibilità del codice.
  Componenti: I componenti sono l'unità fondamentale nell'architettura software, con un design che deve bilanciare la granularità per evitare eccessiva comunicazione o accoppiamento interno. Tecniche come l'approccio attore/azioni e l'event storming aiutano nell'identificazione dei componenti. La progettazione efficace dei componenti è cruciale per la modularità e la testabilità.