SWE- Raccolta di domande e risposte

- **Fornire una breve definizione del cosiddetto ciclo di Deming (anche noto come PDCA) e discutere concisamente se e come i suoi principi siano stati applicati nel proprio progetto didattico.**

  Il ciclo di Deming (o ciclo di PDCA, acronimo dall'inglese Plan–Do–Check–Act) è un metodo di gestione iterativo in quattro fasi utilizzato per il controllo e il miglioramento continuo dei processi e dei prodotti. Questo strumento parte dall'assunto che per il raggiungimento del massimo della qualità sia necessaria la costante interazione tra ricerca, progettazione, test, produzione e vendita. La sequenza logica dei quattro punti ripetuti per un miglioramento continuo è la seguente:

  - P - Plan. Pianificazione: stabilire gli obiettivi e i processi necessari per fornire risultati in accordo con quanto atteso
  - D - Do. Esecuzione del programma, dapprima in contesti circoscritti. Attuare il piano, eseguire il processo, creare il prodotto. Raccogliere i dati per la creazione di grafici e analisi da destinare alla fase di "Check" e "Act".
  - C - Check. Test e controllo, studio e raccolta dei risultati e dei riscontri. Studiare i risultati, misurati e raccolti nella fase del "Do" confrontandoli con i risultati attesi, obiettivi del "Plan", per verificarne le eventuali differenze. Cercare le deviazioni nell'attuazione del piano e focalizzarsi sulla sua adeguatezza e completezza per consentirne l'esecuzione. I grafici dei dati possono rendere questo molto più facile, in quanto è possibile vedere le tendenze di più cicli PDCA, convertendo i dati raccolti in informazioni.
  - A - Act. Azione per rendere definitivo e/o migliorare il processo. Richiede azioni correttive sulle differenze significative tra i risultati effettivi e previsti. Analizza le differenze per determinarne le cause e dove applicare le modifiche per ottenere il miglioramento del processo o del prodotto.

  Il ciclo di Deming è stato applicato all'interno del progetto suddividendo internamente ogni macro periodo definito nel PdP in n sotto-periodi (con n solitamente uguale a 3, di durata media 7 giorni) corrispondenti alle iterazioni del ciclo. Ogni sotto-periodo è a sua volta contraddistinto dalle quattro fasi caratteristiche del ciclo, e si conclude con una riunione riepilogativa che evidenzia problemi emersi e traguardi raggiunti. Alla conclusione di ogni sotto periodo il livello di qualità aumenta incrementalmente sino alla baseline definita dalla conclusione del macro periodo corrispondente.

- **Alla luce dell'esperienza acquisita nel proprio progetto didattico, discutere la ripartizione percentuale di impegno complessivo effettivo dedicato, a livello di gruppo, alle principali attività svolte. Ipotizzare poi la ripartizione ideale rispetto al problema affrontato.**

  Dall'evidenza dei consuntivi emerge che i primi periodi di progetto hanno richiesto un'ingente quantità di ore dedicate ai ruoli di Analista, Amministratore e Responsabile, come prevedibile data la natura dei periodi e l'inesperienza dei membri del gruppo. Nei periodi successivi si sono invece rivelati cruciali i ruoli di Progettista, Programmatore e Verificatore, in particolare l'importanza del primo è stata tendenzialmente sottostimata a favore di quella di Programmatore e Verificatore. In linea di massima, i succitati ruoli relativi alla gestione e all'analisi hanno impegnato all'incirca il 65% del tempo del gruppo, con un 35% dedicato agli altri ruoli. A posteriori, tale 65% si sarebbe potuto ridurre leggermente previo maggiore studio personale dei singoli membri a monte della

realizzazione dei documenti di progetto, dando così maggior respiro ai progettisti. A cascata, ciò avrebbe avuto ripercussioni senz'altro positive sulla codifica e sulla verifica, rese più accurate e precise dal maggior tempo a disposizione.

- **Fornire una definizione di architettura applicabile al dominio dell'ingegneria del software. Discutere poi concisamente se e come tale nozione sia stata utilizzata nel progetto didattico.**

  Architettura - da ISO/IEC/IEEE 42010:
  - *La decomposizione del sistema in componenti*
  - *L'organizzazione di tali componenti, definendo ruoli, responsabilità e iterazioni (chi fa cosa e come)*
  - *Le interfacce (regole strutturate) necessarie all'interazione delle componenti*
  - *I paradigmi (modo esemplare) di composizione, per fare il meglio con meno risorse possibili*

  Tale definizione è stata utilizzata nel periodo di progettazione e codifica nel seguente modo:

  a. Partendo dall'esito di un approfondita analisi delle tecnologie, si è innanzitutto cercato di individuare le componenti macroscopiche del prodotto, raffinando man mano la loro definizione.
  b. Contestualmente, si è individuato il ruolo e la responsabilità di ogni componente, in sé stessa e in relazione con le altre.
  c. Individuate e responsabilizzate le componenti, si è cercato di comporle attraverso adeguati design pattern che garantissero al prodotto e alla sua architettura determinate qualità.

- **Fornire una definizione del concetto di qualità, applicabile al dominio dell'ingegneria del software. Discutere concisamente quali attività il proprio gruppo di progetto didattico abbia svolto nella direzione di tale definizione, e valutarne criticamente l'esito rilevato.**

  Qualità - da ISO 9000:
  Insieme delle caratteristiche di una entità che ne determinano la capacità di soddisfare esigenze espresse ed implicite, deve essere quantificabile per permettere paragoni oggettivi.

  Durante il corso del progetto, la qualità è stata perseguita definendo obiettivi ad essa legati, monitorati mediante misurazioni metriche pertinenti e raggiunti tramite apposite strategie. L'attività di quality assurance ha accertato che il grado di conseguimento degli obiettivi fosse in linea con quanto previsto, e l'impianto amministrativo, le procedure e gli strumenti automatici dedicati hanno concretizzato la politica di qualità stabilita dal gruppo. Sotto uno sguardo critico, lo svolgimento di tali attività è stato tuttavia superficiale e laconico, producendo falle nel sistema di attuazione della qualità cui ha conseguito il mancato raggiungimento di alcuni obiettivi proposti. Tale problema si sarebbe probabilmente potuto mitigare con una migliore attività di formazione del personale.

- **Illustrare concisamente la strategia di verifica tramite test adottata nel progetto didattico (quali tipi di test, quali obiettivi, quale grado di automazione, ecc.). Alla luce dei risultati ottenuti nel progetto da tali attività, in termini di rapporto costi/benefici, discutere gli spazi di miglioramento rilevati e le eventuali eccellenze raggiunte.**

  Nel corso del progetto è stata instaurata dal gruppo una strategia di testing che prevedeva quattro tipologie di test:

- Test di unità
- Test di integrazione
- Test di sistema
- Test di validazione

Ad ogni test viene associato un codice che lo identifica univocamente e la sua esecuzione avviene ad ogni commit sul repository tramite l'integrazione continua offerta dal servizio Travis CI, cui consegue la rilevazione di valori metrici riassunti in apposite dashboard (tramite servizio SonarCloud) e nella documentazione di progetto. All'attività di testing del codice era associato uno specifico obiettivo di qualità, a sua volta associato ad una specifica strategia di conseguimento, che stabiliva l'obbligo di raggiungimento di un esito positivo per ogni test sostenuto. Il sistema instaurato ha permesso il conseguimento di tale obiettivo e il raccoglimento di dati metrici che hanno permesso di migliorare il software anche in ambiti che esulino dai meri test. Spazio di miglioramento è probabilmente ampio sul numero di metriche rilevate contestualmente all'esecuzione dei test, che, se in numero maggiore, permetterebbero di evidenziare ulteriori criticità sul software. È stato invece soddisfacente il grado di automazione generale e la semplicità di generazione dei test.

- **Fornire una definizione del formalismo noto come "diagramma di Gantt", discuterne concisamente le finalità e modalità d'uso, l'efficacia e i punti deboli eventualmente rilevati nell'esperienza del progetto didattico.**

  Il diagramma di Gantt, usato principalmente nelle attività di project management, è costruito partendo da un asse orizzontale, a rappresentazione dell'arco temporale totale del progetto, suddiviso in fasi incrementali (ad esempio, giorni, settimane, mesi), e da un asse verticale, a rappresentazione delle mansioni o attività che costituiscono il progetto. Delle barre orizzontali di lunghezza variabile rappresentano le sequenze, la durata e l'arco temporale di ogni singola attività del progetto. Queste barre possono sovrapporsi durante il medesimo arco temporale ad indicare la possibilità dello svolgimento in parallelo di alcune delle attività. Man mano che il progetto progredisce, delle barre secondarie, delle frecce o delle barre colorate possono essere aggiunte al diagramma, per indicare le attività sottostanti completate o una porzione completata di queste. Una linea verticale è utilizzata per indicare la data di riferimento. Un diagramma di Gantt permette dunque la rappresentazione grafica di un calendario di attività, utile al fine di pianificare, coordinare e tracciare specifiche attività in un progetto dando una chiara illustrazione dello stato d'avanzamento del progetto rappresentato; di contro, uno degli aspetti non tenuti in considerazione in questo tipo di diagrammazione è l'interdipendenza delle attività.

- **Alla luce dell'esperienza acquisita nel proprio progetto didattico, discutere concisamente quali strategie di gestione di progetto hanno portato benefici e quali invece – attuate poco o male, o non tempestivamente – hanno causato problemi e difficoltà**.

  Nel corso del progetto si è senz'altro rivelata efficace la strategia di gestione delle task tramite ticketing, che ha permesso una distribuzione del carico di lavoro efficiente, omogenea e monitorabile. Altra strategia vincente è probabilmente stata quella di gestione dei rischi, che ha permesso di tener conto di possibili calamità e di rispondervi prontamente tramite apposite contromisure. La stessa, se attuata prima di quanto fatto, avrebbe senz'altro avuto un impatto ancor più positivo sull'esito del progetto. Una strategia migliorabile nell'implementazione è invece quella di gestione delle comunicazioni, le cui direttive non sono sempre state seguite causando ritardi rilavanti.

- **Facendo riferimento allo standard ISO/IEC 12207, discutere la differenza di obiettivi, tecniche e strategie di conduzione tra i processi di verifica e validazione.**

  Verifica: processo di supporto, accerta che l'esecuzione delle varie attività non abbia introdotto errori. Per effettuare la verifica è necessario un piano di verifica basato sul way of working.

  Risponde alla domanda: stiamo sviluppando correttamente il software?

  Validazione: controllo rivolto solo al prodotto finale, lungo e costoso, per accertarsi che esso corrisponda alle attese. Abilitato dalle verifiche, accerta che il codice confermi i requisiti.

  Risponde alla domanda: stiamo sviluppando il software corretto?

  Il processo di verifica è dunque orientato alla rilevazione ed eventuale correzione di errori, evidenziati tramite tecniche di analisi statica/dinamica dei prodotti, mentre la validazione accerta la conformità di determinate caratteristiche del prodotto con quanto contrattato con il cliente o stabilito internamente. La verifica viene attuata ripetutamente all'interno di uno stesso periodo di progetto, solitamente al termine di ogni attività, così da rilevare e correggere gli errori da essa introdotti prima che impattino a catena su altre attività dipendenti o meno. La validazione, invece, avviene contestualmente all'approvazione del prodotto come accertamento della bontà delle sue caratteristiche e della corretta esecuzione dell'attività di verifica. Nell'ambito del codice, l'attività di verifica si occupa di eseguire test di unità, integrazione e sistema, che accertano la correttezza di quanto prodotto, mentre quella di validazione effettua i test omonimi per permetterne l'approvazione.

- **Presentare concisamente almeno due metriche utilizzabili per la misurazione di qualità della progettazione software e del codice. Specificare quale tra esse sia stata utilizzata nel proprio progetto didattico e con quale esito.**

  - Grado di instabilità: Il livello di stabilità di un software indica il rischio con una modifica apportata ad un package influenzi il funzionamento dell'intero sistema. La metrica si calcola secondo la seguente formula:

    $$Grado\ di\ instabilità = \frac{Accoppiamento\ efferente}{Accoppiamento\ efferente + Accoppiamento\ afferente}$$

    Dove:
    - Accoppiamento efferente: indica il grado di dipendenza delle classi di un package verso classi di package esterni;
    - Accoppiamento afferente: indica il grado di utilità delle classi di un package verso classi esterne ad esso.

    Il perseguimento di un valore adeguato di tale metrica permette di garantire un certo livello di manutenibilità del software, le cui componenti risulteranno poco accoppiate e quindi semplici da modificare senza impattare sul resto del sistema. Nell'ambito del progetto, questa metrica è stata rilevata mediante strumenti automatici calati in un contesto di integrazione continua, ed ha appunto permesso di ottenere un software le cui componenti siano quanto più possibile manutenibili.

  - Copertura del codice: indica la percentuale di istruzioni, rispetto al totale, che vengono eseguite durante i test. Maggiore è il numero delle istruzioni testate e maggiore è la possibilità di individuare e risolvere errori di codifica. Un valore troppo basso indica che

molte istruzioni non vengono testate, divenendo possibile causa di anomalie. La metrica si calcola con la seguente formula:

$$Code\ coverage = \frac{linee\ di\ codice\ testate}{linee\ di\ codice\ totali}$$

Nell'ambito del progetto, questa metrica è stata rilevata mediante strumenti automatici calati in un contesto di integrazione continua, ed ha permesso di garantire che almeno l'85% del codice risultasse implicato in uno o più test. Si noti che la copertura del codice non garantisce la bontà del codice cui si riferisce, bensì il fatto che esso sia stato testato ed analizzato facendo dunque emergere possibili errori.

- **Descrivere la tecnica di classificazione e tracciamento dei requisiti adottata nel proprio progetto didattico e discuterne l'efficacia e i limiti riscontrati. Evidenziare inoltre quale relazione intercorre tra tracciamento dei requisiti e validazione.**

  Nel contesto del progetto didattico, i requisiti sono stati classificati tramite codice univoco seguente la struttura:

  R[Priorità (ex: O – Obbligatorio)][Tipologia (ex: F – Funzionale)][Numero incrementale]

  Apposite tabelle hanno poi permesso di associare ogni requisito alle proprie fonti e ai casi d'uso corrispondenti. La piattaforma online SWEgo ha permesso di aggiornare e monitorare i requisiti, nonché verificarne agevolmente la copertura. Con queste premesse, non sono emersi particolari problemi nel tracciamento dei requisiti in sé, quanto più nel livello di dettaglio degli stessi che talvolta rendeva ambigua la conferma del loro soddisfacimento. Il processo di validazione ha dunque il compito di accertare la coerenza tra prodotto realizzato e requisiti da esso soddisfatto, così da garantire che lo stesso incontri le esigenze evidenziate dal cliente.

- **Fissando l'attenzione sulla definizione di processo associata allo standard ISO/IEC 12207, indicare come (secondo quali regole), quando (in quali fasi di progetto) e perché (attraverso quali attività) la vostra esperienza di progetto didattico ha visto attuato tale concetto.**

  Processo: un insieme di attività interdipendenti, che trasformano un input in un output

  Nel corso del progetto didattico, si è innanzitutto cercato di determinare i processi necessari allo sviluppo del prodotto attenendosi fedelmente allo standard ISO/IEC 12207. Con un approccio top-down e il riferimento allo standard, si è poi passati alla definizione generica delle attività costituenti ogni processo per poi contestualizzarla e dettagliarla sempre più con il progredire del progetto. Ci si è dunque assicurati che all'istanziazione di ogni processo questo fosse normato e ben definito rispetto alle attività che lo compongono e alle sue pre e post condizioni. Da un punto di vista pratico, questo approccio ha permesso di lavorare secondo una migliore suddivisione in task, per esempio individuandone una sequenzialità o interdipendenza in termini di pre e post condizioni, nonché di migliorare progressivamente la qualità dei processi una volta concluse le relative attività e verificati i relativi prodotti.

- **Spiegare concisamente (dunque a livello di sostanza) la differenza tra il modello di sviluppo iterativo e quello incrementale. Alla luce dell'esperienza acquisita nel progetto didattico, indicare – spiegando a posteriori – quale dei due sarebbe stato più adatto al caso.**

Un'iterazione consiste nella ripetizione di un dato insieme di attività fino a che queste non convergono ad un dato obiettivo, rimandando alla fine l'integrazione delle componenti sviluppate. Il modello iterativo segue un approccio adattivo rispetto alla variabilità dei requisiti e agli imprevisti, come quello incrementale, ma comporta un grave rischio di non convergenza. Il modello incrementale, invece, prevede appunto incrementi che affrontino di volta in volta tutte le fasi del modello sequenziale, nonché rilasci multipli associati ad un costante incremento ed integrazione di funzionalità. Un grande vantaggio offerto è rappresentato dal fatto che le funzionalità critiche vengono trattate per prime, subendo così una ripetuta verifica.
Valutando il progetto a posteriori, il modello più adatto al caso è probabilmente quello incrementale, che grazie al suo focus sulla verifica delle funzionalità critiche dà maggiori garanzia di corretta implementazione delle stesse. La scelta del modello incrementale permette inoltre di ripetere più e più volte varie fasi del progetto, consentendo ad un team inesperto di impratichirsi maggiormente con le attività ad esse legate.

- **Un elemento delle strategie di pianificazione di progetto concerne la ripartizione percentuale (quindi non la quantità ma la proporzione) delle risorse umane (ore/persona) disponibili sulle attività da svolgere. Discutere i criteri che avete utilizzato al riguardo nel progetto didattico, presentare le scelte fatte e valutarle criticamente alla luce di quanto appreso allo stato.**

  Per questioni che esulano da un contesto lavorativo reale, si è deciso di ripartire le ore lavorative equamente rispetto ai vari componenti del gruppo, sia in totale che per periodo. Tale scelta ha impattato negativamente sulla realizzazione, in termini di qualità e non, di alcuni task (costretti per esempio ad essere associati a più o meno persone del necessario), garantendo allo stesso tempo il compimento di ciascuno di essi venendo incontro alle esigenze personali dei membri del team (in termini di studio universitario, lavoro, ecc…), nonché che ogni membro potesse fare esperienza allo stesso modo dei vari ruoli. In un contesto lavorativo reale, o semplicemente in un team legato a meno vincoli temporali e di competenze, una ripartizione equa potrebbe tuttavia non essere una scelta vincente, sostituibile con una strategia pianificata in base agli obiettivi di qualità definiti e che mitighi le lacune in termini di conoscenze/competenze con soluzioni quali task collaborativi (che integrino formazione e compimento del task stesso) o affidati sulla base di una stima delle competenze individuali.

- **Indicare, giustificandolo, il peso percentuale di impegno che ritenete sia da dedicare alla verifica in un contesto di lavoro paragonabile al vostro progetto didattico. Ripartire tale quantità proporzionalmente tra le specifiche attività di verifica attraverso le varie fasi dello sviluppo. Indicare quali di tali attività possano o debbano essere automatizzate, e come.**

  Alla luce dell'esperienza di progetto, ritengo che una quantità proporzionata di tempo da dedicare alla verifica sia circa il 30% del totale a disposizione. Tale tempo è da distribuirsi nel seguente modo:
  a) 20% nell'instaurazione e pianificazione (anche in termini di studio di fattibilità e necessità) di un buon impianto di verifica.
  b) 15% nell'attuazione della verifica dei requisiti, cosicché si garantisca un maggiore agio nel corso del processo di validazione.
  c) 15% nell'attuazione della verifica dei processi, così da garantire la bontà del lavoro svolto e l'ottimizzazione dell'utilizzo delle risorse.
  d) 20% nell'attuazione della verifica della progettazione, così da evitare spiacevoli sorprese in fase di codifica.

e) 20% nell'attuazione della verifica della codifica e nella progettazione e implementazione dei relativi test, così da garantirne la conformità agli obiettivi definiti.
f) 10% nell'attuazione della verifica della documentazione di progetto, per assicurare che possa rappresentare un punto di riferimento rispetto a problemi inerenti l'operatività del gruppo.

Premesso che ognuna delle succitate attività sarebbe avvalorata da un certo grado di automazione, quelle che è possibile automatizzare senza sforzo eccessivo (rendendo ciò quasi un obbligo) sono la verifica di parte della progettazione, del software e della documentazione. Tale automazione viene implementata mediante un set di strumenti in grado di rilevare (e ad un certo livello interpretare) valori metrici relativi ai propri input, restituendo importanti informazioni relative allo stato di un prodotto di processo. Queste informazioni vengono poi interpretare dai Verificatori, che sulla loro base segnaleranno la necessità di azioni correttive e/o migliorative.

- **Durante lo svolgimento del progetto didattico avete sovente confuso la nozione di fase di progetto con quella di attività di progetto (o, meglio ancora, di processo software, inteso come aggregato di attività coordinate e coese). Provate qui a descrivere compiutamente ciascuna delle due nozioni così da evidenziare le differenze e le relazioni tra esse.**

  - *Fase: la durata temporale entro uno stato del ciclo di vita o in una transizione tra essi, durante la quale si svolgono specifiche e univoche attività.*
  - *Attività: insieme di procedure per compiere un compito specifico relativo ad un determinato processo. Vengono svolte tramite tecniche applicate agli strumenti a disposizione.*

  Dalle due definizioni emerge come una fase contenga delle attività, che rappresentano la modalità di raggiungimento di un determinato obiettivo. Mentre la fase è un arco temporale ben definito, l'attività, che vive all'interno di un processo, specifica un'operatività nella direzione degli obiettivi di progetto.

- **Elencare quali attività di progetto software ritenete siano utilmente automatizzabili, in tutto e in parte, provando anche a rapportare i vantaggi che ne deriverebbero con i costi che l'ottenimento di tale automazione potrebbe comportare.**

  Premesso che ogni tipo di attività non creativa (in particolare se estremamente ripetitiva) gioverebbe di un certo grado di automazione, le attività che probabilmente ne trarrebbero i maggiori vantaggi sono quelle legate alla verifica, alla validazione, a parte dello sviluppo e della gestione di progetto. Nello specifico, le attività di verifica e validazione possono e devono godere di un elevato grado di automazione, raggiungibile con un impiego di risorse (economiche e temporali) generante vantaggi tali da permettere di recuperare l'investimento in breve tempo. Allo stesso modo, l'attività di codifica può avvalersi di strumenti appropriati che permettano l'automazione della scrittura di parti di codice (benché forse tali strumenti siano in parte ancora immaturi) o di documentazione, e la gestione di progetto di strumenti per il project management che svincolino il Responsabile di progetto da numerosi incarichi (vedi suddivisione/monitoraggio/mantenimento dei task). Discorso analogo vale per il tracciamento dei requisiti e degli obiettivi di qualità.

- **Con riferimento alla vostra esperienza di progetto didattico, ripercorrete la metodologia con la quale avete trasposto i requisiti utente, espressi esplicitamente o implicitamente nel capitolato, nei requisiti software da voi assunti. Valutate criticamente i frutti di tale metodologia,**

specialmente in rapporto alla maturità finale raggiunta sul problema da voi e dal vostro proponente.

Per la trasposizione dei requisiti, il gruppo ha agito nel seguente modo:

1. Lettura attenta del capitolato d'appalto
2. Primo incontro con la Proponente per discutere i requisiti fondamentali
3. Prima stesura di un'analisi dei requisiti generali che estrapolasse quest'ultimi dalle necessità evidenziate (esplicitamente o meno) nel capitolato o dalla Proponente in sede di riunione
4. Secondo incontro con la proponente per ottenere feedback sui requisiti stilati
5. Seconda stesura dell'analisi con maggiore dettaglio sui requisiti precedentemente analizzati

Tale metodologia, estesasi nei successivi periodi di progetto con un'approfondita analisi delle tecnologie e con numerosi ulteriori incontri con la Proponente, anche a detta di quest'ultima sembrerebbe essersi rilevata efficace nell'individuare le funzionalità del prodotto. È stata premura del gruppo l'instaurare un profondo e duraturo rapporto di collaborazione con la Proponente proprio affinché trovasse il maggior riscontro delle sue esigenze possibile nell'uso del software richiesto. Probabile margine di miglioramento risiede invece nel raffinamento dei requisiti maturato attraverso lo studio delle tecnologie, rivelatosi di importanza tanto fondamentale che una sua anticipazione (anche parziale) avrebbe senz'altro giovato non solo alla comprensione dei requisiti ma anche alle successive attività legate al processo di sviluppo.

- **Inquadrare la pratica nota come "continuos integration" nel dominio dell'ingegneria del software e illustrare concisamente alcuni dei metodi e degli strumenti che consentono di attuarla. Ove possibile, rapportare tali considerazioni all'esperienza personale guadagnata nella loro applicazione.**

  Nell'ingegneria del software, l'integrazione continua è una pratica che si applica in contesti in cui lo sviluppo del software avviene attraverso un sistema di versionamento e consiste nell'allineamento frequente dagli ambienti di lavoro locali degli sviluppatori verso l'ambiente condiviso. La CI viene spesso implementata per mezzo di un repository (ex: GitHub), verso cui ogni commit genera una build automatica in un'ambiente che simula quello di rilascio e a cui spesso è associata l'esecuzione di test automatici e il rilevamento di valori metrici. È proprio nel succitato modo che nel corso del progetto didattico il gruppo ha implementato il proprio sistema di CI, che ha garantito numerosi vantaggi in termini di velocità e qualità di sviluppo grazie ad un costante rilevamento di metriche su cui lavorare in termini di miglioramento e la possibilità di riempire il repository con solo codice testato e funzionante. In termini di qualità, l'applicativo Sonar è stato particolarmente utile nel permettere di stabilire dei gate di qualità per il codice e per la produzione di dashboard che rendessero agevole il suo monitoraggio.

- **Discutere la differenza tra le attività di versionamento e configurazione come applicate all'ambito dello sviluppo software.**

  Controllo di configurazione: le modalità in cui un sistema complesso viene gestito per mettere in sicurezza le baseline e consentire il ritorno a versioni precedenti.

  Configurazione: la modifica delle caratteristiche funzionali di un software sulla base dell'impostazione di opportuni parametri su propria preferenza.

Versionamento: consente tramite repository di contenere tutti i configuration item di ogni baseline e la loro storia completa. Una procedura a supporto del versionamento è il commit che equivale al tracciamento del completamento di un task a fronte della sua integrazione sul repository.

In particolare, la configurazione attiene alle modalità di integrazione delle varie componenti software, e può venire gestita al momento del versionamento (per esempio tramite integrazione continua). Il processo che controlla le attività di versionamento e configurazione è il controllo di configurazione, le cui attività agiscono nell'ottica di produrre con cadenza regolare baseline relative al codice sempre reperibili e a cui sia associato uno storico che permetta di recuperare lo stato precedente ad uno specifico commit.

- **Definire i concetti di design pattern e framework, mettendoli in relazione tra di loro e con la definizione di architettura. Indicare infine dove e quando tali tre concetti svolgano un ruolo significativo all'interno del processo di sviluppo.**

    - *Design pattern: insieme di best practise progettuali atte a garantire determinate buone qualità di un'architettura software che rappresentano una soluzione ad un problema ricorrente.*
    - *Framework: insieme integrato di componenti software, librerie e utilità documentate e di comprovata bontà utilizzate a supporto dello sviluppo di un nuovo applicativo. Esse concernono un approccio bottom-up in quanto forniscono componenti già realizzate, ma anche top-down quando hanno un impatto rilevante sullo stile architetturale.*

  Usualmente, i framework conseguono le proprie buone qualità tramite l'uso di design pattern. In relazione alla definizione di architettura, i framework possono rappresentare una parte (più o meno grande) delle componenti sfruttate dalla stessa ed implicare particolari modalità di integrazione ed interazione. I design pattern, invece, rappresentano i paradigmi architetturali ricorrenti che garantiscono un certo grado di efficienza e di efficacia. I concetti sopra esposti si applicano all'attività di progettazione del processo di sviluppo.

- **Fornire una definizione dei concetti di milestone e baseline, indicando come ciascuno di essi sia da utilizzare all'interno delle attività di progetto.**

    - *Milestone: punto di arrivo strategico, pianificato e misurabile del software, corrispondente al raggiungimento di un insieme di obiettivi definito. Una milestone è concretizzata da almeno una baseline.*
    - *Baseline: stato di riferimento dello sviluppo del prodotto in una data fase di progetto. Essa rappresenta una base verificata, approvata e garantita di un insieme di CI (solitamente collocata in un repository) dalla quale non si può retrocedere e sulla quale viene basato il successivo incremento.*

  Le milestone, concretizzate da una o più baseline, sono dunque elementi da pianificare con cura in quanto rappresentano stati d'avanzamento fondamentali per il progetto e riferimenti per il confronto fra stakeholders. Le baseline sono invece legate strettamente al controllo di configurazione ed in particolare al versionamento, che tramite repository assicura una versione comune (e auspicabilmente misurata e testata) del codice su cui basare i successivi incrementi.

- **Fornire una definizione dei termini efficienza ed efficacia, e discutere, tramite esempi concreti, come essi concorrano alla formazione delle strategie di organizzazione e di governo di progetto.**

    - *Efficienza: capacità di raggiungere gli obiettivi prefissati perseguendo una strategia di risparmio delle risorse (tempo e persone) a disposizione. L'efficienza è inversamente proporzionale alla quantità di risorse impiegate per le attività richieste e quindi è un indicatore quantitativo del consumo delle risorse di progetto;*
    - *Efficacia: il grado di conformità del prodotto rispetto alle norme vigenti e agli obiettivi prefissati. Per perseguire una buona efficacia è necessario impiegare sufficienti risorse. La strategia con la quale il fornitore garantisce la conformità viene fissato nel Piano di Qualifica.*

  Efficienza ed efficacia risultano dunque proprietà da perseguirsi in maniera trasversale all'intero progetto: fare ciò significa ottimizzare l'uso delle risorse massimizzando la resa del prodotto rispetto agli obiettivi fissati. L'efficienza viene perseguita attraverso l'applicazione delle norme stabilite, contestualizzate da un piano di progetto che imponga una ragionevole ottimizzazione delle risorse temporali e umana e orientato alla persecuzione di obiettivi di qualità rilevanti. Analogamente, l'efficacia viene perseguita attraverso politiche ad hoc che tengano conto degli obiettivi di progetto e in particolare della necessità di soddisfare i requisiti vincolati dalla committenza.

- **Illustrare qualche best practice personalmente sperimentata o riconosciuta come tale a posteriori, per il buon svolgimento delle attività di analisi dei requisiti.**

    - Raccolta di dati concreti e specifici che riflettano l'esigenza del prodotto da parte dell'utente: è di massima importanza analizzare ciò che ha creato il bisogno del prodotto nell'utente per comprendere in che modo questo possa davvero rivelarsi utile nei suoi confronti, discernendo dunque tra concreta utilità (che apporta business value) ed elementi accessori.
    - Stretto rapporto di collaborazione con la Proponente per evidenziare e successivamente dettagliare i requisiti: il rapporto con la Proponente permette di identificare e/o confermare le sue esigenze, raffinando i requisiti già emersi o evidenziandone di nuovi.
    - Produzione di requisiti atomici, misurabili e verificabili: ogni requisito confermato deve avere proprietà di atomicità (i requisiti vanno quanto più possibile scomposti), misurabilità (deve essere possibile rilevarne delle metriche a riguardo) e testabilità (deve essere possibile garantirne l'effettivo soddisfacimento), così da poter dimostrare efficacemente all'utente l'adempimento delle sue richieste.

- **Fissato il ruolo di amministratore di progetto all'interno di un organigramma tipo di progetto, descriverne le mansioni principali rispetto a un normale ciclo di sviluppo. Delineare le competenze richieste da tale ruolo e, facendo riferimento alla propria esperienza di progetto didattico, ipotizzare come esse possano essere acquisite efficacemente in funzione dei vincoli derivanti dal piano di progetto e dal piano di qualifica.**

  L'Amministratore di Progetto deve controllare e amministrare l'ambiente di lavoro con piena responsabilità sulla capacità operativa e sull'efficienza. Le sue mansioni sono:

    a. Ricerca di strumenti che migliorino l'ambiente di lavoro e che lo automatizzino ove possibile;

b. Gestione del versionamento;
c. Controllo di versioni e configurazioni del prodotto software;
d. Risoluzione dei problemi di gestione dei processi;
e. Controllo della qualità sul prodotto.

Alla luce di ciò, appare evidente che l'Amministratore necessiti di un'ampia conoscenza dell'ambiente di lavoro e del way of working del team, nonché delle relative conoscenze di project management. Gli è inoltre richiesta una discreta competenza amministrativa e tecnologica nell'ambito del versionamento, dell'integrazione continua e dell'automazione delle attività di progetto e del loro controllo. Esso rappresenta una personalità trasversale che si interfaccia con gli altri ruoli di progetto, redige le norme di progetto a stretto contatto con il team di progettazione e qualifica.

- **Un numero crescente di domini applicativi predilige l'adozione dei metodi di sviluppo agile (nelle loro svariate declinazioni). Richiamandone concisamente le caratteristiche distintive, discutere come esse si adattino alle esigenze del progetto didattico, ove possibile confrontandole con quelle del modello effettivamente adottato.**

  Il manifesto Agile recita:
  - *Individui e interazioni piuttosto che processi e strumenti*
  - *Software funzionante piuttosto che documentazione esaustiva*
  - *Collaborazione con il cliente piuttosto che negoziazione del contratto*
  - *Rispondere al cambiamento piuttosto che seguire un piano*

  Tali principi incarnano lo spirito della metodologia agile, che pone l'accento sul soddisfacimento del cliente piuttosto che sulla pianificazione e qualità del software. Si noti che questo non significa che pianificazione e qualità non siano perseguite da metodi agile, tuttavia non rappresentano la preoccupazione maggiore se comparate con l'effettiva funzionalità del software e soddisfazione del cliente. Calato nel contesto del progetto didattico, una metodologia agile si ritroverebbe probabilmente in conflitto con l'inesperienza del gruppo, che potrebbe naufragare senza il sostegno di una ferrea pianificazione e di una documentazione esaustiva. Tuttavia, attività gestite con metodologia Agile quale lo sviluppo di una technology baseline e di una product baseline si sono rivelate di cruciale e strategica importanza nel corso dello svolgimento del progetto. Benché guidate dalle scadenze di progetto, tali attività hanno senz'altro avuto un impatto positivo sulla comprensione dei requisiti e delle tecnologie, nonché sullo sviluppo della progettazione architetturale, permettendo al gruppo di acquisire conoscenze e competenze in modo pratico, piuttosto che attraverso la redazione di documenti.

- **Secondo ISO/IEC 12207, la gestione di progetto (project management) è un processo organizzativo. Richiamare concisamente le principali attività in esso comprese, associandovi – fornendone opportuna giustificazione - strumenti di supporto utili al loro svolgimento.**

  La gestione di progetto implica attività quali:
  - Gestione della pianificazione e assegnamento dei task: il responsabile di progetto deve pianificare il lavoro da svolgere suddividendolo in task, tenendo conto delle risorse a disposizione, per poi allocarli. Tale attività può essere svolta con l'aiuto di opportuni strumenti automatici specifici per il project management, quali Wrike Asana Bitrix24 e molti altri.

- Monitoraggio dell'esecuzione: il monitoraggio e l'analisi del progresso dell'esecuzione delle varie attività all'interno dei processi, e relativo intervento nel caso di anomalie. A supporto di tale attività possono essere utilizzati strumenti in grado di rilevare metriche pertinenti e proiettarle in dashboard che mostrino l'andamento del progetto a varie granularità.
- Accertamento della valutazione: il Responsabile di progetto deve accertarsi dell'effettiva verifica delle attività e dei processi nell'ottica del raggiungimento degli obiettivi di progetto e del miglioramento degli stessi. Tale attività è da svolgersi interfacciandosi con i Verificatori.

- **Immaginando di redigere un glossario per un documento formale esterno, fornire definizioni concise, efficaci, e valevoli nel dominio dell'ingegneria del software, dei termini: progetto, processo, attività, fase. Indicare fonti bibliografiche autorevoli a supporto delle definizioni fornite.**

  - *Progetto (da H. Kerzner): insieme di attività e compiti con le seguenti proprietà:*
    - *Agiscono per il perseguimento di un obiettivo comune*
    - *Sono associate ad una quantità finita di risorse, che consumano nel loro svolgersi*
    - *Sono contraddistinte da date di inizio e di fine prestabilite*
  - *Processo (da ISO/IEC 12207): insieme di attività correlate e coese che trasformano un input in un output consumando risorse.*
  - *Attività (da ISO 9000): insieme di compiti con uno scopo comune relativo ad un dato processo.*
  - *Fase: durata temporale entro un dato stato del ciclo di vita, o in una transizione tra stati, in cui vengono svolte attività specifiche e univoche.*

- **Fissando l'attenzione sulla definizione di processo associata allo standard ISO/IEC 12207, indicare quali processi sia possibile e opportuno istanziare su una attività di tagli analoga al progetto didattico, e con quale istanziazione concreta (cioè verso quale insieme di attività, obiettivi, e flussi di dipendenza).**

  Suppongo sia ragionevole istanziare almeno i seguenti processi e relative attività:
  - Processi primari
    - Fornitura
      - *Studio di fattibilità*: analisi e accertamento della realizzabilità del prodotto.
      - *Definizione del modello di sviluppo*: scelta ponderata del modello di sviluppo più adeguato.
      - *Pianificazione della qualità*: definizione degli obiettivi di qualità e delle strategie per raggiungerli.
    - Sviluppo
      - *Analisi dei requisiti*: individuazione e catalogazione dei requisiti.
      - *Progettazione*: progettazione architetturale e di dettaglio del prodotto.
      - *Codifica*: trasformazione della progettazione in codice sorgente.
  - Processi di supporto
    - Documentazione
      - *Produzione della documentazione:* sviluppo di una documentazione di progetto a supporto delle varie attività che lo compongono.
    - Gestione della configurazione

                - *Gestione del versionamento*: configurazione e amministrazione degli strumenti, delle procedure e delle modalità di versionamento.
                - *Gestione del rilascio*: definizione delle modalità e procedure inerenti il rilascio del prodotto.
            - Gestione della qualità
                - *Monitoraggio degli obiettivi di qualità*: rilevazione e analisi di metriche inerenti la qualità di processi e prodotti, nell'ottica di monitorare il progresso del raggiungimento degli obiettivi fissati.
                - *Accertamento della qualità*: definizione delle procedure di quality assurance per processi e prodotti
            - Verifica
                - *Analisi*: analisi statica/dinamica dei prodotti.
                - *Verifica*: implementazione della verifica di processi e prodotti (incluso il testing).
            - Validazione
                - *Validazione*: implementazione della validazione dei prodotti
    - Processi organizzativi
        - Gestione di progetto
            - Gestione della pianificazione e assegnamento dei task
            - Monitoraggio dell'esecuzione e gestione delle anomalie
            - Accertamento della valutazione

- **Discutere la relazione tra encapsulation e information hiding, esaminando criticamente l'affermazione: "encapsulation is a programming language feature; information hiding is a design principle".**

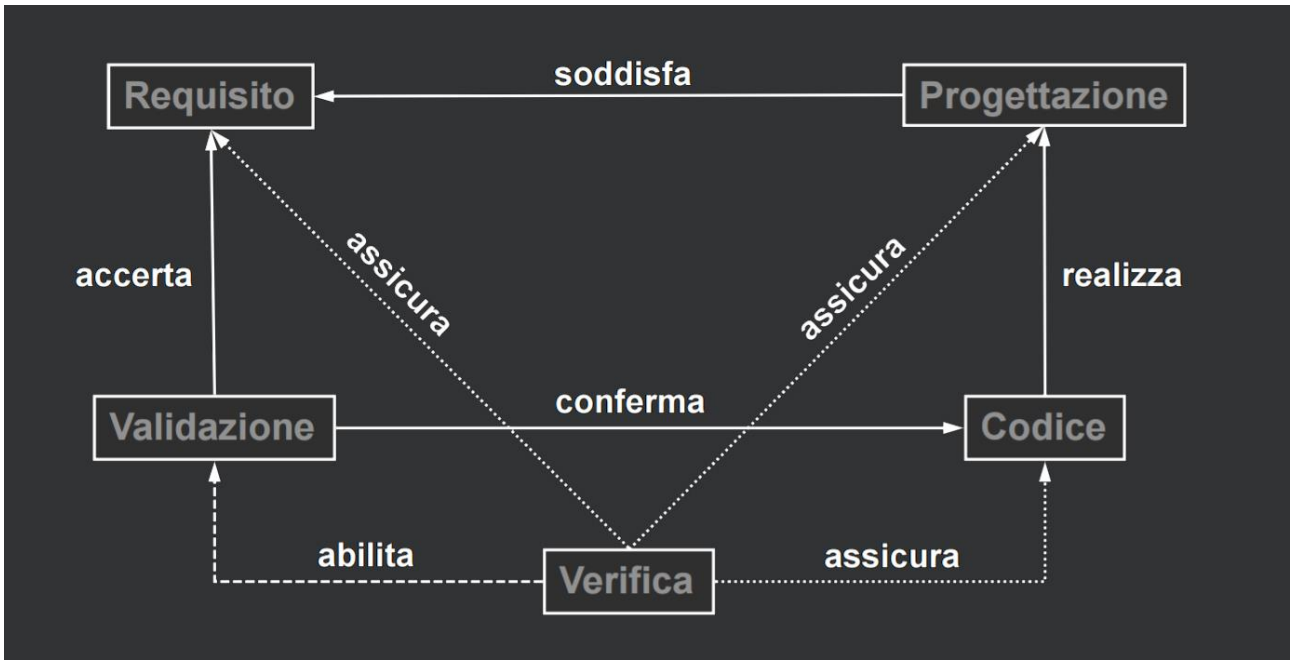
    - Incapsulamento: la tecnica che consiste nel nascondere il funzionamento interno di una parte di un programma, in modo da proteggere le altre parti da un suo malfunzionamento o cambio di imlementazione.
    - Information hiding: il principio secondo cui i dettagli implementativi una classe vanno resi trasparenti all'utente.
      L'information hiding si pone quindi come principio teorico implementato dall'incapsulamento, che ingloba i dettagli implementativi ed espone le funzionalità tramite intefacce.

- **Fornire una definizione di "requisito", applicabile al dominio dell'ingegneria del software, e descrivere, succintamente ma con precisione, il ciclo di vita dei requisiti dall'interno di un progetto del tipo di quello didattico, rappresentandolo come una apposita macchina a stati, specificando anche le attività poste sugli archi di transizione.**

  <u>*Requisito (da IEEE):*</u>

    1. *Capacità necessaria a un utente per risolvere un problema o raggiungere un obiettivo*
    2. *Capacità che deve essere posseduta (o condizione che deve essere soddisfatta) da un sistema per adempiere a un obbligo*
    3. *Descrizione documentata di una capacità interpretata come ai punti precedenti*

  Il ciclo di vita dei requisiti è rappresentato dalla seguente macchina a stati:



Un requisito nasce un'esigenza del cliente, e impatta sulla progettazione (che deve soddisfarlo) e quindi sul codice che la implementa. È necessario che un requisito sia sottoposto dapprima a verifica (che assicura anche la correttezza della progettazione e del software) e poi a validazione (che accerta la conformità del prodotto rispetto ai requisiti).

- **Illustrare le principali differenze, per obiettivi e modalità di svolgimento, tra le tecniche di inspection e di walktrough.**

  L'analisi statica è una tecnica, spesso supportata da strumenti automatici, che permette di individuare anomalie all'interno di documenti e codice sorgente (diversamente dall'analisi dinamica, essa è compiuta senza un'esecuzione del codice) durante tutto il loro ciclo di vita. Si può realizzare tramite due tecniche diverse:

  - *Walkthrough:* tecnica di analisi statica a largo spettro, attività collaborativa onerosa e poco efficiente, ma efficace. Tramite analisi Walkthrough il codice viene esaminato esaustivamente alla ricerca di errori, l'attività viene pianificata e documentata;
  - *Inspection:* viene svolta un'analisi mirata e strutturata, volta a localizzare gli errori segnalati in una lista di controllo stilata in ambito di pianificazione (talvolta sfruttando conoscenze ottenute tramite Walkthrough) con il minor costo possibile. Con l'incremento dell'esperienza, la lista di controllo viene progressivamente estesa rendendo l'ispezione via via più efficace.