FS14 SE3 Repetitionsfragen

Antworten zu den Repetitionsfragen

Falls vorhanden befinden sich diese im GitHub Repository. Ergänzungen oder ganze Antwortensets sind jederzeit herzlich willkommen. https://github.com/moonline/HSR.modules.SE3

Symbolerklärung

[DP] Wissensfragen des Dozenten (Slides) für die Prüfung

[EW] Wissensfrage, die über den Stoff der Vorlesung hinaus geht.

1 Java Advanced

1.0.1. Wie wurden bis Java 1.4 Generische Schnittstellen Typisiert? Zu welchem Zeitpunkt wurde die Typensichereit sichergestellt? Durch wen? Welche negative Konsequenzen zog dies nach sich.

1.0.2. Welchen Vorteil bringen Generics bezüglich Sicherheit?

1.0.3. Sind List<String> und List<Object> kompatibel, sodass einer Liste von Object eine Liste von Strings zugewiesen werden kann?

1.0.4. Allgemein: Welche Bedingungen müssen erfüllt sein, um eine Variable a einer Variable b zuzuweisen zu können?

1.0.5. Warum wäre es extrem gefährlich, wenn einer List<Person> eine List<Student> zugewiesen werden könnte? Welche Konsequenzen würde dies nach sich ziehen?

1.0.6. Warum ist folgende Zuweisung legitim?

```
List<String> ls = new ArrayList<String>();
```

1.1 Wildcards

1.1.1 Was sind Wildcards und wozu dienen sie? Erklären Sie das zugrunde liegende Problem, das Wildcards notwendig macht.

1.1.2. Warum kann trotz Wildcard innerhalb einer Funktion mit Wildcard Parametern auf diesen keine Änderung vorgenommen werden.

1.1.3. Erlären Sie Upper Bound Wildcards und Lower Bound Wildcards. Wozu dienen sie? Machen Sie je ein Beispiel.

1.1.4. Welche Einschränken bringen Upper Bound Wildcards mit? Welche Lower Bound Wildcards?

1.2 Generische Methoden

1.2.1. Was sind generische Methoden? Welches Problem lösen sie?

1.3 Raw-Type

1.3.1. Warum gibt es von einer generischen Klasse nur eine Definition? Warum gibt es keine non-generic Version?

1.4 Erasure

1.4.1. Was ist ein Erasure? Was macht es?

1.4.2. Was sind Bridge Methoden? Warum braucht es sie?

1.4.3. Erklären Sie Class Sharing. Erklären Sie das Beispiel unten.

```
System.out.println(
        (new List<String>()).getClass() == (new List<Integer>().getClass())
); // true
```

1.4.4. Warum funktioniert "new T()" nicht? Welche zwei Alternativen gibt es? Nennen Sie Vor- und Nachteile.

2 Referenzen

2.0.1. Was passiert mit nicht mehr erreichbaren Objekten?

2.0.2. Erklären Sie die Zustände

a. created

b. in use

c. invisible

d. unreachable

e. collected

f. finalized

2.0.3. Was ist resurection? Was ist zu berücksichtigen?

2.1 Schwache Referenzen

2.1.1. Was sind schwache Referenzen?

2.1.2. Erklären Sie die folgenden Referenz-Typen:

a. Weak-Referenzen

b. Soft-Referenzen

c. Phantom-Referenzen

2.1.3. Erklären Sie die Erreichbarkeiten:

a. strongly reachable

b. softly reachable

c. weakly reachable

d. phantom reachable

e. unreachable

2.1.4. Was sind java.lang.ref.SoftReference's?

2.1.5. Was sind java.lang.ref.WeakReference's?

2.1.6. Wozu dient die ReferenceQueue?

2.1.7. Was sind java.lang.ref.PhantomReference's? Welche Eigenschaften weisst sie auf?

2.1.8. Erklären Sie die folgenden Methoden, die java.lang.Runtime zur Verfügung stellt:

a. gc()

b. runFinalization()

c. freeMemory()

d. totalMemory()

e. maxMemory()

f. addShutdownHook()

3 AOP

3.0.1. Was ist AOP? Welche Vorteile bietet es?

3.1 AspectJ

3.1.1. Was sind Pointcut, Advice und Aspect? Was sind JoinPoints?

3.1.2. Was ist der Unterschied zwischen execute() und call()?

3.1.3. Wie kann in einem around() auf die Parameter zugegriffen werden?

3.1.4. Wozu dient thisJoinPoint? Warum kann dies nicht über this erreicht werden?

3.1.5. Wozu dienen target() und args()? Machen Sie ein Beispiel.

3.1.6. Wie kombinieren Sie Pointcuts?

3.1.7. Welche Zugriffsrechte haben Sie auf die Felder der gewavten Klasse? Wie können Sie auf private member zugreifen?

4 Agile Programmierung

4.0.1. Erklären Sie die Begriffe "Prinzip", "Methode" und "Prozess".

4.0.2. Wie ist der RUP aufgebaut? Welche Dokumente werden üblicherweise erstellt?

4.1 Scrum

4.1.1. Was ist Scrum?

4.1.2. Wie ist Scrum aufgebaut? Machen Sie eine Skizze.

4.1.3. Wozu dient das Backlog? Was sind Product- und Release Backlog?

4.1.4. Welche Aufgabe hat der Product Owner?

4.1.5. Wozu dienen die "Sprint Planning Meetings"?

4.1.6. Wer ist der ScrumMaster? Was ist seine Aufgabe?

4.1.7. Was ist ein Sprint?

4.1.8. Wozu dienen "Daily Scrum Meetings" und was wird diskutiert?

4.1.9. Was ist das "Executable Increment"?

4.1.10. Welche Rollen gibt es bei Scrum?

4.2 Agile Prinzipien

4.2.1. Fassen sie die agilen Prinzipien kurz zusammen.

4.2.2. Nennen Sie die drei Prämissen agiler Software Entwicklung.

4.2.3. Wie funktioniert Agile Planung?

4.2.4. Was ist das Ziel jeder Iteration?

4.2.5. Wie wird der Fortschritt gemessen?

4.3 Agile Teams

4.3.1. Welche Rolen gibt es? Wie werden die Rollen wahrgenommen? Wer ist verantwortlich?

4.3.2. Nennen Sie einige Erfolgsfaktoren für Agile Teams.

4.4 Agile Projekte

4.4.1. Was ist der "Inception Desk"?

4.5 User Stories

4.5.1. Was sind User Stories? Wie werden sie notiert (Format)?

4.5.2. Nennen Sie einige Eigenschaften guter User Stories.

4.5.3. Erklären Sie den Unterschied zwischen User Storie und Use Case.

4.6 Aufwandschätzung

4.6.1. Wie soll zu Projektbegin geschätzt werden?

4.6.2. Wie funktioniert agiles schätzen?

4.6.3. Was ist Punkte basiertes schätzen und wie funktioniert es?

4.6.4. Wann sollte neu geschätzt werden und wann nicht?

4.7 Agile Planung

4.7.1. Wie unterscheidet sich agile von statischer Planung?

4.7.2. Wie funktioniert agile Planung?

4.7.3. Wie flexibel ist der Projektumfang? Was ist zu tun, wenn neue User Stories hinzukommen?

4.7.4. Listen Sie auf, aus welchen Schritten ein agiler Plan besteht.

4.7.5. Zeichnen Sie eine Burn-Down Chart, bei der in Iteration 2 und 5 je 5 Punkte neue Features hinzukommen. Nehmen Sie alle andern notwendigen Annahmen an.

4.8 Agile Methoden & Management

4.8.1. Erklären Sie die folgenden Methoden:

a. TDD

b. Refactoring

c. Continious Integration

4.8.2. Was ist Just-In-Time Analysis?

4.8.3. Was ist ein Iteration Planning Meeting und wie ist es aufgebaut?

4.8.4. Welche Vorarbeiten müssen geleistet sein, um beim IPM die nächste Iteration planen zu können und was wird konkret mit wem geplant?

4.8.5. Wozu dient die Mini Retrospektive?

4.8.6. Wozu dienen die Daily Stand-up Meetings und was wird besprochen?

5 Modellierung

5.0.1. [DP] Was sind Modelle und zu welchem Zweck werden Modelle erstelt?

5.0.2. Nennen Sie einige prägende Eigenschaften eines Modells.

5.0.3. [DP] Nennen Sie zu den folgenden Modellen Original, Zweck, Zielgruppe, Vereinfachung und Notation:

a. Technische Zeichnung eines Segelschiffs.

b. Einrichtungsplan eines Wohnung

c. Musiknoten

d. Elektrischer Schaltplan

5.0.4. [DP] Wo liegt der Unterschied zwischen Modellieren und Programmieren, bzw. zwischen einem Modell und einem Programm?

5.0.5. Wozu dient das Zachman Framework?

5.0.6. Nennen Sie einige Modellaspekte der SW Entwicklung.

5.0.7. Erklären Sie folgende Modelle

a. dynamische/statische

b. Blackbox/Glasbox

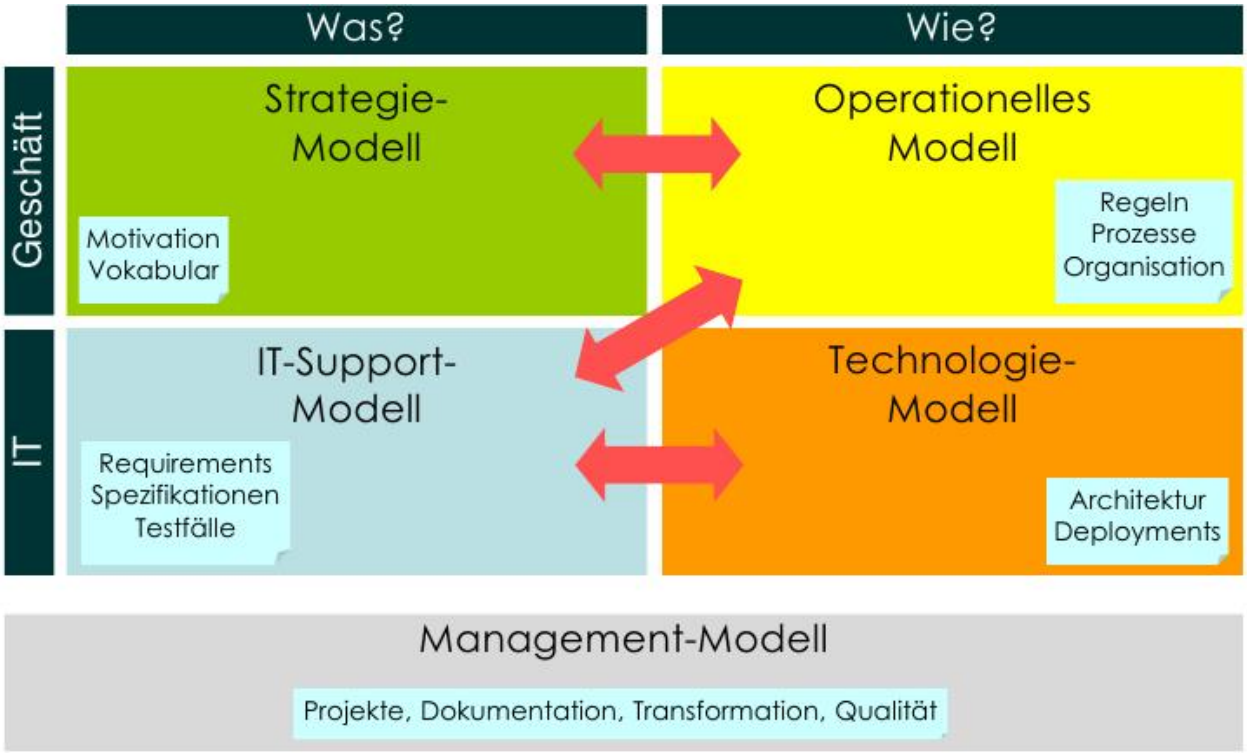
c. konstruktive/analytische

5.0.8. [DP] Was ist der Zusammenhang zwischen Eindeutigkeit, Vollständigkeit und Ausführbarkeit?

5.1 Model Driven Enterprise Engineering

5.1.1. Was ist die Idee hinter MDEE?

5.1.2. Erklären Sie die Grafik:



5.1.3. Was ist das BMM? Woraus setzt es sich zusammen?

5.1.4. Welche Arten von Geschäftsregeln gibt es?

5.1.5. Wozu dient SysML?

5.1.6. Was ist xUML? Welchen Vorteil bieten sie?

5.1.7. Was ist UTP?

6 Analysemuster

6.0.1. Wozu dient die Validierung eines Models? Wie wird sie gemacht?

6.0.2. Was ist ein Muster in eine Modell?

6.0.3. Was ist eine Master-Detail Beziehung?

6.0.4. Warum muss eine M:M Beziehung hinterfragt werden? Wo ist der Unterschied zwischen einer aufgelösten M:M Beziehung mit Linkklasse und mit zwischenObjekt?

6.0.5. Was ist eine Dreieck-Struktur uns warum muss sie hinterfragt werden? Was ist eine Viereck-Struktur?

6.0.6. Was ist eine Doppel-V Struktur? Warum muss sie hinterfragt werden? Wie kann sie aufgelöst werden? Wie können allfällig entstandene Dreiecke aufgelöst werden?

7 Aspektorientierte Analyse

7.0.1. Welches Problem lösen Aspekte gegenüber Vererbung/Mehrfachvererbung?

7.0.2. Was sind Aspekte? Welche vier Elemente umfasst ein Aspekt?

7.0.3. Was ist "Basic Existence"? Was ist der Sinn davon?

7.0.4. Was sind Subtypen/Supertypen von Aspekten und was sind Slices?

7.0.5. Was ist parallele Spezialisierung?

8 Metamodellierung

8.0.1. Was ist Metamodellierung?

8.0.2. Wozu dient Metamodellierung in UML?

8.0.3. Was sind UML Powertypes?

8.0.4. Skizzieren Sie die OMG Meta Layers?

8.0.5. Was ist ein UML Stereotyp?

8.0.6. Was ist UML Profiling?

8.0.7. Stellen Sie Profiling und Metamodel Modification einander gegenüber.

9 xUML

9.0.1. [DP] Was sind Use Cases? Was gehört in einen Use Case? Was nicht?

9.0.2. [DP] Wozu werden Use Cases gemacht?

9.0.3. [DP] Wie werden Use Cases schlussendlich in den Code abgebildet.

9.0.4. Erklären Sie die folgenden Elemente eines Use Cases: Intent, Precondtion, Postcondition, Normal interaction sequence, Alternative interaction sequence, user interface

9.0.5. Erklären Sie, auf welche Use Cases ein Actor zugriff hat, und auf welche nicht

9.0.6. Wie werden Use Cases zur Verifikation benutzt?

10 Verhaltensmodellierung

10.0.1. Was sind Ereignisse, Zustandsübergänge, Aktionen und Reaktionen eines Objektes?

10.0.2. Wozu dienen Zustandsdiagramme?

10.0.3. Wie hängen Zustandsdiagramme und Geschäftsregeln zusammen?

10.0.4. Was ist ein Zustand? Was ist ein interner Zustandsübergang?

10.0.5. Was ist ein Superzustand?

10.0.6. Was sind Parallelzustände?

10.0.7. Was sind Synchronisationspunkte?

10.0.8. Wie werden Übergangsbedingungen definiert?

10.0.9. Was sind Pseudo-Ereignisse?

11 Glass Box Modelling

11.0.1. Was ist XUML Black-Box modelling, was Glass-Box modelling?

11.0.2. Was sind die folgenden Elemente? Wie werden sie in UML und Java definiert?

a. Derivations

b. Action Rules

c. Constraints

12 Objectives Constraint Language

12.0.1. Was ist OCL? Wozu dient sie?

12.0.2. Was sind abgeleitete Attribute? Wie werden sie in UML, Java und C# umgesetzt?

12.0.3. Welche Rolle spielt der Kontext bei Constraints?

12.0.4. Erklären Sie den Unterschied zwischen einer Expression und einer Action anhand von if.

12.0.5. Was sind Conditional Expressions?

13 Geschäftsprozesse

13.0.1. Was sind Geschäftsprozesse?

13.0.2. Nennen Sie einige Gründe, warum Geschäftsprozesse überhaupt modelliert werden sollen?

13.0.3. Erklären Sie die Begriffe Akteur, Prozessor, Geschäftsaktivität und Workflow.

13.0.4. Erklären Sie den Unterschied zwischen AUfbauorganisation und Ablauforganisation.

13.0.5. Erklären Sie den Unterschied zwischen freien und restriktiven Prozessen. Welche lassen sich einfach automatisieren?

13.0.6. Erklären Sie einmahlige und repetitive Prozesse.

13.0.7. Erklären Sie den Unterschied und das Anwendungsgebiet von detailierten und leichten Prozessmodellen.

13.0.8. Was ist ein Geschäftsziel?

14 Business Rules Ansatz

14.0.1. Was ist der Business Rules Ansatz? Was sind Business Rules?

14.0.2. Was ist das Faktenmodell? Erklären Sie den Unterschied zum Klassenmodell.

14.0.3. Was sind Konzepte?

14.0.4. Erklären Sie den Unterschied zwischen Intensionaler und Extensionaler Definition.

14.0.5. Was sind Faktentypen? Wie unterscheiden sich unäre und binäre Faktentypen?

14.0.6. Wie werden Klassifikation und Generalisation abgebildet?

14.0.7. Was sind Ableitungsregeln?

14.0.8. Was sind Prozessregeln?

14.0.9. Erklären Sie die folgenden Eigenschaften von Regeln: volatilität, Durchsetzung, Gültigkeit, Verantwortliche(r), Quelle, Umsetzung, Motivation.

14.0.10. Was sind Textschablonen für Einschränkungen, Prozessregeln und Ableitungsregeln? Wie sind sie aufgebaut?

14.0.11. Wieso sind Textschablonen basierte Regeln maschinenlesbar, obwohl es sich um natürliche Sprache handelt?

14.0.12. In welchem Fall ist eine Entscheidungstabelle zu bevorzugen? Wie kann eine Entscheidungstabelle benutzt werden um Regeln zu überprüfen oder zu vereinfachen?

14.1 Implementation

14.1.1. Rule Engines können auf vier verschiedene Arten implementiert werden. Erklären Sie die drei Ansätze und nennen Sie Vor- und Nachteile.

14.1.2. Erklären Sie das Prinzip der Push- und Pull Schnittstelle. Wie wirkt sich dies auf die Testbarkeit aus?

15 Model Driven Architecture

15.0.1. Warum reicht UML/xUML nicht aus?

15.0.2. Was sind xUML compilers?

15.0.3. Erklären Sie die Abkürzungen: CIM, PIM, PM PSM, PSI.

15.0.4. Erklären Sie den Prozess vom Geschäft zum Code.

15.0.5. Wie funktioniert Modell-Evolution?

15.0.6. Wer macht welche Modelle? Welche Rollen gibt es?

15.0.7. Erklären Sie die Transformationen M2T, T2M, M2M.

15.0.8. Erklären Sie die beiden Varianten die es gibt, einer Transformationen Details hinzuzufügen. Nennen Sie Vor- und Nachteile.

15.0.9. Welche Probleme bringt Code-Generierung grundsätzlich mit?

15.0.10. Was ist MOF?