Prüfungsvorbereitungsfragen SE2

Woche 1

Projektmanagement

1. Erklären Sie den Satz „Projektmanagement ist keine Insel“.

Woche 2

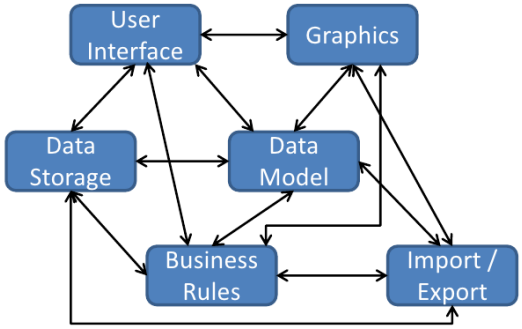
Build Server / Automation

2. Warum braucht es Buildserver?
3. Nennen sie drei Anforderungen an ein Build Tool.
4. Was sind CRISP Builds?

Woche 3

Best Practices

5. Nennen Sie einige „good practices“ beim Erstellen von Requirements.
6. Warum soll Qualität eine Anforderung sein, und wie soll sie spezifiziert werden?
7. Wie gehen Sie mit Änderungen um?
8. Was ist Achieve Orthogonality? Wo liegen die Vorteile?
9. Wo sind die Probleme bei solchen Strukturen?

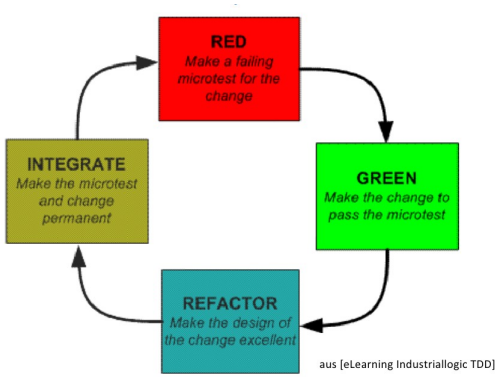


10. Was meint das Prinzip „Fix broken windows“?
11. Wann soll refactort werden? Vorgehen?
12. Nennen sie „good practices“ bei Reviews.

Woche 4

TDD & Refactoring

13. Nennen Sie die vier Schritte des Grundprinzips von Test Driven Development.
14. Zeigen sie auf, wo TDD und agile Entwicklung Hand in Hand gehen, und wo man aufpassen muss.
15. Erklären Sie folgende Grafik:



aus [eLearning Industriallogic TDD]

16. Erklären Sie die TDD Patterns „Specify It“, „Frame It“ und „Evolve It“.
17. Was ist gemeint mit mit den Sätzen „Do the simplest thing that could possibly work“ und „Brake it to make it“?

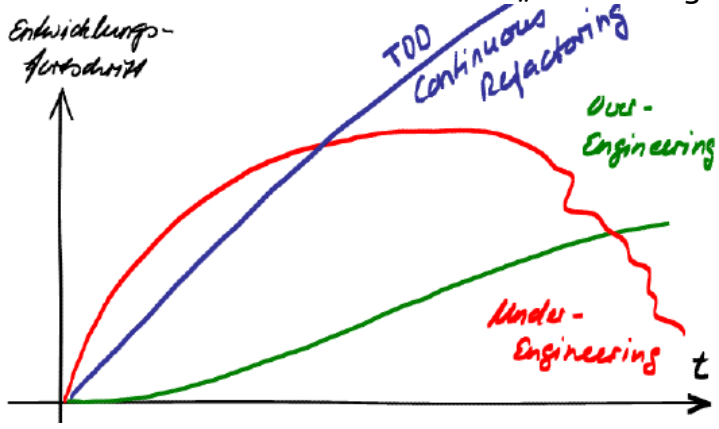
Woche 5

Refactoring

18. Erklären Sie die Vorbedingungen und 4 Schritte des Refactoring Prozesses.
19. Warum sollen beim Refactoring immer nur kleine Schritte gemacht werden, und nach jedem Schritt die Tests neu ausgeführt werden?
20. Warum soll nie gleichzeitig Refactoring betrieben und neue Funktion hinzugefügt werden?
21. Beschreiben Sie die folgenden Refactoring Kategorien:
    Composing Methods, Moving Features between Classses, Organizing Data, Simplifying Conditional Expressions, Making Method Calls simpler, Dealing with Generalization. Nennen Sie zu jeder Kategorie klassische Beispiele.
    (Tipp: Folien 05-Refactoring/06_Refactoring-Teil2-CodeSmells-Refactorings.pdf, Folie 9-17 verwenden da an der Prüfung auch zugelassen)
22. Beschreiben Sie einige Code Smells. Verwenden Sie auch hierzu die Folien.

Woche 6

Refactoring

23. Erklären Sie die beiden Ansätze von „Refactoring to Patterns“.
24. 

25. Nehmen Sie zu diesem Diagramm Stellung. Was passiert beim Over- und Underengineering? Was sind jeweils die Folgen für die Zukunft des Projektes. Beschreiben Sie den „blauen Mittelweg“.
26. Erklären Sie den folgende Code Smells und die Refactorings dazu
    1. „Combinatorial Explosion“
    2. „Conditional Complexity“
    3. „ Indecent Exposure“
    4. „ Oddball Solution“
    5. „ Solution Sprawl“
27. Wie können Sie Singletons mit Inline Singletons entschärfen?

Fit & Fitnesse

28. Auf welchen Abstraktionsebenen testen Sie und wer testet auf welcher Ebene?
29. Auf welcher Ebene werden Black- / Whiteboxtest gemacht, wann wird Junit, wann Fitnesse eingesetzt? Warum?
30. Welches sind die Vorteile von Fitnesse gegenüber Junit? Was testet Fitnesse?
31. Warum sollten Sie die Businesslogik unabhängig vom GUI Testen?
32. Wie spezifizieren Sie in Fitness einen Test?
33. Was ist ein Test Fixture?

Woche 7

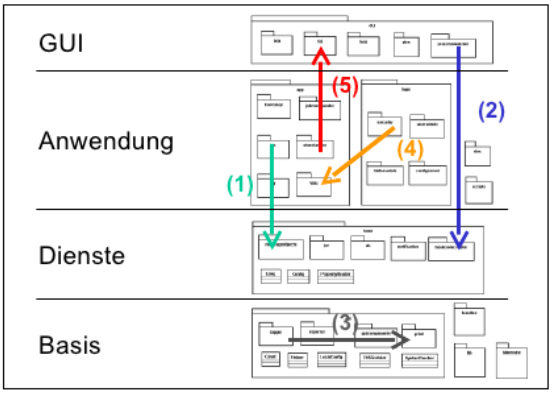
Aufwandschätzung

34. Nennen Sie vier Gründe, warum wir beim Schätzen daneben liegen.
35. Nennen Sie die drei wichtigsten Einflussfaktoren in einem Softwareprojekt. Nennen Sie die vier wichtigsten Einflussfaktoren nach COCOMO II und ISBSG.org.
36. Welche vier grundsätzlichen Faktoren ziehen Sie zur Aufwandschätzung heran?
37. Erklären Sie die Definition von LOC.
38. Erklären Sie grob, wie sich die Kosten eines Projektes auf die Projektphasen verteilt.
39. Erklären Sie Bottom-Up und Top-Down Schätzungen.
40. Nennen Sie mindestens fünf Dinge, die gerne vergesssen gehen beim Schätzen des Aufwandes.
41. Warum sollten Sie immer drei Schätzungen machen?

Woche 8

Architektur

42. Beschreiben Sie das klassische Schichtenmodell nach Larmen
43. Warum sind Schichten asymetrisch, Tiers symetrisch? Wie läuft die Kommunikation?
44. Was sind Partitionen?
45. Beschreiben Sie die 5 Arten von Abhängigkeiten. Welche Arten sind zu vermeiden / don'ts?



46. Was bedeutet „Separation of Concerns“?
47. Nennen Sie die drei Schritte der Datenmodellierung
48. Wie setzen Sie Vererbung ein?
49. Wie ordnen Sie ein Klassenmodell? Welche Vorteile bringt dies? Zählen Sie mindestens 5 auf.
50. Was unternehmen Sie, wenn Sie in ihrem Klassenmodell Rhomben, Doppelgabeln und Schleifen finden? Insbesonder Schleifen: Was können Schleifen nach sich ziehen?
51. Warum ist das Datenmodell so wichtig? (2:5:20)

Woche 9

Reviews

52. Besch reiben Sie die unterschiedlichen Auswirkungen, wenn ein Fehler während der Entwicklung oder wenn er beim Kunden gefunden wird.

53. Nennen Sie (ausser dem Obigen) zwei weitere Vorteile von Reviews.
54. Nennen Sie je ein Paar Ziele von Reviews für die folgenden Phasen: Requirements Specification, Architektur/Design, Code, Dokumentation
55. Warum sind Reviews die einzige Möglichkeit, Fehler in den Requirements und im Design zu finden?
56. Erklären Sie die folgenden Qualitäts Attribute von Requirements: Correct, Complete, Consistent, Concise, Traceable, Testable, Adequate.
57. Auf welchen Elementen / Dokumenten basiert ein Architektur/Design Review? Nennen Sie vier Punkte, auf die Sie besonders achten müssen bei Architektur/Design Reviews.
58. Nennen Sie vier Arten von Analysesoftware, die Sie beim Code Review unterstützen sollte.
59. Wie bereiten Sie sich auf ein Review vor?
60. Welche Vorbedingungen müssen für ein Review erfüllt sein?
61. Nennen Sie vier wichtige Fragen, denen Sie bei einem Review nachgehen sollten.
62. Was für Nacharbeiten sollten Sie für beim Review gefunden Fehler tun?

Woche 10

Design By Contract (DBC)

63. Was ist ein Contract? Was bedeutet es für Software?
64. Was bedeutet ein Contract konkret für Systemoperationen und Klassen? Was sind Klasseninvarianten?
65. Wie verhalten sich Contracts bei Vererbung in Bezug auf Pre- und Postconditions?
66. Welche Möglichkeiten haben Sie in Java, um Design by Contract umzusetzen?
67. Was können in Java Sie mit Assertions realisieren? Wie können Sie Zustände zum Vergleichen Sichern?
68. Nennen Sie gemeinsamkeiten und Unterschiede zu Unit Tests.
69. Nennen Sie einige Nutzen von „DBC“.

Woche 11

Produktmetriken

70. Erklären Sie, was eine Metrik und was ein Indikator ist.
71. Erklären Sie innere und äussere Merkmale. Nennen Sie je vier Beispiele.
72. Nennen Sie mindestens fünf Ziele von Metriken.
73. Nennen Sie fünf Typen von Source Code Metriken.
74. Erklären Sie die zyklomatische Zahl (McCabe Metrik).
75. Nennen Sie drei Typen von Source Code Metriken für Objektorientierte Software.
76. Erklären Sie NOC, NSC, NOI, DIT, NORM, NOM, NOF, TLOC, KLOC, MLOC, CC, WMC, LCOM
77. Was bringen Code Metriken?

Woche 12

Design Patterns

78. Nennen Sie drei Probleme, die Design Patterns mit sich bringen können.
79. Erklären Sie das Builder Pattern. Machen Sie ein Beispiel.
80. Erklären Sie den Unterschied zwischen dem Builder Pattern und der Abstract Factory.
81. Erklären Sie das Prototype Pattern. Machen Sie ein Beispiel.
82. Stellen Sie Prototyp- und Classorientierte Objektorientierung einander gegenüber.
83. Erklären Sie den Unterschied zwischen dem Prototype Pattern und der Abstract Factory.
84. Erklären Sie das Flyweight Pattern. Welche Möglichkeiten haben Sie, um den Zustand des Flyweight zu speichern.
85. Erklären Sie den Unterschied zwischen dem Flyweight Pattern und dem State Pattern.
86. Erklären Sie das Proxy Pattern. Machen Sie ein Beispiel. Nennen Sie vier Anwendungsfälle für das Proxy Pattern.

87. Wie unterscheidet sich das Proxy Pattern vom Adapter Pattern und dem Decorator Pattern.
88. Erklären Sie das Chain of Responsibility Pattern.
89. Erklären Sie den Uterschied zwischen dem Chain of Responsibility Pattern und dem Command Pattern.

Woche 13

Design Patterns

90. Erklären Sie das Interpreter Pattern. Machen Sie ein Beispiel.
91. Erklären Sie das Mediator Pattern. Nennen Sie einen Nachteil des Mediators.
92. Erklären Sie den Unterschied zwischen dem Mediator Pattern und dem Facade Pattern.
93. Erklären Sie das Memento Pattern. Erklären Sie eine Anwendung des Memento Patterns.
94. Erklären Sie, wie das Memento Pattern zusammen mit dem Command Pattern genutzt werden kann.
95. Erklären Sie das Visitor Pattern. Machen Sie ein Beispiel.
96. Erklären Sie die Vorteile des Visitor Patterns. Erklären Sie die Nachteile. Was passiert, wenn Sie die Objektstruktur anpassen müssen?
97. Erklären Sie die Unterschiede zwischen dem Visitor Pattern und dem Interpreter Pattern.
98. Visitor Pattern: Erklären Sie, wie die Fallunterscheidung durch Virtual Call und der Double Dispatch funktioniert.

Woche 14

Domain Specific Languages (DSL)

99. Erklären Sie, was Multi-Target-Code ist.
100. Erklären Sie, was eine interne und was eine externe DSL ist. Machen Sie je ein Beispiel.
101. Nennen Sie zwei Vor- und zwei Nachteile der internen DSL.
102. Erklären Sie, was Metaprogrammierung ist.
103. Erklären Sie grob wie Xtext funktioniert.