
Name Vorname Matrikelnummer

Universität Zürich

Software Qualität

Frühjahrssemester 2008

2. Juni 2008

Musterlösungen

- Für diese Prüfung stehen Ihnen 90 Minuten zur Verfügung.
- Verwenden Sie nur das ausgeteilte Papier für Ihre Lösung - Zusatzpapier erhalten Sie auf Anfrage.
- Schreiben Sie auf das Deckblatt Ihren Namen und Ihre Matrikelnummer.
- Lösen Sie alle Aufgaben alleine.
- Es sind ausschliesslich folgende Hilfsmittel erlaubt:
  - Ein doppelseitig (oder zwei einseitig) beschriebenes Blatt A4 mit selbst geschriebenen, handschriftlichen Notizen.
  - Für Studierende, deren Muttersprache nicht Deutsch ist: ein Wörterbuch.
- Ein Betrugsversuch hat ein Nichtbestehen dieses Tests zur Folge (d. h. 0 Punkte).
- Legen Sie Ihren Studienausweis („Legi“) auf ihren Arbeitsplatz.
- Heben Sie bei Fragen die Hand.

Bitte leer lassen!

Teil 1	Teil 2	Teil 3	Total
/20	/30	/40	/90

Teil 1: Wissensfragen (insgesamt 20 Punkte, ca. 20 Minuten Bearbeitungszeit)

Kreuzen Sie Zutreffendes an. Zu jeder Frage können mehrere Aussagen richtig sein.

Bitte beachten Sie, dass Ihnen für jedes falsch gesetzte Kreuz gleich viele Punkte abgezogen werden, wie Sie für eine korrekte Ankreuzung erhalten. Negative Punktzahlen ergeben null Punkte für die betreffende Frage.

Frage 1.1: Grundlagen (3 Punkte)

	Richtig	Falsch
Qualität ist ein absolutes Mass für Güte.	❑	■
Prozess- und Projektqualität dienen der Erreichung von Produktqualität.	■	❑
Bei der qualitativen Repräsentation von Qualität ist eine direkte formale Prüfung möglich.	❑	■
Qualiätsplanung, Qualitätslenkung und Qualitätsverbesserung sind Verfahren des Qualitätsmanagements.	■	❑
Zur statischen Prüfung gehören neben dem Testen und der Simulation auch das Model Checking.	❑	■
Als Qualitätssicherung bezeichnet man denjenigen Teil des Qualitätsmanagements, der auf das Festlegen der Qualitätsziele, der notwendigen Ausführungsprozesse und der benötigten Ressourcen zur Erfüllung der Qualitätsziele ausgerichtet ist.	❑	■

Frage 1.2: Produktqualität (3 Punkte)

	Richtig	Falsch
Der Qualitätsfaktor “Korrektheit” definiert, inwiefern ein System unberechtigte Veränderungen von Daten verhindert.	❑	■
Die kausalen Zusammenhänge zwischen Kenngrössen, Merkmalen und Faktoren sind in Qualitätsmodellen statisch abgesichert.	❑	■
Das Qualitätsmodell von McCall und Matsumoto weist einer Menge von Qualitätszielen einen Satz charakteristischer Merkmale zu.	■	❑
Eine Anforderung, die Restriktionen bezüglich Datenmengen oder Datenraten spezifiziert, gehört zu den funktionale Anforderungen.	❑	■
Ein inharäntes Merkmal ist eine kennzeichnende Eigenschaft einer Einheit aus sich selbst heraus.	■	❑
Quality Function Deployment (QFD) verfolgt das Ziel, systematisch Kundenanforderungen in Produktmerkmale umzusetzen.	■	❑

Frage 1.3: Prozessqualität (3 Punkte)

	Richtig	Falsch
Die Produktqualität bleibt das eigentliche Ziel bei der Software-Prozessverbesserung.	■	❑
Das Capability Maturity Model Integrated (CMMI) bietet einen Rahmen um eine Menge von Prozessbereichen auf einer Ordinalskala zu beurteilen.	■	❑
Eine zunehmende Prozessbürokratie und abnehmende Abhängigkeit von Individuen sind die grössten Gefahren des prozessorientierten Arbeitens.	❑	■
Die CMMI Reifestufe 2 (managed) sagt aus, dass die beurteilte Organisation einen quantitativ geführten Prozess institutionalisiert hat.	❑	■
Die Ziele des CMMI Prozessgebietes “Requirements Development” (RD) müssen nur von einer Firma mit optimierendem Reifegrad erreicht werden.	❑	■
SPICE und ISO/IEC 9126-1 sind neben CMMI alternative Modelle zur Prozessbeurteilung.	❑	■

Frage 1.4: Qualität bei evolutionärer Entwicklung (2 Punkte)

	Richtig	Falsch
Der Umfang einer Teillieferung beträgt bei der agilen Software-Entwicklung typisch 3 Wochen bis 6 Monate.	❑	■
Automatisierte, kontinuierliche Regresionstests sind ein integraler Bestandteil des agilen Qualitätsmanagements.	■	❑
Werkzeuge (Konfigurationsverwaltung, kontinuierliche Integration etc.) haben in der agilen Entwicklung keine grosse Bedeutung.	❑	■
Bei der evolutionären Entwicklung wird das System in eine Reihe von lauffähigen Iterationen unterteilt.	■	❑

Frage 1.5: Testen (3 Punkte)

	Richtig	Falsch
Getestet wird immer gegen die Spezifikation.	❑	■
Anwendungsfallorientiertes Testen gehört zur Familie der strukturorientierten Tests.	❑	■
Testautomatisierung verbessert die Testproduktivität.	■	❑
Die Güte datenflussorientierter Tests wird mit Hilfe verschiedener Überdeckungsgrade definiert.	■	❑
Der Abnahmetest hat zum Ziel, Fehler im gelieferten Sysem zu finden.	❑	■
Die Behebung von Fehlern ist ein integraler Bestandteil des Testens.	❑	■

Frage 1.6: Model-Checking (3 Punkte)

	Richtig	Falsch
Mit Aussagenlogik können keine zeitlichen Aussagen gemacht werden. Prädikatenlogik ermöglicht zeitliche Aussagen.	❑	■
Mit Hilfe des Model-Checking wird gezeigt, dass ein Modell eine geforderte Eigenschaft aufweist.	■	❑
Die Komplexität des effizienten Model Checking steigt linear mit der Anzahl der möglichen Zustände.	■	❑
Lebendigkeitseigenschaften machen Aussagen über die Unmöglichkeit verbotener/gefährlicher Eigenschaften.	❑	■
In der Temporalen Logik wird die Zeit als eine diskrete Folge von Zuständen modelliert.	■	❑
Beim symbolischen Model Checking wird das Model gezielt durch den Modellierer vereinfacht.	❑	■

Frage 1.7: Debugging (3 Punkte)

	Richtig	Falsch
Für eine statische Analyse zur Defektlokalisierung muss die Software ausgeführt werden.	❑	■
Eine Hypothese zur Defektlokalisierung darf nicht auf Annahmen basieren, die bereits als Teil einer falschen Hypothese verworfen wurden.	■	❑
Beim Delta Debugging wird die Menge der Eingabedaten in zwei gleich grosse Hälften und dann jeweils rekursive die Hälfte, die den Fehler enthält, geteilt.	❑	■
Nach der Problembehebung muss mit Hilfe eines Regressionstests auf unerwünschte Nebenwirkungen geprüft werden.	■	❑
Ein Defekt in einem Programm führt in der Regel sofort zu einem Fehler.	❑	■
In einem optimal vereinfachten Testfall sind alle Bestandteile relevant, d.h. die einzelnen Bestandteile können den Fehler unabhängig voneinander reproduzieren.	❑	■

Teil 2: Anwendungsaufgaben (insgesamt 30 Punkte, ca. 30 Minuten Bearbeitungszeit)

Aufgabe 2.1: Produktqualität (12 Punkte, ca. 12 Minuten Bearbeitungszeit)

a. Geben Sie eine kurze Definition (maximal zwei Sätze) der folgenden Qualitätsfaktoren:

- Zuverlässigkeit
- Flexibilität
- Wiederverwendbarkeit

(6 Punkte)

b. Geben Sie zwei Eigenschaften von Software an, die gemäss dem ISO/IEC 9126 Qualitätsmodell einen positiven Effekt auf deren Benutzbarkeit haben. Begründen Sie jeweils kurz (maximal zwei Sätze) den positiven Effekt.
(4 Punkte)

c. Welches Mass kann zur Beurteilung des Qualitätsmerkmals "Zuverlässigkeit" herangezogen werden? Erläutern Sie was genau dieses Qualitätsmass misst.
(2 Punkte)

Musterlösung

1. **Zuverlässigkeit**: Die Fähigkeit eines Systems, die verlangte Funktionalität unter gegebenen Randbedingungen für eine gegebene Zeit zu erfüllen.

   Flexibilität: Die Leichtigkeit, mit der ein System abgeändert werden kann, um es in Anwendungen oder Umgebungen zu nutzen, für die es nicht entwickelt worden ist.

   Wiederverwendbarkeit: Das Ausmass, in dem ein Stück Software in mehr als einem Programm oder Software-System verwendet werden kann.

2. **Usability:**
   Understandability
   Learnability
   Operability
   Attractiveness
   Compliance

3. MTTF (Mean Time To Failure): Durchschnittliche Zeit zwischen zwei Ausfällen

Aufgabe 2.2: Debugging (18 Punkte, ca. 18 Minuten Bearbeitungszeit)

Gegeben sei folgendes Codesegment:

```
01    //Determine constants for the equation.
02    float a = ellipse.getWidth() / 2f;
03    float b = ellipse.getHeight() / 2f;

04    float deltaY = convOut.getY() - convIn.getY();
05    float deltaX = convOut.getX() - convIn.getX();
06    if (deltaX == 0) deltaX = 0.1f;    // Prevent infinite slope
07    float c = deltaY / deltaX;         // line slope
08    float d = c * (0 - convIn.getX()) + convIn.getY();
09    float e = Math.sqrt(c * d);

10    //Solve the quadratic equation
11    float qa = 1f / (a * a) + (c * c) / (b * b);
12    float qb = ((2 * c * d) / (b * b));
13    float qc = ((d * d) / (b * b)) - 1f;
14    double discrim = qb * qb - 4 * qa * qc;

15    double x = 0; //X coordinate of intersection point
16    double y = 0; //Y coordinate of intersection point
17    double z = 0; //Z coordinate of intersection point

18    if ((c > 0 && deltaY > 0) ||
19        (c < 0 && deltaY < 0) ||
20        (c == 0 && deltaX > 0)) {
21        x = (-qb + Math.sqrt(discrim)) / (2 * qa);
22        y = c * x + d;
23        z = e * x + c;
24    } else if ((c < 0 && deltaY > 0) ||
25                (c > 0 && deltaY < 0) ||
26                (c == 0 && deltaX < 0)) {
27        x = (-qb - Math.sqrt(discrim)) / (2 * qa);
28        y = c * x + d;
29        z = e * x + c;
30    }

31    Point intersect = new Point((int)x, (int)y);
```

a. Geben Sie all Use-Set und Set-Set Beziehungen im obenstehenden Code an.
(4 Punkte)

b. Um den vorliegenden Code detaillierter zu analysieren wollen die Wartungsprogrammierer wissen, wie das Ergebnis in Zeile 31 zu Stande kommt. Geben Sie dazu den statischen Backward Slice S(31, {intersect}) an. Tragen Sie Ihr Ergebnis direkt im gegebenen Codefragment ein.
(8 Punkte)

c. Beschreiben Sie den Zusammenhang zwischen einem Defekt und einem (beobachtbaren) Fehler.
(2 Punkte)

d. Geben Sie vier Techniken zur Defektlokalisierung an und beschreiben Sie diese ganz kurz (jeweils maximal ein Satz).
(4 Punkte)

Musterlösung

1. **Use-Set** (1P): Linie 6: deltaX

   Set-Set (total 3P, jeweils 0.5P): 21-15, 22-16, 23-17, 27-15, 28-16, 29-17

2. **Backward Slice**

   zu streichende Zeilen (je 2P pro richtig gestrichenen Linie): 9, 17, 23, 29

3. **Defekt**: schadhafte Stelle im Programm
   Fehler: Ergebnisse weichen von erwartetem Wert ab

   Nicht jeder Defekt führt zu einem Fehler. Ein Fehler kann durch die Kombination mehrerer Defekte verursacht werden.

4. Hypothesen aufstellen und testen
   Programmzusände analysieren
   Programmablauf beobachten
   Zusicherungen dynamisch überprüfen
   Ursache-Wirkungsketten erkennen
   Debugging nach Gefühl

Teil 3: Essay (insgesamt 40 Punkte, ca. 40 Minuten Bearbeitungszeit)

Aufgabe 3.1: Qualität im agilen Umfeld (40 Punkte, ca. 40 Minuten Bearbeitungszeit)

Vergleichen Sie detailliert die unterschiedlichen Ansätze des Qualitätsmanagements bei einer agilen und einer konventionellen Entwicklung. Machen Sie sich auch Gedanken zu den unterschiedlichen Voraussetzungen und zum Umfeld, in dem die Entwicklung abläuft.