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 è un modello studiato per il miglioramento continuo della qualità in un'ottica a lungo raggio. Serve per promuovere una cultura della qualità che è tesa al miglioramento continuo dei processi e all'utilizzo ottimale delle risorse. 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. Per migliorare la qualità e soddisfare il cliente, le quattro fasi devono ruotare costantemente, tenendo come criterio principale la qualità.

- P - Plan. Pianificazione.
- D - Do. Esecuzione del programma, dapprima in contesti circoscritti.
- C - Check. Test e controllo, studio e raccolta dei risultati e dei riscontri.
- A - Act. Azione per rendere definitivo e/o migliorare il processo.

Nel progetto didattico i principi del PDCA sono stati attuati per garantire una elevata qualità del prodotto, si sono quindi alternate in continuazione le attività di pianificazione come l'analisi dei requisiti, scelta del framework, gestione delle ore e del personale mediante piano di progetto, definizione delle norme di progetto si sono messe in atto le pianificazioni in base alla fase di lavoro (RR/RP/RQ/RA) suddivise in piccole sotto fasi controllando ogni azione e risultato per catalogare e risolvere i problemi riscontrati, i problemi trovati sono stati quindi stati registrati nel piano di progetto nei vari consuntivi di fase e discusse all'interno del team per alzare il livello qualitativo.

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.

Le ore pianificate nel progetto, riportate nel piano di progetto su tabelle e diagramma di gantt, relative alle varie fasi sono state rispettate al meglio, sforando di qualche ora di lavoro in più in alcune fasi. Questo problema si è verificato dal fatto che alcuni membri del team non hanno garantito la stessa efficienza e disponibilità di altri membri più diligenti. Il problema maggiore si è verificato durante la fase di codifica, qui si sono incontrati i maggiori problemi, in particolare la poca esperienza con la tecnologia HTML5 e Javascript adottata e la visione del codice come unica priorità lasciando in dietro l'aggiornamento di alcuni documenti da parte di alcuni membri. Proprio per quest'ultimo motivo il team ha sofferto dei ritardi nelle consegne e scarsa qualità dei documenti dovuto al fatto che spesso ci si è trovati a lavorare in meno coprendo le ore buche di altre persone.

La ripartizione ideale a mio avviso, dato il progetto di tipo didattico e la scarsa esperienza di sviluppo software lavorando in gruppo, non la trovo possibile. Le ore disponibili e i progetti accademici per ogni materia sono troppi e spesso vanno a cozzare portando fuori strada il lavoro in quanto ogni membro dà un'importanza diversa alle sue priorità penalizzando il resto del team. L'unica soluzione ovvia, che esiste poi nel mondo esterno, è la necessità di un membro leader stabile (non sottoposto a girare tra i vari ruoli).

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.

L'architettura SW nella storia inizia ad esistere con l'avvento dell'ingegneria software. Fino alla fine anni '80 definiva solo la struttura hardware fisica. Secondo boehm nel '95 e poi kructen nel '99 l'architettura sw è la decomposizione del sistema sw in componenti costitutive, l'organizzazione di tali componenti (ruoli, iterazioni e responsabilità), le interfacce necessarie all'interazione tra tali componenti ed i modelli di composizione di queste componenti.

[L'architettura software è l'organizzazione di base di un sistema, espressa dai suoi componenti, dalle relazioni tra di loro e con l'ambiente, e i principi che ne guidano il progetto e l'evoluzione.]

Nel mio progetto, per la pianificazione e l'evoluzione, sono state definiti mediante documenti (in particolare norme di progetto e piano di qualifica) i passi da fare per l'elaborazione del prodotto. Ogni singolo componente, riguardante il metodi di sviluppo, l'ambiente HW e SW, i framework le entità del prodotto finale ecc., sono state minimizzate in piccole parti secondo il design pattern MV* adottato dal team di sviluppo. La suddivisione tra le entità è avvenuta cercando di ottenere il miglior equilibrio tra information-hiding, coesione e accoppiamento.

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.

La capacità di un insieme di caratteristiche di un prodotto, sistema o processo, di soddisfare le esigenze dei clienti e degli altri portatori di interesse.

[Gestione della qualità: L'insieme delle attività coordinate per dirigere e controllare un'organizzazione rispetto alla qualità. La qualità può essere gestita secondo diverse componenti quali la politica e gli obbiettivi di qualità. Pianificazione della qualità, il suo controllo ed accertamento, il suo miglioramento.

Pianificazione della qualità : Le attività della gestione della qualità mirate a definire gli obbiettivi della qualità ed i processi e le risorse necessarie per conseguirli. Controllo della qualità :Le attività della gestione della qualità messe in atto affinché il prodotto soddisfi i requisiti. Accertamento della qualità: Le attività della gestione della qualità messe in atto per accertare che i requisiti siano soddisfatti.

L'accertamento è mirato al soddisfacimento degli obbiettivi. La percezione del livello di soddisfacimento

Si può parlare di qualità riferita a:

- prodotto (visione verticale): bene o servizio
- sistema: insieme degli elementi in cui il prodotto si colloca
- processi (visione orizzontale): attività correlate finalizzate alla realizzazione degli obbiettivi
- organizzazione : sulla struttura e l'amministrazione, a maggior ritorno economico.]

Nei documenti per garantire la qualità si applicano: quality assurance(si intende l'insieme delle attività volte a garantire il soddisfacimento degli obiettivi della qualità), verifica e validazione.

Per garantire un alto livello di qualità il team ha adottato vari metodi, tra i principali si è utilizzato il PDCA quindi monitoraggio continuo per migliorare auto verificandosi, o mediante i documenti dove sono state inserite tutte le pianificazioni e obiettivi da rispettare, come i requisiti analizzati inizialmente si sono verificati facendo un opportuno studio di fattibilità e successivamente verificati (per esempio tramite i test). Un altro esempio può essere dato dalla pianificazione delle ore di lavoro e del personale (per mezzo di abelle e diagrammi, es gantt), questa è una pianificazione alza il livello qualitativo in quanto il progetto viene processato adeguatamente secondo la tabella di marcia prestabilita.

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.

dislocazione temporale delle attività per rappresentarne la durata, la sequenzialità e il parallelismo. Il collocamento delle attività è arbitrario, l'importante è il collocamento preciso nell'asse del tempo per avere chiare inizio e fine (piano preventivo). Può succedere che l'attività abbia uno sviluppo diverso dalla pianificazione e con i diagrammi si può notare questo fatto anche visivamente. Essi infatti possono fungere da strumento visivo per lo studio di attività (parallelismo, ordine, ritardi, dipendenze, slittamenti).

Al diagramma di Gantt visto precedentemente mancano le dipendenze quantificate in tempo tra le attività,

Nel mio team bla bla bla..

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.

Le strategie che hanno portato i maggiori benefici senza dubbio le pianificazioni effettuate nel piano di progetto grazie al diagramma di Gantt. Con questo strumento si è potuto avere una ben delineata pianificazione dello sviluppo del progetto limitando gli sprechi di tempo nonostante le mancanze di personale avute in alcune fasi. Grazie quindi alla definizione delle fasi il team è riuscito a riorganizzare il lavoro anche senza la totale presenza dei membri.

Nella fase di pianificazione iniziale, durante l'analisi dei requisiti, si sono valutati erroneamente alcune necessità che si sono rivelate poi troppo difficili da rispettare. Per esempio nel nostro progetto bisognava implementare un trasferimento dati tra uno script HTML5 e un servizio cloud a scelta. Per natura i browser hanno una protezione che limita questa transizione verso l'esterno dei dati impedendo quindi il corretto soddisfacimento del requisito.

Un altro punto a favore riguarda la buona assunzione del preventivo, a fine progetto ci si ritrova ad aver sforato di 70€ in più rispetto a quanto chiesto inizialmente.

Avendo usato il PDCA abbiamo potuto valutare continuamente il nostro operato e risolvere tempestivamente la maggior parte dei problemi incontrato minimizzando lo sforzo.

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 : intende accertare che l'esecuzione di un dato processo non abbia introdotto errori. È principalmente rivolta al processo, ma applica anche ai prodotti di processi intermedi.

Validazione : intende accertare che l'uscita dell'insieme di processi eseguiti sia il prodotto atteso.

Verifica e validazione: tramite revisione interna e/o esterna, prototipazione, analisi del modello concettuale.

I processi di verifica, facendo riferimento anche al modello PDCA di verifica continua, vengono e devono essere effettuati per tutta la durata del progetto verificando (e quindi definendo il livello di qualità) che ogni attività non introduca nuovi errori o problemi. La verifica deve essere fatta sulle singole entità minimizzate e ognuna deve essere registrata per poi ottenere un report garantendo il continuo miglioramento. Alcune tecniche di verifica possono essere:

- nel caso dei requisiti la loro realizzabilità tramite un opportuno studio di fattibilità e non solo;
- nel caso di metodi (del codice sorgente) si eseguono test di unità e per le classi test di integrazione;
- nel caso dell'ambiente di sviluppo si fa un'analisi per capire cosà è possibile fare e in quanto tempo.

La validazione riguarda invece il prodotto uscente, quindi se rispetta tutti i requisiti definiti inizialmente ed e quindi di gradimento da parte del committente/richiedente o degli utenti finali. La validazione va quindi ad attestare alla fine il risultato dichiarando la qualità finale del prodotto.

Presentare concisamente almeno due metriche utilizzabile 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.

Risulta utile, al fine di sapere quanto e quale codice è stato testato, misurare la copertura del codice durante le varie esecuzioni. Queste misurazioni possono osservare diversi aspetti, tra i quali segnaliamo: statement coverage: indica quali statement in un metodo, o in una classe, sono stati eseguiti. In particolare, risulta utile per identificare quali blocchi di codice non sono stati raggiunti; branch coverage: indica quali rami decisionali sono stati percorsi durante l'esecuzione.

Il numero di cammini linearmente indipendenti di un metodo può essere indice di alcune anomalie del codice. Tale numero e conosciuto come complessità ciclomatica (C). Se rappresentiamo il codice come un grafo, C è calcolabile in questo modo: C = E - N + P dove E è il numero di archi, N il numero di nodi e P le componenti connesse da ogni arco. Un alto valore di C può evidenziare un codice più complesso, quindi difficilmente leggibile. Tuttavia un basso valore di C non sempre garantisce un'elevata comprensibilità.

Descrivere la tecnica di classificazione e tracciamento dei requisiti adottata nel proprio progetto didattico e discuterne l'efficacia e i limiti riscontrati.

Per la classificazione dei requisiti il team ha adottato varie tecniche tra cui Interviste con il cliente, Discussioni creative tipo 'Brainstorming' (approccio maieutico), Osservazione dei comportamenti e dei bisogni ricercando su internet studi adeguati al nostro progetto. La tecnica più utilizzata è stata il Brainstorming, il team ha quindi utilizzato parte delle ore per definire e tracciare i requisiti. Il problema di questo approccio deriva dalla poca esperienza di ogni membro del team. Ci siamo quindi trovati di fronte a delle problematiche che sono state assunte per logica e studi mirati all'occasione. Un adeguato livello di esperienza avrebbe sicuramente giovato maggiormente l'esito della validazione finale. Questo problema non è però da considerarsi grave in quanto un prossimo progetto potrà essere facilitato dall'esperienza appresa da questa prova.

Verifica e validazione

Due termini molto ricorrenti in ingegneria dei requisiti, sono:

Verifica: accertare che l'esecuzione delle attività di un processo non abbia introdotto errori. È rivolta ai processi, e viene fatta quindi più volte in un ciclo di vita.

Validazione: verificare che il prodotto realizzato rispetti i requisiti. Viene fatta solo sui prodotti.

qualifica = verifica + validazione.

La verifica sui documenti è obbligatoria, tipicamente si fa tramite una lettura dell'intero documento, questa tecnica si chiama walkthrough, in italiano sarebbe andare a pettine. Questa tecnica è molto costosa, perché impiega un sacco di tempo, soprattutto per documenti tanto corposi.

Una tecnica alternativa è inspection, cioè non una lettura completa ma mirata alla lettura dei punti dove è più probabile riscontrare errori. L'ispezione viene usata quando si ha un buon grado di fiducia su chi ha scritto il documento e un buon grado di fiducia sul rispetto delle regole definite per la stesura del documento.

Verifica sui requisiti: durante la verifica mi rendo conto che un requisito non è atomico, devo suddividerlo in sottorequisiti atomici. Inoltre ogni requisito deve essere foglia, qualsiasi requisito
non dovrà mai avere un padre.

Verifica e validazione: Revisione interna, Prototipazione, prove formali.

Nella verifica e validazione continuare con analisi statica e dinamica e quindi test.

Analisi statica e dinamica

Analisi statica: il svolgimento di questa analisi non richiede l'esecuzione del programma, tipicamente viene usata nell'attività di verifica.

Analisi dinamica: il suo svolgimento richiede l'esecuzione del programma, tipicamente è usata nell'attività di validazione. Questa analisi deve essere ripetibile. Gli strumenti per fase analisi dinamica sono driver(componente per pilotare una parte), stub(componente per simulare una parte, logger(componente per registrare i dati di esecuzione).

L'affidabilità invece è dimostrabile con la combinazione di analisi statica e dinamica.

L'analisi statica sarà l'attività preliminare, invece l'analisi dinamica sarà l'attività di completamento.

L'usabilità verrà eseguita un'analisi statica preliminare e una verifica dei manuali d'uso.

Efficienza: avrà anch'essa un'analisi statica preliminare e una verifica alle norme di codifica.

Manutenibilità: analisi statica preliminare e un controllo alle norme di codifica e sulle prove di stabilità.

Portabilità: analisi statica come strumento ideale e un'analisi delle norme di codifica.

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.

Durante il test didattico sono stati svolti diversi tipi di test. un esempio sono:

Test di unità: il test di unità verrà svolto con un'analisi dinamica ed il supporto in piccola scala di attività mirate di analisi statica. Si svolge con il massimo grado di parallelismo. Lo scopo è di verificare la correttezza del codice. Meglio se fatto in modo automatico

Test di integrazione: integra le varie unità in maniera incrementale ricostruendo il sistema. In condizioni ottimali l'integrazione è priva di problemi. Rileva tutti i problemi di incompatibilità tra le varie unità ed integrazioni con riuso mal fatto.

Test di sistema e collaudo: verrà eseguita una validazione che consiste in un test di sistema per il fornitore e un collaudo per il committente.

Test di regressione: l'insieme di test necessari per accertare che la modifica di una parte P di S non causi errori in P o nelle altre parti di S che dipendono da P. modifiche effettuate per aggiunta, correzione o rimozione non devono causare problemi a componenti già verificati, questo è più probabile che accada nel caso di un alto indice di accoppiamento e una bassa incapsulazione. Queste verifiche sono avvenute tramite test funzionali e strutturali (Verica funzionale (black box): si opera a scatola chiusa, controllando che il sistema, dato un input, ritorni l'output atteso. Si applica ad agglomerati di componenti o al programma nella sua interezza; Verica strutturale (white box): si opera aprendo la scatola, cioè andando ad utilizzare le funzionalità delle singole componenti del sistema e controllando la loro correttezza). Poi ti inventi che alcuni test sono stati utili perché hanno rilevato inesattezze causate da persone che non hanno moltissima dimestichezza con la programmazione permettendo al programma di raggiungere un'elevata qualità...

Alfa-testing, insieme di controlli effettuati all'interno dell'azienda, volto ad assicurare che siano stati soddisfatti i requisiti del prodotto, tanto quelli espliciti, quanto quelli impliciti. Serve anche a rilevare i malfunzionamenti più evidenti, qualora siano presenti. Tramite test di sistema
Beta-testing, è il collaudo vero e proprio, che avviene alla presenza del committente ed ha valenza contrattuale in quanto serve a dimostrare al cliente che i requisiti da lui imposti sono stati pienamente soddisfatti. In questa sotto-fase si svolgono per lo più: ***Test di accettazione***, l'insieme di test volti a verificare che il sistema nella sua interezza soddisfi i requisiti utente specificati nel capitolato e nelle riunioni con il committente.
Le risorse necessaria ai test sono: risorse umane(amministratore del progetto, responsabile del progetto, programmatori e verificatori), risorse software e risorse hardware.
La fase di test è implementata utilizzando Jasmine[g], permette di effettuare i test dinamici in tutto il codice javascript. Jasmine è un framework per il tesing di tipo BDD (Behavior-Driven Development), questo significa che, ad ogni esecuzione permette di fare test di regressione perché testa tutto il codice.

Analisi dei requisiti

Classificazione dei requisiti:
Attributi di prodotto: definiscono le caratteristiche richieste dal sistema. Questi attributi creano requisiti funzionale, prestazionali e qualitativi.
Attributi di processo: pongono dei vincoli sui processi del progetto, per esempio una imposizione di una particolare tecnologia o linguaggio da utilizzare. Questi attributi creano requisiti extra-funzionali, spesso si inseriscono come proprietà attese dal sistema.

I requisiti si dividono in 4 tipi:
Requisiti funzionali: Requisiti su ciò che il sistema deve fare. Specificano quali funzioni il sistema deve fornire per soddisfare i bisogni degli stakeholder.
Requisiti del prodotto che descrivono le interazioni tra il sistema e l'utente indipendenti dall'architettura;
Requisiti prestazionali: requisiti del prodotto che aumentano l'efficienza del software;
Requisiti qualitativi: requisiti del prodotto che possono aumentare la qualità del software;
Requisiti dichiarativi: requisiti che descrivono le condizioni obbligatorie imposte dall'utente.

Questi requisiti devono però essere verificati, a seconda del requisito si hanno varie modalità di test:
Requisiti funzionali: test, dimostrazione formale;
Requisiti prestazionali: misurazione;
Requisiti qualitativi: verifica per ogni singolo requisito;
Requisiti dichiarativi: revisione;

Un'altra classificazione dei requisiti è secondo il grado di importanza:
Requisiti obbligatori: Il prodotto deve soddisfare necessariamente tutti i requisiti definiti obbligatori;
Requisiti desiderabili: il prodotto non deve necessariamente soddisfare questi requisiti, ma se soddisfatti crea valore aggiunto. Tipicamente questi requisiti vengono sviluppati a progetto ultimato nel caso sia rimasto del tempo libero.
Requisiti opzionali: sono dei requisiti relativamente utili, possono dare un valore aggiunto al progetto nel caso di una gara di appalto nella quale il committente voglia del valore aggiunto

---

Accoppiamento e Coesione
Coesione: è una proprietà dei singoli componenti, impone di decomporre il grande in sottopezzi e
le funzioni con lo stesso obbiettivo devono essere contenute nello stesso componente. La coesione và massimizzata. Misura la rilevanza di un certo componente. Per esempio un metodo apre e un metodo chiude sono tra di loro coesi. La coesione aiuta la copertura e rende la vita più facile nella verifica. Se un componente è massimamente utilizzato Fan-in e minimamente accopiato fun-out allora è coeso.
Accoppiamento, bisogna ridurre il grado di dipendenza stretta tra le componenti, in maniera tale da permettere la modifica di una componente senza che ciò comporti necessariamente la modifica delle altre.

Evitare la complessità eccessiva, ma usare una coesione massima e un accoppiamento minimo, quindi evitare il collegamento con l'interno della classe con l'esterno, un accoppiamento eccessivo rende difficile la manutenzione, mentre la coesione collega tutto l'interno e non l'esterno. Le classi sono utili per la coesione, ma devo partire da un design volto alla coesione. Anche l'accoppiamento si evita con il design e soprattutto progettando applicando l'information hiding.

Collaudo

Il collaudo interno verrà eseguito verificando le seguenti parti:
Verifiche finali di tracciamento: controllo di errori durante il passaggio tra le varie fasi di sviluppo
Prove di esecuzione: esecuzione del programma e delle funzionalità proposte

Collaudo esterno: visualizzare tutte le caratteristiche del programma a committente e cliente.

Scopo di ogni ruolo

- Responsabile (RE): ha il compito di scrivere e mantenere la documentazione riguardante il piano di progetto. Si occupa inoltre di gestire le risorse e pianificare le attività.
- Amministratore (AM): scrive e mantiene le norme di progetto, tiene sotto controllo le risorse, l'infrastruttura e gli strumenti di sviluppo, garantendone la corretta funzionalità.
- Analista (AN): si occupa dell'analisi e della comprensione dei requisiti impliciti ed espliciti, oltre che a redigere lo studio di fattibilità e l'analisi dei requisiti.
- Progettista (PT): redige la specifica tecnica e si occupa di progettare l'architettura del sistema.
- Programmatore (PM): converte il progetto di dettaglio in codice, in maniera ben documentata e seguendo le norme di codifica.
- Verificatore (VR): è responsabile dell'attività di verifica del sistema, illustra l'esito e la completezza delle prove effettuate e redige il piano di qualifica.

Gestione dei rischi

Analisi dei rischi: Insufficienza di personale, Incapacità di svolgere un ruolo, Instabilità dei requisiti, Pianificazione e budget non realistici, Problematiche relative alle tecnologie.
Metriche di misurazione: basse, medie, alte
Il processo di valutazione nei seguenti passaggi:

- individuazione delle sorgenti di pericolo, è penalizzata ad individuare gli elementi in grado di causare un effetto indesiderato;
- individuazione dei soggetti esposti;
- stabilire la priorità dei rischi, viene prodotta la stima del rischio (Risk Assessment);
- scelta degli interventi, in base alle priorità si risolve il problema;
- attuare le misure di controllo sugli interventi, periodicamente bisogna controllare l'efficacia di un intervento messo in pratica;
- valutare l'efficacia dell'intervento, acquisiti i dati di controllo si discute dell'efficacia ottenuta.