1. Cosa è un metodo?

Un sistema per fare qualcosa in modo migliore

Un procedimento generale per risolvere classi di problemi

Un particolare approccio o filosofia per fare qualcosa

1. Quale di queste affermazioni non è appropriata per il modello di processo a cascata ?

Le fasi del processo sono in progressione sequenziale

I semilavorati all’uscita di una fase sono congelati e non possono essere più modificati

Le funzionalità del sistema sono sviluppate in maniera incrementale

1. Un prototipo viene sviluppato per:

Ridurre i costi di sviluppo

Rilasciare rapidamente al cliente una prima versione del sistema

Interagire con il committente per convalidare i requisiti

1. Quale tipo di prototipazione parte con i requisiti meglio compresi?

Prototipazione esplorativa

Prototipazione throw-away

Prototipazione mock-ups

1. Quale tipo di prototipazione tende a realizzare l’interfaccia utente?

Prototipazione esplorativa

Prototipazione breadboards

Prototipazione mock-ups

1. Quale di queste affermazioni è appropriata per il modello di processo incrementale?

 Ogni versione produce funzionalità/sottosistemi più affidabili

 Ogni versione aggiunge nuove funzionalità/sottosistemi al sistema

 Ogni versione raffina le funzionalità/sottosistemi che sono presenti fin dall’inizio

7. Quale di queste affermazioni è appropriata per il modello di processo iterativo?

 Ogni versione raffina le funzionalità/sottosistemi che sono presenti fin dall’inizio

 Ogni versione aggiunge nuove funzionalità/sottosistemi al sistema

 I requisiti a più alta priorità vengono rilasciati per primi

8. Cosa è un team ?

 Un insieme di persone a cui è assegnato un task comune, ma che lavorano individualmente e

senza necessità di interazione

 Un insieme di persone che rivedono un work product e che propongono azioni

 Un piccolo insieme di persone che lavorano in stretta interazione sulla stessa attività o task

9. Il tempo durante il quale un certo lavoro in un progetto deve essere fatto, viene denominato:

 Work-product

 Schedule

 Task

10. Il lavoro che deve essere eseguito da un partecipante al progetto, viene denominato:

 Work-product

 Schedule

 Task

11. Quale di queste affermazioni non è valida per una struttura organizzativa di reporting

gerarchica?

 Lo stato è riportato dall’alto verso il basso

 Le decisioni vengono comunicate dall’alto verso il basso

 Le informazioni relative a stato e decisioni sono unidirezionali

12. Cosa è un liason all’interno di un team?no

 Il leader del team

 Il responsabile della comunicazione con un altro team

 Il consulente amministrativo del team

13. Che tipo di ruolo è un configuration manager?no

 Uno sviluppatore

 Un manager

 Un liason

14. Che cosa è un GANNT ?no

 Un grafico a barre che descrive lo schedule e la durata dei vari task

 Un grafo che descrive le dipendenze tra i task del progetto

 Un grafo che descrive le dipendenze tra i work-product del progetto

15. Che cosa è un PERT?NO

 Un grafico a barre che descrive lo schedule e la durata dei vari task

 Un grafo che descrive le dipendenze tra i task del progetto

 Un grafo che descrive le dipendenze tra i work-product del progetto

16. Quale dei seguenti non è una comunicazione pianificata?

 Definizione del problema

 Review di progetto

 Richiesta di modifica

17. Quale dei seguenti è una comunicazione non pianificata?

 Risoluzione di un problema

 Ispezione

 Rilascio

18. Quale dei seguenti meccanismi di comunicazione è asincrono? NO

 Intervista strutturata

 Riunione

 Posta elettronica

19. Cosa è un modello?

 Un’astrazione che descrive il sistema o un sottoinsieme di un sistema

 Un insieme di regole grafiche o testuali per rappresentare viste

 Una vista di aspetti del sistema

20. Cosa è una vista?

 Un’astrazione che descrive il sistema o un sottoinsieme di un sistema

 Una visualizzazione di particolari aspetti di un modello

 Un insieme di regole grafiche o testuali

21. Cosa è una notazione?

 Un’astrazione che descrive il sistema o un sottoinsieme di un sistema

 Una visualizzazione di particolari aspetti di un modello

 Un insieme di regole grafiche o testuali per rappresentare viste

22. Cosa è UML?

 Una notazione grafica per progettare sistemi software

 Un insieme di linguaggi per modellare software

 Un modello astratto per descrivere sistemi software

23. Quale di questi diagrammi non è usato per descrivere il comportamento dinamico di un

sistema software?

 Sequence diagram

 Activity diagram

 Use case diagram

24. I diagrammi dei casi d’uso:

 Descrivono il comportamento funzionale del sistema così come visto dagli utenti

 Descrivono il comportamento dinamico tra gli attori e il sistema e tra gli oggetti del sistema

 Descrivono il comportamento dinamico di un sistema, in particolare il workflow

25. I diagrammi delle attività:

 Descrivono il comportamento funzionale del sistema così come visto dagli utenti

 Descrivono il comportamento dinamico tra gli attori e il sistema e tra gli oggetti del sistema

 Descrivono il comportamento dinamico di un sistema, in particolare il workflow

26. I diagrammi di sequenza:

 Descrivono il comportamento funzionale del sistema così come visto dagli utenti

 Descrivono il comportamento dinamico tra gli attori e il sistema e tra gli oggetti del sistema

 Descrivono il comportamento dinamico di un sistema, in particolare il workflow

27. I diagrammi di stato:

 Descrivono la struttura statica del sistema: oggetti, attributi e relazioni

 Descrivono il comportamento dinamico tra gli attori e il sistema e tra gli oggetti del sistema

 Descrivono il comportamento dinamico di un singolo oggetto come una macchina a stati

Finite

28. I diagrammi delle classi:

 Descrivono la struttura statica del sistema: oggetti, attributi e relazioni

 Descrivono il comportamento dinamico tra gli attori e il sistema e tra gli oggetti del sistema

 Descrivono il comportamento dinamico di un singolo oggetto come una macchina a stati

Finite

29. In UML le classi e gli oggetti (istanze) sono rappresentate con:

 Rettangoli

 Ovali

 Rettangoli con angoli arrotondati

30. In UML i casi d’uso sono rappresentati con:

 Rettangoli

 Ovali

 Rettangoli con angoli arrotondati

31. In UML gli stati sono rappresentati con:

 Rettangoli

 Ovali

 Rettangoli con angoli arrotondati

32. Quale di questi elementi non è parte di un diagramma dei casi d’uso:

 Attore

 Caso d’uso

 Classe

33. Quale di questi elementi non è parte di un diagramma di sequenza:

 Attore

 Caso d’uso

 Oggetti

34. Che tipo di relazione può esistere tra un attore e un caso d’uso:

 Associazione

 Dipendenza

 Generalizzazione

35. Che tipo di relazione può esistere tra due attori:

 Associazione

 Dipendenza

 Generalizzazione

36. Quale di queste relazioni non esiste tra due casi d’uso:

 Associazione

 Dipendenza

 Generalizzazione

42. Quali di questi è un requisito funzionale:

 Il sistema deve visualizzare l’ora in base alla sua locazione

 Il tempo di risposta deve essere meno di un secondo

 Il linguaggio di implementazione deve essere Java

43. Quali di questi è un requisito non funzionale:

 Il sistema deve visualizzare l’ora in base alla sua locazione

 Il tempo di risposta deve essere meno di un secondo

 Il linguaggio di implementazione deve essere Java

44. Cosa si intende per correttezza dei requisiti?

 I requisiti rappresentano la vista dell’utente

 Sono descritti tutti i possibili scenari del sistema

 Non ci sono requisiti funzionali e non funzionali che si contraddicono

45. Cosa si intende per completezza dei requisiti?

 I requisiti rappresentano la vista dell’utente

 Sono descritti tutti i possibili scenari del sistema

 Non ci sono requisiti funzionali e non funzionali che si contraddicono

46. Cosa si intende per consistenza dei requisiti?

 I requisiti rappresentano la vista dell’utente

 Sono descritti tutti i possibili scenari del sistema

 Non ci sono requisiti funzionali e non funzionali che si contraddicono

47. Cosa si intende per chiarezza dei requisiti?

 Non ci sono ambiguità nei requisiti

 I requisiti possono essere implementati e rilasciati

 Non ci sono requisiti funzionali e non funzionali che si contraddicono

48. Cosa si intende per realismo dei requisiti?

 Non ci sono ambiguità nei requisiti

 I requisiti possono essere implementati e rilasciati

 Per ogni funzione del sistema è possibile individuare un insieme di requisiti funzionali

49. Cosa si intende per tracciabilità dei requisiti?

 Non ci sono ambiguità nei requisiti

 I requisiti possono essere implementati e rilasciati

 Per ogni funzione del sistema è possibile individuare un insieme di requisiti funzionali

50. Quale di questi requisiti vanno affrontati durante il design ma non durante

l’implementazione?

 Requisiti ad alta priorità

 Requisiti a media priorità

 Requisiti a bassa priorità

51. Cosa si intende per interface engineering?

 Lo sviluppo parte da zero, non esiste un sistema precedente

 Riprogettazione o reimplementazione di un sistema esistente con nuove tecnologie

 Fornire i servizi di un sistema esistente in un nuovo ambiente operativo

52. Che tipo di scenari sono usati per descrivere un sistema futuro?

 as-is scenarios

 visionary scenarios

 training scenarios

53. Che tipo di scenari sono usati per descrivere un sistema esistente?

 as-is scenarios

 visionary scenarios

 training scenarios

54. Che tipo di scenari sono usati per guidare un utente nuovo nell’utilizzo del sistema?

 as-is scenarios

 visionary scenarios

 training scenarios

55. Perché si usano gli scenari?

 Per individuare i requisiti del sistema

 Per analizzare le associazioni tra gli oggetti del sistema

 Per analizzare il flusso di eventi di un caso d’uso

56. Come si descrive un caso d’uso?

 Mediante uno scenario

 Mediante un flusso di eventi

 Mediante un insieme di requisiti funzionali

57. Cosa esprime una relazione di inclusione tra casi d’uso?

 Una variante del normale flusso di eventi di un caso d’uso

 Una specializzazione di un caso d’uso

 Una decomposizione funzionale di un caso d’uso

58. Cosa esprime una relazione di estensione tra casi d’uso?

 Una variante del normale flusso di eventi di un caso d’uso

 Una specializzazione di un caso d’uso

 Una decomposizione funzionale di un caso d’uso

59. Cosa esprime una relazione di generalizzazione tra casi d’uso?

 Una variante del normale flusso di eventi di un caso d’uso

 Una specializzazione di un caso d’uso

 Una decomposizione funzionale di un caso d’uso

60. Un oggetto che interagisce con un attore in un sequence diagram è:

 Un entity object

 Un boundary object

 Un control object

61. Quale di queste categorie di requisiti non funzionali indica la facilità di cambiamenti al

sistema dopo il rilascio ?

 Reliability

 Performance

 Supportability

62. Quale di queste categorie di preudo-requirements indica vincoli legati alle modalità di

rilascio del sistema?

 Interface requirements

 Packaging requirements

 Operations requirements

63. Una trasformazione applicata agli oggetti di una classe e definita in fase di analisi è

chiamata:

 Operazione

 Signature

 Metodo

64. Una connessione tra due istanze di oggetti è chiamata:

 Associazione

 Link

 Relazione

65. Una associazione è:

 Una connessione tra due istanze di oggetti

 Una dipendenza tra classi

 Un mapping bidirezionale tra classi

66. Quale di questi tipi di oggetti cambia meno frequentemente in un sistema software ?

 Boundary object

 Control object

 Entity object

67. Cosa indica un ruolo ?

 Una estremità dell’associazione

 La navigabilità dell’associazione

 La direzione del nome dell’associazione

68. Cosa indica la direzione di una associazione ?

 Una estremità dell’associazione

 La navigabilità dell’associazione

 La direzione del nome dell’associazione

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 69. Nel diagramma seguente, “filename” | Directory | filename | 1 0..1 | File |  |
|  |
|  E’ un attributo della classe directory   E’ un attributo della classe file   E’ il nome della associazione | | | | | |

70. Cosa indicano le frecce in un diagramma di sequenza ?

 Eventi inviati da un oggetto ad un altro

 Relazioni tra oggetti

 Dipendenze tra oggetti

71. Cosa indica la linea tratteggiata verticale sotto un oggetto in un diagramma di sequenza ?

 L’invio di un messaggio all’oggetto

 Il periodo di attivazione di una operazione dell’oggetto

 Il periodo durante il quale è possibile inviare un messaggio all’oggetto

72. Cosa indica un rettangolo verticale sotto un oggetto in un diagramma di sequenza ?

 L’invio di un messaggio all’oggetto

 Il periodo di attivazione di una operazione dell’oggetto

 Il periodo durante il quale è possibile inviare un messaggio all’oggetto

73. Cosa dovrebbe contenere la prima colonna di un diagramma di sequenza ?

 L’attore che inizia il caso d’uso

 Un boundary object

 Il control object che gestisce il caso d’uso

74. Cosa dovrebbe contenere la seconda colonna di un diagramma di sequenza ?

 L’attore che inizia il caso d’uso

 Un boundary object

 Il control object che gestisce il caso d’uso

75. Cosa dovrebbe contenere la terza colonna di un diagramma di sequenza ?

 L’attore che inizia il caso d’uso

 Un boundary object

 Il control object che gestisce il caso d’uso

76. Quali di queste affermazioni non è corretta?

 Un boundary object accede ad un entity object

 Un entity object accede ad un control object

 Un control object crea un boundary object

77. In un diagramma di stato, quando viene valutata la condizione di guardia su una transizione

?

 Quando il sistema si trova nello stato sorgente della transizione

 Quando si verifica l’azione indicata sulla transizione

 Quando si verifica l’evento indicato sulla transizione

78. Quale di queste affermazioni non si riferisce ad un’attività in un diagramma di stato ?

 Operazione che prende tempo per essere completata

 E’ associata ad un evento

 E’ associata ad uno stato

79. Quale di queste affermazioni non si riferisce ad un’azione in un diagramma di stato ?

 Operazione che prende tempo per essere completata

 E’ associata ad uno stato

 E’ associata ad un evento

80. Quale di queste affermazioni si riferisce ad un diagramma di stato ?

 Serve ad individuare le relazioni temporali tra oggetti nel tempo

 Serve ad individuare la sequenza di operazioni come risposta ad uno o più eventi

 Serve ad individuare i cambiamenti degli oggetti nel tempo

81. Quale di queste affermazioni non si riferisce ad un diagramma di sequenza ?

 Serve ad individuare i cambiamenti degli oggetti nel tempo

 Serve ad individuare le relazioni temporali tra oggetti nel tempo

 Serve ad individuare la sequenza di operazioni come risposta ad uno o più eventi

82. Quale di questi prodotti dell’analisi dei requisiti vanno usati durante la fase di definizione

degli obiettivi di design ?

 Modello dei casi d’uso

 Modello a oggetti

 Requisiti non funzionali

83. Quale di questi prodotti dell’analisi dei requisiti vanno usati durante la fase di

decomposizione del sistema ?

 Modello dei casi d’uso

 Modello a oggetti

 Requisiti non funzionali

84. Quale di questi prodotti dell’analisi dei requisiti vanno usati durante la fase di design

relativa a mapping hardware/software e a gestione dei dati persistenti ?

 Modello dei casi d’uso

 Modello a oggetti

 Requisiti non funzionali

85. Come viene rappresentato un sottosistema in UML ?

 Con una classe

 Con un caso d’uso

 Con un package

86. Cosa è un servizio di un sottosistema ?

 Un insieme di operazioni con signature completamente specificata

 Un gruppo di operazioni che condividono uno scopo comune

 Un insieme di associazioni, eventi e vincoli legati tra di loro

87. Cosa è l’interfaccia di un sottosistema ?

 Un insieme di operazioni con signature completamente specificata

 Un gruppo di operazioni che condividono uno scopo comune

 Un insieme di associazioni, eventi e vincoli legati tra di loro

88. Quali di queste affermazioni è vera ?

 L’accoppiamento misura le dipendenze tra le classi di un sottosistema

 In un sistema con elevato accoppiamento le modifiche ad un sottosistema hanno forte

impatto sugli altri sottosistemi

 L’obiettivo del system design è massimizzare l’accoppiamento

89. Quali di queste affermazioni è vera ?

 La coesione misura le dipendenze tra i sottosistemi di un sistema

 In sottosistemi con elevata coesione le modifiche ad un sottosistema hanno forte impatto

sugli altri sottosistemi

 Le classi di un sottosistema con elevata coesione eseguono task simili

90. A cosa servono le partizioni di un sistema ?

 A dividere verticalmente un sistema in sottosistemi debolmente accoppiati

 A dividere orizzontalmente un sistema in sottosistemi a diversi livelli di astrazione

 A dividere un sistema sia orizzontalmente che verticalmente in sottosistemi indipendenti

91. A cosa servono i layer di un sistema ?

 A dividere verticalmente un sistema in sottosistemi debolmente accoppiati

 A dividere orizzontalmente un sistema in sottosistemi a diversi livelli di astrazione

 A dividere un sistema sia orizzontalmente che verticalmente in sottosistemi indipendenti

92. A chi fornisce servizi un layer di un sistema ?

 Ai layer di livello più basso

 Ai layer dello stesso livello

 Ai layer di livello più alto

93. In un’architettura software aperta:

 I sottosistemi di un livello possono accedere solo ai sottosistemi del livello immediatamente

inferiore

 I sottosistemi di un livello possono accedere ai sottosistemi di qualunque livello inferiore

 I sottosistemi di un livello possono accedere sia ai sottosistemi dei livelli inferiori che a

quelli dei livelli superiori

94. In un’architettura software chiusa:

 I sottosistemi di un livello possono accedere solo ai sottosistemi del livello immediatamente

inferiore

 I sottosistemi di un livello possono accedere solo ai sottosistemi dello stesso livello

 I sottosistemi di un livello possono accedere ai sottosistemi di qualunque livello inferior

95. Quali sono gli obiettivi di design di un’architettura software aperta ?

 Manutenibilità

 Efficienza

 Affidabilità

96. Quali sono gli obiettivi di design di un’architettura software chiusa ?

 Usabilità

 Efficienza

 Manutenibilità

97. Quale di queste affermazioni è falsa?

 In un’architettura client-server, il server fornisce i servizi al client

 In un’architettura client-server, il server conosce l’interfaccia del client

 Una repository architecture è un caso particolare di architettura client-server

98. Quali di queste affermazioni è vera ?

 In un’architettura peer-to-peer non c’è differenza tra client e server

 Un’architettura peer-to-peer favorisce una gestione centralizzata dei dati

 Un’architettura peer-to-peer consente di evitare problemi di deadlock

99. Un’architettura model-view-controller, il sottosistema “model”:

 E’ responsabile della conoscenza del dominio applicativo

 E’ responsabile della visualizzazione degli oggetti del dominio applicativo

 E’ responsabile della sequenza dei interazioni con l’utente

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 100. Dire quali di questi diagrammi model-view-controller è corretto: | | | | | | | |
|  | Controller | | initiator | | | | |
| 1  \* | | | notifier | |
|  | | | | | Model | |  |
| 1 | | | | | | repository | |
|  | | View | | subscriber | |
| \* | | | |
|  | | | | | | | |
|  | Controller | | initiator | | | | |
| \* 1 | | | repository | |
|  | | | | | Model | |  |
| 1 | | | | | | notifier | |
|  | | View | | subscriber | |
| \* | | | |
|  | | | | | | | |
|  | Controller | | initiator | | | | |
| \* 1 | | | repository | |
|  | | | | | Model | |  |
| 1 | | | | | | subscriber | |
|  | | View | | notifier | |
| \* | | | |
|  | | | | | | | |

101. Le relazioni tra i componenti in un component diagram sono

 associazioni

 dipendenze

 generalizzazioni

102. Quale di queste affermazioni relative ad un component diagram è falsa ?

 Mostra il mapping hardware/software

 Mostra la struttura del sistema a compilation time

 Mostra la struttura a design time

103. Quale di queste affermazioni relative ad un deployment diagram è falsa ?

 Mostra la struttura del sistema a run-time

 Mostra il mapping hardware/software

 Mostra le dipendenze tra componenti e interfacce dei sottosistemi

104. Le relazioni tra i nodi di un deployment diagram sono:

 associazioni

 dipendenze

 generalizzazioni

105. In un thin client model

 Il sistema è two thier e la logica applicativa è eseguita sul server

 Il sistema è two thier e la logica applicativa è eseguita sul client

 Il sistema è three thier e la logica applicativa è eseguita sul client

106. Se il mio obiettivo è fornire implementazioni diverse per uno stesso sottosistema

 Uso un bridge pattern

 Uso un adapter pattern

 Uso un façade pattern

107. Se il mio obiettivo è quello di realizzare una architettura chiusa

 Uso un adapter pattern

 Uso un bridge pattern

 Uso un façade pattern

108. Un invariante è

 Un predicato che deve essere vero prima dell’invocazione di un metodo di una classe

 Un predicato che è vero dopo l’invocazione di un metodo di una classe

 Un predicato che è vero prima e dopo l’invocazione di un metodo di una classe

109. Quale di queste affermazioni relative a JavaDoc è corretta ?

 Il commento che precede un metodo consente di specificare la precondizione del metodo

 Il commento che precede un metodo consente di specificare la postcondizione del metodo

 Il commento che precede un metodo consente di specificare l’invariante della classe

110. Se devo realizzare una associazione qualificata uso come struttura dati

 Una tabella

 Un insieme

 Una lista

111. Se devo estendere una classe mi interessa conoscere

 Tutti i membri privati, protetti e pubblici della classe

 Solo i membri protetti e pubblici della classe

 Solo i membri pubblici della classe

112. La realizzazione delle associazioni del modello a oggetti

 E’ una trasformazione del modello a oggetti

 E’ una trasformazione di tipo forward engineering

 una trasformazione di refactoring

113. Se devo memorizzare dati voluminosi e necessari per un periodo breve di tempo

 uso una struttura dati in memoria

 uso un file

 uso un database

114. Se devo effettuare una trasformazione prima sul codice sorgente e poi sul modello a oggetti

faccio

 Prima operazioni di forward engineering e poi di refactoring

 Prima operazioni di object model transformation e poi di forward engineering

 Prima operazioni di refactoring e poi di reverse engineering

115. Se eredito un contratto in UML quale di queste affermazioni non è corretta ?

 La precondizione del metodo nella sottoclasse consente al metodo di gestire meno casi del

corrispondente metodo nella superclasse

 Il metodo nella sottoclasse deve assicurare la stessa post-condizione del corrispondente

metodo nella superclasse

 L’invariante della sottoclasse può essere più restrittivo dell’invariante della superclasse

116. Quali di questi operazioni non è opportuno implementare durante la realizzazione di

contratti ?

 Controllare che la precondizione all’inizio di un metodo sia soddisfatta

 Controllare che la postcondizione all’uscita di un metodo sia soddisfatta

 Incapsulare il codice di controllo in metodi separati ai fini del riuso

117. L’evento percepito dall’utente come differenza tra comportamento atteso di un sistema

software e comportamento esibito dal sistema software viene denominato

 Failure

 Fault

 Error

118. Nel testing di integrazione bottom-up ho bisogno di realizzare

 Test Driver

 Test Stub

 Sia test driver che test stub

119. Quale di queste affermazioni è falsa ?

 Nel testing white box la derivazione dei casi di test mira a coprire la maggior parte di una

classe di elementi della struttura del codice

 Nel testing white box il numero di casi di test da eseguire dipende dal criterio di copertura

adottato

 Nel testing white-box non si controlla se l’output prodotto in corrispondenza di un input è

uguale all’output atteso

120. Quando si effettua il testing di regressione ?

 Dopo aver effettuato il test di integrazione e prima di effettuare il test di sistema

 Dopo aver effettuato il testing di sistema e prima di effettuare il testing di accettazione

 Dopo aver effettuato delle modifiche al codice per correggere un fault

1. Quale delle seguenti affermazioni è corretta:

• Il sequence diagram è adatto alla comunicazione con il cliente

• Il sequence diagram è più intuitivo degli use case

• V Il sequence diagram fornisce una prospettiva diversa che consente di individuare oggetti mancanti e aree non chiare nelle specifiche

1. (Sequence Diagram) quali di queste affermazioni non sono corrette:

• V Gli oggetti sono attivati quando vengono istanziati

• Una classe stabilisce i messaggi a cui un oggetto può rispondere

• V I messaggi sono inviati da oggetti

1. Un class diagram è tipicamente usato per modellare:

• Le relazioni tra oggetti dei sequence diagram individuati

• V Il glossario di un sistema: sono prese decisioni relativamente alle astrazioni da considerare

• V Lo schema concettuale di un database

1. Quale/i delle seguenti affermazioni è/sono corretta/e? In una architettura client/server le funzioni eseguite dal server di database sono:

• Gestione decentralizzata dei dati

• V Garantire la sicurezza del Database

• V Gestire la concorrenza delle operazioni (multiple user access)

• Elaborazioni decentralizzate

1. Un Boundary use case descrive (indicare la/e risposte corrette);

• V La configurazione del sistema

• V Le scelte relative allo sturtup, allo shutdown

• Le condizioni limite

1. In che modo possono essere specificati i contratti in UML?

• In Javadoc

• V La Object Contract Language

1. Se devo effettuare una trasformazione prima sul modello a oggetti e poi ottengo il codice, faccio:

• Prima operazioni di forward engineering e poi di refactoring

• V Prima operazioni di object model transformation e poi di forward engineering

• Prima operazioni di refactoring e poi di reverse engineering

1. Durante l’ispezione ai fini del fault detection (indicare le risposte corrette):

• Lo sviluppatore presenta gli artefatti

• Si controlla l’efficienza degli algoritmi con le richieste funzionali

• V Lo sviluppatore interviene solo se si richiedono chiarimenti

1. Il testing mira a valutare l’aderenza del sistema ai requisiti non funzionali viene denominato:

• Integration testing

• Acceptation testing

• V Performance testing

**10. Quale di queste affermazioni è vera:**

• Il testing white-box consente di capire se non ho implementato delle funzionalità

• Il testing black-box consente di individuare caratteristiche non richieste

• V Il testing black-box consente di capire se non ho implementato delle funzionalità

**11. Quale delle seguenti affermazioni è/sono corretta/e?**

• Un programma è esercitato da un caso di test (insieme di dati di input)

• Un test è formato da un caso di test

• V L’esecuzione del test consiste nell’esecuzione del programma per tutti i casi di test

• Un test ha successo se non rileva malfunzionamenti del programma

**12. Quale di queste affermazioni è vera?**

• Con il top down integration testing ho bisogno di driver

• V Con il bottom up integration testing ho bisogno di driver

• Con il sandwich testing non ho bisogno di driver

**13. Quale di queste affermazioni relativa ad un package in UML è falsa ?**

-Consente di decomporre un sistema in sottosistemi

-Consente di raggruppare classi Java

-E’ un meccanismo per strutturare e migliorare la leggibilità dei modelli

**15. Una entità fuori dal sistema che si sta modellando è**

-Una classe

-Una istanza

-Un attore

**16.Cosa si intende per backward traceability dei requisiti?**

-Il sistema garantisce la compatibilità con il sistema precedente

-Dato un requisito funzionale è possibile individuare le funzioni del sistema che lo realizzano

-Per ogni funzione del sistema è possibile individuare un insieme di requisiti funzionali

**18. A cosa serve un qualifier?**

-A specificare la molteplicità di una associazione

-A ridurre la molteplicità di una associazione

-A identificare una estremità di una associazione

**19. Cosa è un Lower-CASE?**

-Uno strumento per la generazione automatica di interfacce utente

-Uno strumento che supporta le fasi di programmazione, testing e debugging

-Uno strumento che supporta le attività delle fasi di analisi dei requisiti e di progettazione

**20. Quale di queste affermazioni relativa ad un package in UML è falsa?**

-Consente di decomporre un sistema in sottosistemi

-Consente di raggruppare classi Java

-E’ un meccanismo per strutturare e migliorare la leggibilità dei modelli

**21. Come viene rappresentato un invio condizionato di un messaggio in un diagramma di sequenza UML?**

-Una condizione racchiusa tra parentesi angolari precede il nome del messaggio

-Una condizione racchiusa tra parentesi quadre precede il nome del messaggio

-Un asterisco precede il nome del messaggio

**22. Quale di queste affermazioni non è valida per una struttura organizzativa di reporting gerarchica?**

-Lo stato è riportato dall’alto verso il basso

-Le decisioni vengono comunicate dall’alto verso il basso

-Le informazioni relative a stato e decisioni sono unidirezionali

**23. Cosa si intende per forward traceability dei requisiti?**

-Il sistema garantisce la compatibilità con il sistema precedente

-Dato un requisito funzionale è possibile individuare le funzioni del sistema che lo realizzano

-Per ogni funzione del sistema è possibile individuare un insieme di requisiti funzionali

**24. –**

**Serve ad individuare i cambiamenti degli oggetti nel tempo**

-Serve ad individuare le relazioni temporali tra oggetti nel tempo

-Serve ad individuare la sequenza di operazioni come risposta ad uno o più eventi

**25. Cosa non va individuato in fase di analisi dei requisiti?**

-Le classi del dominio applicativo

-Le classi del dominio delle soluzioni

-Le operazioni delle classi

**26. Se devo realizzare un’associazione qualificata, quale struttura dati Java utilizzo?**

-Una mappa (Map)

-Un insieme (Set)

-Una lista (List)

**27. Se voglio realizzare l’associazione uno-a-molti “ha-superato” tra le classi Studente ed Esame (uno studente può aver sup erato più esami), quale mapping è più adatto?**

-Una Collection sulla classe Esame in cui memorizzare gli studenti che hanno superato l’esame ed un riferimento sulla classe Studente in cui memorizzare l’esame superato

-Una Collection sulla classe Studente in cui memorizzare gli esami superati ed un riferimento sulla classe Esame in cui memorizzare lo studente che ha superato l’esame

-Una Collection sulla classe Esame in cui memorizzare gli studenti che hanno superato l’esame ed una Co llection sulla classe Studente in cui memorizzare gli esami superati

28**. Se voglio realizzare l’associazione molti-a-molti “ha-superato” tra le classi Studente ed Esame (uno studente può aver sup erato più esami e ciascun esame può essere stato superato da più studenti), quale mapping è più adatto?**

-Una Collection sulla classe Esame in cui memorizzare gli studenti che hanno superato l’esame ed un riferimento sulla classe Studente in cui memorizzare l’esame superato

-Una Collection sulla classe Studente in cui memorizzare gli esami superati ed un riferimento sulla classe Esame in cui memorizzare lo studente che ha superato l’esame

-Una Collection sulla classe Esame in cui memorizzare gli studenti che hanno superato l’esame ed una Collection sulla classe Studente in cui memorizzare gli esami superati

**29. Quale di queste affermazioni non è appropriata per il modello di processo a spirale?**

-Per ogni sistema è possibile utilizzare diversi modelli di processo

-Non è adatto per la manutenzione del software

-è difficile definire il contratto

**Quale di queste affermazioni non è vera?**

-I prototipi possono essere utili per ridurre i rischi del progetto

-Il modello a cascata prende in considerazione la gestione sistematica dei rischi del progetto

-Il rischio è legato alla qualità e alla quantità di informazioni possedute

**Quale di queste affermazioni non è vera?**

-Il cammino critico è formato da attività che sono particolarmente complesse

-Il ritardo di un attività sul cammino critico ritarda l’intero progetto

-Ogni attività sul cammino critico ha uguale minimo tempo di completamento e massimo tempo di completamento

**Quale di queste affermazioni non è vera?**

-Nel forwarding engineering si parte dal codice per derivare i modelli del software

-nel greenfield engineering si parte da zero non esiste un sistema precedente

-Il software re-engineering è realizzato per ridurre i costi di manutenzione

**Quale di queste affermazioni è vera?**

-Un deliverable è la descrizione di un task

-Un wordpackage è la descrizione di un milestone

-Una attività può essere scomposta in task che possono essere svolti in parallel

-Un task dura per tutta la durata del Progetto

**Quale di queste affermazioni non è vera?**

-Il modello a cascata consente di ridurre il rischio di overrun

-Il modello a cascata non ha una buona visibilità

-Nel modello a cascata le fasi del processo sono in progressione sequenziale

**Quale di queste affermazioni è vera?**

-Lo sviluppo esplorativo è adatto per sistemi interattivi a vita lunga

-Il processo di sviluppo esplorativo non consente di ridurre il rischio legato alla cattiva comprensione dei requisiti

-nello sviluppo esplorativo vengono costruite varie versioni del sistema sottoposte a validazione da parte dell’utente

**Quale di queste affermazioni non è vera?**

-La coesione misura le dipendenze tar i sottosistemi di un sistema

-In sottosistemi con elevata coesione le modifiche ad un sottosistema hanno forte impatto sugli altri sottosistemi

-Le classi di un sottosistema con elevata coesione eseguono task simili/strettamente collegati

**Dall’informazione della matrice di tracciabilità in cui ho collegato i requisiti ai relativi test case**

-posso ricavare quali requisiti sono stati testate

-posso ricavare quali requisiti non sono stati realizzati

-posso determinare se ho realizzato un testing white-box o black-box

**Quale di queste affermazioni è vera?**

-in un’architettura a repository il client rappresenta il collo di bottiglia

-in un’architettura a repository è possibile aggiungere un numero qualsiasi di client se il repository conosce l’interfaccia del client

-un’architettura a repository consente una gestione centralizzata dei dati

**Quale di queste affermazioni è vera?**

-il ruolo del project manager in SCRUM è proposto a facilitare e proteggere il team

-in SCRUM requisiti non possono cambiare durante lo sprint

-in SCRUM il focus factor viene utilizzato per definire la team velocity (use case diagram)

**Quale di queste affermazioni non è vera?**

-è possibile definire una relazione di estensione tra attori

-una relazione di estensione tra casi d’uso consente di esprimere casi d’uso per la gestione delle boundary condition

-è possibile definire una relazione di estensione tra attore e caso d’uso

**Quale tipo di prototipazione parte con i requisiti meglio compresi ?**

-Prototipazione esplorativa

-Throw-away

-Mock-ups

**Cosa non va individuato in fase di analisi dei requisiti ?**

-Le classi del dominio applicativo

-Le classi del dominio delle applicazioni

-Le operazioni delle classi

**Qual è l’obbiettivo della prototipazione della prototipazione throw-away?**

-Costruire un sistema in modo rapido

-Ridurre i costi di sviluppo e manutenzione

-Avere una chiara comprensione dei requisiti

**Che tipo di relazione puo’ esistere tra due attori in un diagramma dei casi d’uso?**

-Associazione

-Dipendenza

-Generalizzazione

**Cosa si intende per consistenza dei requisiti?**

-i requisiti rappresentano la vista dell’utente

-sono descritti tutti i possibili scenari del sistema

-non ci sono requisiti che si contraddicono

**Cosa esprime una relazione di estensione tra casi d’uso?**

-una variante del normale flusso di eventi in un caso d’uso

-una specializzazione di un caso d’uso

-una decomposizione funzionale di un caso d’uso

**Quando si effettua il testing di integrazione?**

-Dopo aver effettuato il testing di sistema e prima di effettuare il testing di accettazione

-Dopo aver effettuato delle modifiche al codice per coorreggere un fault

-Dopo aver effettuato il test di unità e prima di effettuare il test di sistema

**Quale di queste affermazioni è vera?**

-Con il top down integration testing ho bisogno di driver

-con il bottom up integration testing ha bisogno di driver

-con il sandwitch testing non ho bisogno di driver