Ingegneria del Software a.a. 2021-22
Prova Scritta del 11 Luglio 2022

Esercizio di sbarramento

COGNOME **NOME** **MATRICOLA**

Rispondere alle seguenti domande. Per ogni domanda, solo una soluzione è corretta. L'esercizio si ritiene superato se si risponde correttamente ad **almeno 7 domande.**

Domanda 1

Questa figura vista a lezione rappresenta il processo di rilascio del metodo di sviluppo denominato Extreme Programming. Che cosa rappresentano i cubi i1, i2 e i3?

a) Rappresentano degli artefatti generici (ad esempio requisiti, codice, design, piani dei test) che vengono rilasciati al cliente
b) Rappresentano versioni di codice che vengono rilasciate ad un insieme di clienti selezionati per la fase di Beta-testing. Solo dopo questa fase il codice verrà rilasciato a tutti i clienti
c) Rappresentano versioni di codice funzionante (nel senso che può essere eseguito), via via sempre più complete, che vengono rilasciato al cliente
d) Rappresentano versioni di codice che vengono passate al gruppo di test per effettuare le fasi di validazione e verifica di sistema. Solo dopo questa fase il codice sarà rilasciato al cliente

---

Domanda 2

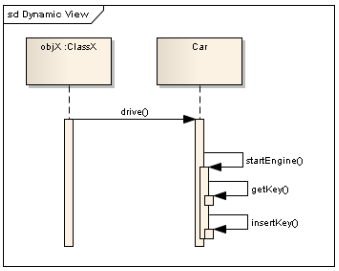
E' risaputo che sviluppare un sistema software di buona qualità è complesso. Tale complessità è da attribuire a varie ragioni, ma ci sono dei fattori che sono più importanti di altri. In particolare durante il corso di IS è stato sottolineato spesso un fattore che in molti casi è la principale causa di fallimento di un progetto:

a) la complessità intrinseca del problema. Spesso i sistemi software che devono essere prodotti risolvono problemi molto complessi (ad esempio sistemi di controllo di reattori nucleari)
b) il fattore tecnologico. Esistono molti "prodotti" simili forniti da diversi vendor (ad esempio linguaggi di programmazione, framework, application server) e scegliere quello adatto al proprio progetto è molto complesso
c) il fattore umano. Gestione del personale, problemi di comunicazione tra gli stakeholder e gli analisti/sviluppatori, difficoltà a lavorare in gruppo
d) il fattore economico, ovvero mancanza di liquidità durante lo sviluppo del progetto

---

Domanda 3

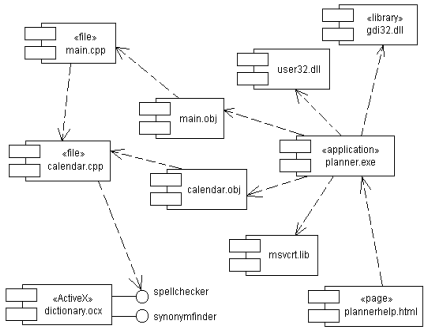
Quale delle seguenti affermazioni è **scorretta**?



a) objX chiama l'operazione drive() di Car, a sua volta drive() chiama startEngine(), startEngine chiama getKey() e insertKey() in successione
b) getKey() è un self-message
c) drive() è un'operazione della classe ClassX
d) insertKey() è chiamata all'interno dell'operazione startEngine()

Domanda 4

Nel seguente component diagram UML "preso da internet" cosa non è corretto rispetto a quanto visto a lezione?



a) Gli archi che mostrano dipendenza tra componenti sono mostrate con una freccia tratteggiata ed invece dovrebbe essere continua
b) I due lollipop (spellchecker e synonymfinder) sono interfacce esposte e quindi dovrebbero essere rappresentate con i socket
c) Gli stereotipi (ovvero lo specificare il tipo di componente tra parentesi angolari << >>) non sono ammessi per le componenti SW

d) In questo diagramma sono rappresentati gli artefatti al posto delle componenti. Le componenti sono entità logiche che sono realizzate da artefatti (che sono invece entità fisiche).

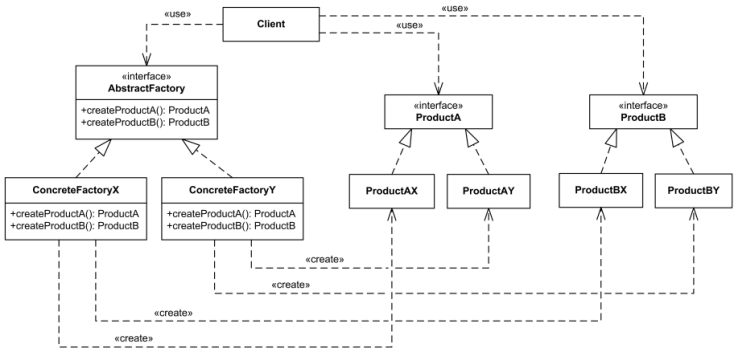
Domanda 5

Supponiamo di dover implementare un software per il gioco del monopoli con le classi: Monopoli, Scacchiera, Casella, Pezzo e Giocatore. Chi dovrebbe creare un oggetto di tipo Casella? (ovvero chi ha la responsabilità di creare un istanza della classe Casella?)

a) La classe Monopoli
b) La classe Scacchiera
c) La classe Casella
d) La classe Pezzo

Domanda 6

Quando viene comodo utilizzare il seguente design pattern?



a) quando si vuole convertire l'interfaccia di una classe in un'altra interfaccia che il cliente si aspetta
b) quando si vogliono introdurre strategie per la gestione della memoria per migliorare le prestazioni (es. Caching)
c) quando si vogliono aggiungere facilmente nuovi prodotti (ad esempio ProductC)
d) quando si vogliono creare famiglie di prodotti (oggetti connessi o dipendenti tra loro), in modo che non ci sia necessità da parte dei client di specificare le classi concrete dei prodotti all'interno del proprio codice

---

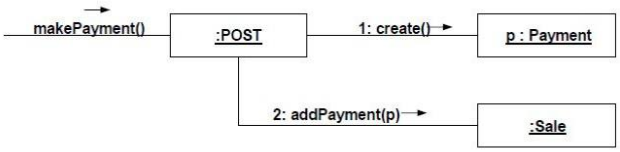
Domanda 7

I metodi (o approcci) di sviluppo plan-driven vengono usati in certi contesti (ad esempio quando abbiamo a che fare con un grosso sistema safety-critical) mentre i metodi agili sono più adatti in altri (ad esempio con sistemi e team piccoli, clienti e utenti disponibili, ambiente e requisiti volatili). Come ci si deve regolare se dobbiamo affrontare un progetto che possiede dei fattori "contrastanti"? Ad esempio un progetto in cui sistemi e team sono piccoli ma clienti e utenti non sono disponibili e i requisiti sono stabili?

a) E' preferibile andare sul sicuro ed utilizzare un metodo plan-driven quale ad esempio UP
b) Siccome è noto che i metodi agili (ad esempio XP) sono i più usati e sono decisamente più performanti è meglio utilizzare un metodo agile anche nei casi dubbi
c) Bisogna cercare di bilanciare il rigore/disciplina dei metodi plan-driven con la flessibilità dei metodi agili
d) In questi casi rari è consigliato utilizzare il metodo di sviluppo "code & fix"

Domanda 8

Si supponga un sistema per la gestione dei pagamenti dove POST è una classe usata per gestire nuovi acquisti con l'operazione makePayment(). Guardando il diagramma quale tra le seguenti affermazioni **è l'unica vera**?



a) La classe POST è accoppiata fortemente (strong coupled) con le altre classi
b) Il diagramma UML in questione è un object diagram

c) makePayment() crea un'istanza di Payment chiamata p e successivamente chiede al reparto vendite (Sale) di aggiungere il pagamento p
d) La prima chiamata dell'operazione makePayment() (identificata con 1:) crea un'istanza di Payment chiamata p. La seconda chiamata di makePayment() (identificata con 2:) chiede al reparto vendite (Sale) di aggiungere il pagamento p

---

Domanda 9

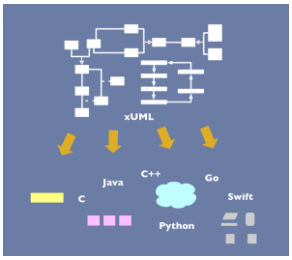
Che cosa è la **Textual Analysis** (o analisi del testo) nel contesto dell'ingegneria del Software?

a) è la maniera a cui ci si riferisce all'uso dell'elaborazione del linguaggio naturale, analisi testuale e linguistica computazionale per identificare ed estrarre informazioni soggettive da diverse fonti
b) è un processo utilizzato durante la fase di estrazione dei requisiti e per passare al design del sistema. Serve in particolar modo per costruire i diagrammi comportamentali UML (es. state machine e activity diagram)
c) è un processo utilizzato per analizzare il dominio di un Sistema (system domain). Aiuta ad identificare le classi, gli use case e gli attori a partire dalla descrizione del problema data in forma testuale
d) è un insieme ampio ed eterogeneo di tecniche manuali o assistite da computer di interpretazione contestualizzata di documenti provenienti da processi di comunicazione in senso proprio (testi) o di significazione (tracce e manufatti), aventi come obiettivo finale la produzione di inferenze valide e attendibili

---

Domanda 10

Che cosa potrebbe esemplificare la seguente figura?



a) Una lista di linguaggi di programmazione
b) La migrazione di un software su diveris nodi computazionali
c) Il metodo di sviluppo agile programming (programmazione agile)
d) L'applicazione del model driven development (MDD)