
Name Vorname Matrikelnummer

Universität Zürich

Software Engineering

Herbstsemester 2008

Bachelor

13. Januar 2009

Musterlösungen

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

Bitte leer lassen!

Teil 1	Teil 2	Teil 3	Total
/40	/40	/40	/120

Teil 1: Wissensfragen (insgesamt 40 Punkte, ca. 30 Minuten Bearbeitungszeit)

Wir verwenden neu drei Typen von Multiple-Choice-Fragen:

Typ 1 Zu jeder gestellten Frage ist genau eine Antwort anzukreuzen.

Typ N Zu jeder gestellten Frage sind *n* Antworten anzukreuzen. Die Anzahl ist in der jeweiligen Frage vermerkt.

Typ ALL Für jede Aussage muss die zutreffende Antwort angekreutzt werden.

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 [Typ 1] 1.1: Das Qualitätsmodell der ISO/IEC 9126 (1 Punkt)

Welche der folgenden Eigenschaften aus dem Qualitätsmodell (nach ISO/IEC 9126) beschreibt die Fähigkeit einer Software, ihr Leistungsniveau unter festgelegten Bedingungen über einen festgelegten Zeitraum zu bewahren?

- ❑ Funktionalität
- ❑ Sicherheit
- ■ Zuverlässigkeit
- ❑ Änderbarkeit
- ❑ Effizienz

Frage [Typ 1] 1.2: Merkmale einer guten Anforderungsspezifikation (1.5 Punkte)

Gegeben sind fünf Merkmale einer guten Anforderungspezifikation:

(A) Widerspruchsfreiheit

(B) Verständlichkeit

(C) Eindeutigkeit

(D) Prüfbarkeit

(E) Adäquatheit

Welches der genannten Merkmale...

1. ...stellt sicher, dass die Spezifikation realisierbar ist?
   ■ (A) ❑ (B) ❑ (C) ❑ (D) ❑ (E)
2. ...vermeidet Fehler durch Fehlinterpretationen?
   ❑ (A) ❑ (B) ■ (C) ❑ (D) ❑ (E)
3. ...ermöglicht es festzustellen, ob das realisierte System die Anforderungen erfüllt?
   ❑ (A) ❑ (B) ❑ (C) ■ (D) ❑ (E)

Frage [Typ 1] 1.3: Techniken der Informationsbeschaffung (1.5 Punkte)

Gegeben sind fünf Techniken zur Informationsbeschaffung:

(A) Umfragen/Fragebogen

(B) Beobachtung

(C) Gemeinsame Arbeitstagungen

(D) Prototyp

(E) Beteiligtenanalyse

Welche Technik ist die adäquateste, um...

1. ...eine Liste von Anforderungen zu priorisieren?
   ❑ (A) ❑ (B) ■ (C) ❑ (D) ❑ (E)
2. ...die momentane Arbeit und die Probleme eines Stakeholders zu verstehen?
   ❑ (A) ■ (B) ❑ (C) ❑ (D) ❑ (E)
3. ...die Machbarkeit eines zukünftigen Systems zu studieren?
   ❑ (A) ❑ (B) ❑ (C) ■ (D) ❑ (E)

Frage [Typ ALL] 1.4: Spezifikationsmethoden (2 Punkte)

Welche der folgenden Spezifikationsdarstellungen sind konstruktive bzw. deskriptive Darstellungen?

	konstr.	**deskr.**
Spezifikation mit Anwendungsfällen (use cases)	■	❑
Spezifikation mit Objektmodellen	■	❑
Algebraische Spezifikation	❑	■
Verhaltensspezifikation mit Zustandsautomaten	■	❑

Frage [Typ ALL] 1.5: Entwurfsprinzipien (2 Punkte)

Welche der folgenden Kriterien sind gute bzw. schlechte Charakteristiken eines Designs?

	Gut	**Schlecht**
Eine starke Kopplung zwischen Modulen	❑	■
Die verfügbaren Ressourcen sind nicht beachtet worden	❑	■
Die Struktur der Architektur wird entsprechend der Struktur des Problems gewählt	■	❑
Das Design ist änderungsfreundlich	■	❑

Frage [Typ 1] 1.6: Modularität (1 Punkt)

Kohäsion ist eine qualitative Bezeichnung für den Grad, wie sehr ein Modul...

❑ ...kompakter geschrieben werden kann.

■ ...sich auf eine Sache fokusiert.

❑ ...fähig ist, seine Aufgabe rechtzeitig zu vollenden.

❑ ...mit anderen Modulen verbunden ist.

Frage [Typ 1] 1.7: Entwurfsmuster (1.5 Punkte)

Gegeben sind die folgenden fünf Entwurfsmuster (design patterns):

(A) Strategie (Strategy)

(B) Beobachter (Observer)

(C) Proxy

(D) Adapter

(E) Abstrakte Fabrik (Abstract Factory)

Welches dieses Entwurfsmuster ermöglicht es,...

1. ...den Zugriff auf ein anderes Objekt zu kontrollieren?
   ❑ (A) ❑ (B) ■ (C) ❑ (D) ❑ (E)
2. ...verschiedene Algorithmen zu definieren und diese austauschbar zu machen?
   ■ (A) ❑ (B) ❑ (C) ❑ (D) ❑ (E)
3. ...ein System mit einer Familie verwandter Produkte zu konfigurieren?
   ❑ (A) ❑ (B) ❑ (C) ❑ (D) ■ (E)

Frage [Typ 1] 1.8: Zusammenarbeit und Architekturstile (1 Punkt)

Welche Kombination von einem Architekturstil mit einem Kollaborationsstil ist offensichtlich **falsch**?

❑ Ein ereignisbasiertes System mit Informationsaustausch nach dem Abonnementsprinzip

❑ Eine Schichtenarchitektur (layered architecture) mit Leistungserbringungen

❑ Eine Prozessorientierte Architektur mit Informationsteilhabe durch gemeinsame Speicherbereiche

❑ Eine Blackboard-Architektur mit einer gemeinsam genutzten Datenabstraktion

■ Eine Pipe-and-Filter Architektur mit Informationsaustausch nach dem Holprinzip

Frage [Typ 1] 1.9: Systematisches Programmieren (1 Punkt)

Eine der folgenden Praktiken macht den Programmcode **nicht** besser verständlich. Welche?

■ Namen von Variablen werden kurz gehalten, um den Code kompakt zu halten

❑ Ein visuelles Layout wird erstellt, welches die Struktur sichtbar macht

❑ Die Geltungsbereiche von Variablen werden so klein wie möglich gehalten

❑ Ein Programm wird in kleinere Subprogramme aufgeteilt

❑ Die Semantik der Interfaces wird dokumentiert

Frage [Typ N] 1.10: Fehlerterminologie (1 Punkt)

Welche **zwei** der folgenden Terme werden oft als “Bug” bezeichnet?

■ Defekt

❑ Irrtum

■ Fehler

❑ Befund

❑ Ausfall

Frage [Typ ALL] 1.11: Prüfverfahren (2 Punkte)

Welche der folgenden Methoden sind dynamische bzw. statische Prüfverfahren?

	dyn.	stat.
Review	❑	■
Model Checking	❑	■
Testen	■	❑
Simulieren	■	❑

Frage [Typ 1] 1.12: Testarten (1 Punkt)

Welche Testart ist **nicht** speziell dazu da, um Fehler zu finden?

- ❑ Der Integrationstest
- ❑ Der Systemtest
- ❑ Der Komponententest
- ■ Der Abnahmetest
- ❑ Der Belastungstest

Frage [Typ 1] 1.13: Testverfahren (1 Punkt)

Welche Arten von Fehlern können beim Black-Box-Test übersehen werden und beim White-Box-Test erkannt werden?

- ❑ Fehlende oder falsche Funktionalität
- ■ Logische Fehler
- ❑ Performanzfehler
- ❑ Tippfehler
- ❑ Initialisierungs- und Terminierungsfehler

Frage [Typ ALL] 1.14: Testen von Teilsystemen (2 Punkte)

Welche der folgenden Eigenschaften sind jeweils der Abwärts- bzw. Aufwarts-integration zuzuordnen?

	Ab.	Auf.
Elementare Funktionen werden früh getestet	❑	■
Wichtige Anwendungslogik wird früh getestet	■	❑
Man muss keine Test-Treiber (drivers) schreiben	■	❑
Man muss keine Test-Stümpfe (stubs) schreiben	❑	■

Frage [Typ 1] 1.15: Review-Formen (1 Punkt)

Welches der vier genanten Ziele ist das wichtigste Ziel einer Inspektion?

- ❑ Fehlerbehebung
- ❑ Beurteilung der Stärken eines Programms
- ❑ Beurteilung der Programmiererqualität
- ■ Entdeckung von Defekten

Frage [Typ 1] 1.16: Skalentypen (1.5 Punkte)

Bei der Messung von Merkmalen unterscheiden wir fünf Skalentypen:

(A) Nominalskala

(B) Ordinalskala

(C) Intervallskala

(D) Verhältnisskala

(E) Absolutskala

Welchen Typ haben die Skalen der folgende Masse?

1. Anzahl Codezeilen zur Messung der Programmgroße?
   ❑ (A) ❑ (B) ❑ (C) ■ (D) ❑ (E)
2. Celsiusskala zur Messung der Temperatur?
   ❑ (A) ❑ (B) ■ (C) ❑ (D) ❑ (E)
3. Zählskala für die Anzahl der Fehler in einem Modul?
   ❑ (A) ❑ (B) ❑ (C) ❑ (D) ■ (E)

Frage [Typ 1] 1.17: Komplexitätsmasse (1 Punkt)

Die zyklomatische Komplexität (McCabe) liefert dem Programmierer Informationen über...

❑ ...die Anzahl an Schleifendurchläufen im Programm.

❑ ...die Anzahl an Fehlern im Programm.

■ ...die Anzahl unabhängiger logische Pfade im Programm.

❑ ...die Anzahl von Anweisungen im Programm.

Frage [Typ ALL] 1.18: Messen von Software (2 Punkte)

Welches dieser Masse ist ein Produkt- bzw. ein Prozessmass?

	Produkt	**Prozess**
Die Fehlerrate	❑	■
Mean Time To Failure	■	❑
Die Qualitätskosten	❑	■
Die Fehlerdichte	■	❑

Frage [Typ 1] 1.19: Program Slicing (1 Punkt)

Zu welcher dieser Aktivitäten ist Slicing **unnütz**?

❑ Fehlersuche

❑ Restrukturierung

■ Messung von Kopplung

❑ Änderungsanalyse

❑ Programmoptimierung

Frage [Typ 1] 1.20: Wartungsarten (1 Punkt)

Welches Wort fehlt um den folgenden Satz korrekt zu vervollständigen?

Das Verbessern der Performanz und Wartbarkeit der Software ist eine _____ Softwarewartung.

- [x] perfektionierende
- [ ] adaptive
- [ ] korrigierende
- [ ] präventive

Frage [Typ 1] 1.21: Voraussage der Wartung (1 Punkt)

Welcher der folgenden Faktoren beeinflusst die Beziehung zwischen einem System und seiner Umgebung **nicht**?

- [ ] Die Anzahl der Systemschnittstellen
- [ ] Die Komplexität der Systemschnittstellen
- [ ] Die Anzahl der Geschäftsprozesse, welche das System benutzen
- [ ] Die Anzahl der sich ändernden Anforderungen
- [x] Die Anzahl der Komponenten innerhalb des Systems

Frage [Typ 1] 1.22: Reengineering-Prozess (1 Punkt)

Welche der folgenden Aktivitäten bildet den Anfang des Reengineering-Prozesses?

- [ ] Forward Engineering
- [ ] Refactoring
- [ ] Prototyping
- [x] Reverse Engineering
- [ ] Restructuring

Frage [Typ 1] 1.23: Ergebnisorientierte Phasenmodelle (1 Punkt)

Ein ergebnisorientiertes Phasenmodell für die Softwareentwicklung ist...

- [x] ...ein angemessenes Modell wenn die Anforderungen genau definiert sind.
- [ ] ...ein geeignetes Modell wenn schon früh ein lauffähiger Kern des Programms gebraucht wird.
- [ ] ...das beste Modell für Projekte mit grossen Entwicklerteams.
- [ ] ...ein geeignetes Modell für Projekte mit grossen Risiken.

Frage [Typ 1] 1.24: Wachstumsmodelle (1 Punkt)

Welche der folgenden Aussagen über evolutionäre Softwareprozesse ist **falsch**?

- ❑ Sie haben einen iterativen Charakter.
- ❑ Sie können wechselnde Produktanforderungen gut aufnehmen.
- ❑ Sie produzieren früh lauffähige Systeme.
- ■ Sie orientieren sich am natürlichen Softwarelebenslauf.
- ❑ Sie reduzieren die Fehlerkosten.

Frage [Typ 1] 1.25: Agile Softwareentwicklung (1 Punkt)

Welcher der folgenden Sätze passt **nicht** zu den Prinzipien der agilen Softwareentwicklung?

- ❑ Individuen und Interaktionen sind wichtiger als Prozesse und Tools
- ❑ Eine lauffähige Software ist wichtiger als eine umfangreiche Dokumentation
- ■ Vertragsverhandlungen sind wichtiger als eine Zusammenarbeit mit dem Kunden
- ❑ Auf Änderungen zu reagieren ist wichtiger als einem Plan zu folgen

Frage [Typ 1] 1.26: Projektplanung (1 Punkt)

Der erste Schritt bei der Projektplanung ist...

- ❑ ...das Budget zu bestimmen.
- ❑ ...die Organisationsstruktur des Teams zu planen.
- ❑ ...die Projekteinschränkungen zu bestimmen.
- ■ ...die Ziele und den Rahmen des Projektes zu ermitteln.

Frage [Typ 1] 1.27: Projektplan (1 Punkt)

Der Projektplan gibt Antwort auf sechs W-Fragen. Welche Frage ist für den Projektplan am **wenigsten** wichtig?

- ❑ Warum wird das System entwickelt?
- ❑ Wer ist für was verantwortlich?
- ■ Welcher Architektur-Stil wird gebraucht?
- ❑ Welche Mittel werden für das zu entwickelnde System benötigt?
- ❑ Wie wird der Projekterfolg sichergestellt?

Frage [Typ 1] 1.28: Meilensteine (1 Punkt)

Für welche Aufgabe können Meilensteine **nicht** hilfreich eingesetzt werden?

- ❑ Einhaltung von Terminen
- ❑ Erreichung von Sachzielen
- ❑ Erreichung von Kostenzielen
- ■ Reduzierung von Personalrisiken

Frage [Typ 1] 1.29: Schätzverfahren (1 Punkt)

Algorithmische Schätzverfahren basieren typischerweise auf...

- ❑ ...Beurteilungen durch Experten anhand von bisherigen Projekterfahrungen
- ❑ ...der Verfeinerung von geschätzten, erwarteten Werten
- ■ ...statistischen Modellen, abgeleitet von Daten vergangener Projekte
- ❑ ...dem Ermitteln von Parametern und Koeffizienten durch Versuch und Irrtum (trial and error)

Frage [Typ 1] 1.30: COCOMO (1 Punkt)

Eine der folgenden Attributarten ist **nicht** relevant für den geschätzten Aufwand anhand des COCOMO Modells. Welche?

- ■ Prozessattribute
- ❑ Personalattribute
- ❑ Computerattribute
- ❑ Produktattribute
- ❑ Projektattribute

Frage [Typ 1] 1.31: Stufen des CMM (1 Punkt)

Welcher der folgenden Begriffe bezeichnet **keine** Stufe des CMM (Capability Maturity Model)?

- ❑ Ad hoc (Initial)
- ❑ Wiederholbar (Repeatable)
- ❑ Definiert (Defined)
- ■ Organisiert (Organized)
- ❑ Optimierend (Optimizing)

Frage [Typ 1] 1.32: Software-Prozessverbesserung (1 Punkt)

In welcher Phase des zyklischen Vorgehens (Plan-Do-Check-Act) zur Software-Prozessverbesserung werden **Schwächen eliminiert**?

- ❑ Beim Planen (Plan)
- ❑ Beim Ausführen (Do)
- ❑ Beim Überprüfen (Check)
- ■ Beim Handeln (Act)

Frage [Typ 1] 1.33: Definitionen betreffend Software-Konfigurationsverwaltung (1 Punkt)

Welche Wörter fehlen um den folgenden Satz korrekt zu vervollständigen?

Ein(e) neue(r/s) (1)_____ wird definiert, wenn Änderungen an einer (2)_____ vorgenommen wurden.

- ❑ (1) Release / (2) Konfiguration
- ❑ (1) Variante / (2) Software-Einheit
- ❑ (1) Variante / (2) Version
- ■ (1) Version / (2) Software-Einheit
- ❑ (1) Konflikt / (2) Konfiguration

Teil 2: Anwendungsaufgaben - Skedoole (insgesamt 40 Punkte, ca. 30 Minuten Bearbeitungszeit)

Während des Weihnachtsessens beklagte sich Ihre Tante über eine immer wiederkehrende und mühsame Aufgabe, welche sie im Rahmen ihres Jobs erledigen muss: Sitzungen für ihren Boss planen. Gemäss ihrer Aussage ist die nervigste und zeitraubendste Aufgabe von typischen Geschäftsleuten das Planen von Sitzungszeiten. Egal ob Face-to-Face oder per Telefon, einen passenden Termin für ein Treffen zu finden ist mühsam, auch wenn nur zwei Leute daran beteiligt sind. Wenn mehrere Leute einen gemeinsamen Termin finden müssen, vergehen Tage, und Duzende von Emails werden gesendet bis ein passender Zeitpunkt gefunden wird.

Während der Heimreise entschliessen Sie sich, ein Softwareprodukt zu entwickeln, welches das Leben Ihrer Tante — und anderer Geschäftsleute — einfacher macht.

Hinweis: Ähnlichkeiten mit bestehenden Produkten sind zufällig ;-).

Während einer Vorlesung über Verteilte Algorithmen haben Sie von verschiedenen Protokollen zum Lösen des Konsens-Problems gehört. Aufgeregt schreiben Sie ein visionäres Dokument über *Skedoole*, Ihr zukünftiges Produkt.

> „Skedoole wird eine Web-Plattform, welche eine Menge von Usern beim Finden eines gemeinsamen Termins unterstützt. Um dieses Ziel zu erreichen, werden die Interaktionen zwischen den Usern in drei Phasen aufgeteilt.
>
> In der Vorschlags-Phase benutzt der Organisator seinen Browser, um auf die Plattform zuzugreifen und eine Abstimmung zu erstellen. Eine Abstimmung besteht grob aus einer Menge von alternativen Terminen, für welche die Teilnehmer später stimmen können. Um die Abstimmung zu beschreiben, kann der Organisator einen Titel eintippen und einen Kommentar eingeben. Skedoole speichert daraufhin diese Einstellungen in einer internen Datenbank ab und liefert dem Organisator einen Schlüssel zurück. Jeder generierte Schlüssel identifiziert genau eine Abstimmung.
>
> In der zweiten Phase (Abstimmungs-Phase) sendet der Organisator den Schlüssel an alle Teilnehmer, welche zum zukünftigen Treffen eingeladen sind. Zurzeit gehört das Senden des Schlüssels nicht zu den Verantwortlichkeiten der Plattform. Der Schlüssel erlaubt einem User den Zugang zur Abstimmung. Wenn ein User verbunden ist, kann er diejenigen Termine auswählen, welche für ihn passend sind. Um die Chance zu erhöhen, dass ein gemeinsamer Termin gefunden wird, kann der User alle bereits abgegebenen Stimmen zur Abstimmung sehen.
>
> In der Abschluss-Phase entscheidet der Organisator über den endgültigen Termin für das Treffen anhand der durch die Teilnehmer abgegebenen Stimmen. Dann informiert er die Teilnehmer über seine Entscheidung. Das System bietet keinerlei Unterstützung in dieser Phase. Es findet keine Entscheidungshilfe durch das System statt und die Benachrichtigung muss mit einer anderen Technik (zum Beispiel via Email) gesendet werden — analog zur Versendung des Schlüssels.
>
> Die Plattform muss so intuitiv und simpel wie möglich sein. Daher wird keine Authentifizierung der Benutzer benötigt. Allerdings muss das System den Zugang zu einer Abstimmung für all diejenigen Benutzer verweigern, welche den Schlüssel nicht kennen!
>
> Um den Wartungsaufwand so niedrig wie möglich zu halten, sehen wir keine Notwendigkeit für einen Administrator. Eine Abstimmung wird automatisch aus der Datenbank gelöscht, wenn einen Monat lang kein Zugriff auf die Abstimmung stattfindet.“

Einer Ihrer Kollegen hat Ihr visionäres Dokument erweitert:

> „Wir erwarten, dass sich die Anzahl an Benutzern exponentiell erhöhen wird: jedes Mal, wenn ein Treffen mit Hilfe von Skedoole erfolgreich geplant wurde, werden die Teilnehmer die Plattform möglicherweise als Organisator in einem anderen sozialen Netzwerk erneut benutzen!
>
> Falls das System erfolgreich ist, werden wir weitere Funktionen implementieren. Dazu könnte zum Beispiel eine Funktion gehören, die das Exportieren der Daten in andere Formate ermöglicht, um sie zu archivieren. Eine weitere Möglichkeit wäre, Skedoole mit populären Agenda-Tools zu verbinden. Oder man könnte die Domäne von Skedoole erweitern, von einem Terminplanungs-Tool zu einem Tool für allgemeine Umfragen.“

Frage [Typ ALL] 2.1: Klassifikation von Anforderungen (2 Punkte)

„Allerdings muss das System den Zugang zu einer Abstimmung für all diejenigen Benutzer verweigern, welche den Schlüssel nicht kennen."

Diese Anforderung...

	richtig	**falsch**
...ist eine Tatsache.	❑	■
...kann graduell erfüllt sein.	❑	■
...ist operational	■	❑
...beschreibt ein Systemverhalten.	■	❑

Aufgabe 2.2: Anforderungsdokumentation (14 Punkte, ca. 10,5 Minuten Bearbeitungszeit)

Ihre Aufgabe ist es, den obigen Text zu analysieren und mit einigen Techniken zu dokumentieren, die Sie aus der Vorlesung kennen. Ignorieren Sie für diese Aufgabe die Ergänzungen Ihres Kollegen.

a. Erstellen Sie ein Glossar mit mindestens **drei** relevanten Begriffen. **(6 Punkte)**

Musterlösung

Abschluss-Phase eine Phase, in welcher die Endgültige Entscheidung getroffen wird

Abstimmung eine Menge von Alternativen (d.h. die Abstimmung als Objekt), oder ein Prozess, in dessen Verlauf eine Entscheidung getroffen wird (d.h. der Prozess als Objekt)

Abstimmung-Phase eine Phase, in welcher die Teilnehmer ihre Verfügbarkeiten bekannt geben

Organisator eine Person, die ein Meeting organisieren will (und nicht eine Person, welche eine Abstimmung einrichtet)

Phase jede einzelne Zeitspanne in einer Sequenz von Ereignissen

Schlüssel ein Objekt (eine URL, eine Kette von Zeichen), welches eine Abstimmung identifiziert oder Zugang zu ihr gewährt

Teilnehmer jemand, der an einem Meeting teilnehmen soll (und nicht eine Person, welche eine Stimme abgibt)

Termin eine formell arrangierte Versammlung (aber nicht zufällig oder unerwartet)

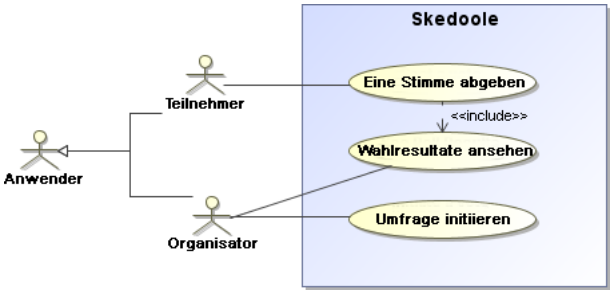
Umfrage der Vorgang, in dem man viele Menschen befragt, um ihre Meinungen zu bestimmten Sachverhalten heraus zu finden

Vorschlags-Phase eine Phase, in welcher ein Organisator eine Menge von möglichen Meetings vorschlägt

Web-Plattform a) a la web 2.0: eine Web-Applikation, welche Benutzer verbindet, oder b) (wurde für die Plattform akzeptiert) eine technische Plattform, auf der eine Applikation ausgeführt wird (J2EE, .Net)

b. Zeichnen Sie ein Anwendungsfalldiagramm, welches die Haupt-Anwendungsfälle von Skedoole repräsentiert. Beschreiben Sie kurz den normalen Ablauf jedes identifizierten Anwendungsfalls. **(8 Punkte)**

Musterlösung



Abstimmung initiieren 1. Der Organisator gibt einige Informationen zur Umfrage ein (Titel, Kommentar) und bestimmt die Auswahlmöglichkeiten.

2. Das System speichert die Umfrage in seiner internen Datenbank und liefert dem Organisator einen eindeutigen Schlüssel zurück

Wahlresultate ansehen 1. Der Benutzer gibt den Schlüssel für die Umfrage ein

2. Das System holt die Umfrage aus der Datenbank und zeigt sie dem Benutzer

Eine Stimme abgeben 1. (beinhaltet das Szenario *Wahlresultate ansehen*)

2. Der Teilnehmer wählt die für ihn passenden Wahlmöglichkeiten aus

Aufgabe 2.3: Risikoanalyse (6 Punkte, ca. 4,5 Minuten Bearbeitungszeit)

Ihr Bankberater hat sich geweigert, Ihnen einen Kredit zu geben, weil Sie nicht die Risiken analysiert haben, welche Ihr Software-Projekt bedrohen. Identifizieren Sie **zwei** dieser Risiken (mindestens eines davon muss spezifisch auf Skedoole zutreffen). Schätzen und begründen Sie kurz die Gefährlichkeit jedes Risikos. Schlagen sie zu jedem Risiko eine Massnahme vor, wodurch Sie das Risiko beherrschen können.

Musterlösung

Anforderungen Unzureichende Plattform für Terminplanung, Performance Probleme

Personal Mangel an qualifiziertem Personal, eine Schlüsselperson ist nicht verfügbar

Technologie Bezogen auf Middleware, Netzwerk Infrastruktur

Tools benutzt für die Entwicklung von Software

Organisatorisch Mangel an Ressourcen, das Projekt muss gestoppt werden
z.B. Eintretenswahrscheinlichkeit (p) = niedrig, Schadenhöhe (s) = hoch, Risikofaktor (f) = p * s = mittel
Massnahme: differenzierte Ressourcenplanung

Schätzung Entwicklungskosten unterschätzt, Verwendung überschätzt

Aufgabe 2.4: Software-Prozessmodelle (6 Punkte, ca. 4,5 Minuten Bearbeitungszeit)

a. Welches Prozessmodell wählen Sie für Ihr Projekt? **(2 Punkte)**

Musterlösung

Wachstumsmodell

b. In welchem Bezug ist ihr gewähltes Prozessmodell besonders vorteilhaft? **(2 Punkte)**

Musterlösung

Für Wachstumsmodell oder Agile Softwareentwicklung:

- Kann umgehen mit einem System welches wächst (Prototyp ist Pilotsystem)
- Kürzere Feedback-Schlaufe mit den Benutzern
- Frühe Lieferung eines funktionierenden Systems

Für Phasenmodell:

- Planung
- Einfach zu verstehen, folgt dem Lebenszyklus von Software
- Frühes finden von Fehlern

c. Welche Nachteile oder Voraussetzungen haben Sie davon abgehalten, ein anderes Prozessmodell zu wählen? **(2 Punkte)**

Musterlösung

Agile SW-Entwicklung Benötigt einen Kunden, welcher nicht existiert (die Tante hat vielleicht keine Zeit); benötigt erfahrene Entwickler

Phasenmodell Benötigt präzise formulierte Anforderungen; das System wird spät und auf einen Schlag geliefert; Risiko der nicht-Adäquatheit; benötigt know-how

Wasserfallmodell Veraltetes Modell

Wachstumsmodell Risiko einer schlecht strukturierten Software; Unmöglichkeit, das System in partielle Lieferungen aufzuteilen (was hier nicht der Fall ist, da wir bereits wissen, welches die Features für den nächsten Ausbauschritt sind)

Aufgabe 2.5: Software-Architekturstile (12 Punkte, ca. 9 Minuten Bearbeitungszeit)

Ihr visionäres Dokument verlangt offensichtlich die Anwendung des Client/Server-Architekturstils. Klienten sind dabei die Browser der Benutzer, während der Server als Web-Server den Terminplanungs-Service anbietet.

Finden Sie für die folgenden Beschreibungen jeweils einen passenden Architekturstil und geben Sie diesen an. Begründen Sie Ihre Wahl kurz.

a. Die initiale Funktionalität der Plattform: Benutzer können ihr Terminfindungsproblem lösen, indem sie nacheinander ihre Stimmen abgeben. **(4 Punkte)**

Musterlösung

Blackboard. Jede Umfrage ist ein Blackboard (gemeinsame Daten: die Alternativen und die Stimmen der Teilnehmer). Wissensquellen sind die Benutzer (oder das User Interface). Benutzer sind miteinander verbunden durch das Blackboard (jeder kann die vorläufigen Resultate der Abstimmung sehen, wie im Chat Beispiel). Die Möglichkeit zur Weiterentwicklung ist der Hauptvorteil hier, man kann die Domain von Skedoole erweitern, oder neue Arten hinzufügen, wie auf die Umfragen zugegriffen werden kann.

b. Die zukünftige Export-Funktion, die es den Usern erlaubt, die Resultate einer Abstimmung in verschiedene Formate zu exportieren, und anhand verschiedener Regeln zu sortieren und zu filtern. **(4 Punkte)**

Musterlösung

Pipe and filter. Konverter, Filter und Sortierungen sind Filter (Komponente). Filter sind miteinander verbunden durch Pipes (Konnektoren). Daten fliessen von der Quelle (die Daten der Umfrage) zur Senke (die Browser der Benutzer). Flexibilität ist der grosse Vorteil hier, man kann neue Filter hinzufügen oder die Filter neu anordnen.

c. Die zukünftige Funktion, die automatisch nach der Abschluss-Phase den endgültigen Termin in einem Agenda-Tool des Users hinzufügen kann. Sie können davon ausgehen, dass es ein Standardformat gibt, um Termine zwischen verschiedenen Tools auszutauschen. **(4 Punkte)**

Musterlösung

Publish-Subscribe. Agenda Tools sind die Abonnenten (subscriber), Skedoole ist der Publisher. Der Publisher publiziert beschlossene Meetings bei allen registrierten Agenda Tools. Der Konnektor ist die Infrastruktur für die Registrierung und die Übertragung von Ereignissen. Der Vorteil ist die Interoperabilität: jedes Tool, welches das iCal Format unterstützt, kann an Skedoole angeschlossen werden.

Teil 3: Anwendungsaufgaben - Implementierung (insgesamt 40 Punkte, ca. 30 Minuten Bearbeitungszeit)

Gegeben ist die folgende Implementierung einer Datenstruktur.

```
1  /**
2   * A collection designed for holding integers.
3   * This collection has a limited capacity.
4   * Elements are ordered in a LIFO
5   * (last-in-first-out) manner.
6   */
7  public class IntCo
8  {
9    private int h = 0;
10   private int cap;
11   private int[] elts;
12
13   /**
14    * Erzeugt einen leeren Stapel.
15    * @param c die maximale Stapelhoehe
16    * @return einen neuen leeren Stapel
17    */
18   public IntCo(int c)
19   {
20     cap = c;
21     elts = new int[cap];
22   }
23
24   /**
25    * @return true if the stack is full
26    * @pre true
27    * @post result == (h == cap)
28    */
29   public boolean isFull() {return h == cap;}
30
31   /**
32    * @return true if the stack is empty
33    * @pre true
34    * @post result == (h == 0)
35    */
36   public boolean isEmpty() {return h == 0;}
37
38   /**
39    * Pop an element out of the stack.
40    */
41   //Decrement the height prior
42   //to return the element it indices
43   public int remove() {return elts[--h];}
44
45   /**
46    * Push the specified element into the stack
47    */
48   //First store the element,
49   //then increase the height
50   public void add(int e) {elts[h++] = e;}
51 }
```

Frage [Typ 1] 3.1: Programmverstehen (2 Punkte)

Welche Datenstruktur wird hier implementiert?

- ❑ Warteschlange (Queue)
- ❑ Binärbaum (Binary Tree)
- ❑ Halde (Heap)
- ■ Stapelspeicher (Stack)
- ❑ Hashtabelle (Hashtable)

Aufgabe 3.2: Systematisches Programmieren (8 Punkte, ca. 6 Minuten Bearbeitungszeit)

Beurteilen Sie diesen Code hinsichtlich der systematischen Programmierung.

a. Formulieren Sie **fünf** Kritikpunkte an der Dokumentation dieses Quelltextes. **(5 Punkte)**

Musterlösung

Klasse Es fehlen die Tags Autor, Responsibility, Version, History

Attribute Kommentare fehlen

Methoden Inkonsistente Kommentare: Verträge fehlen (pre/post), Beschreibung von Parametern und Rückgabewerten

Innerhalb von Methoden Redundant mit dem Code

Sprache Mix aus Englisch und Deutsch

b. Die Namensgebung einiger Attribute, Methoden und Parameter ist offensichtlich ungenügend. Finden Sie bessere Namen für **drei** von diesen schlecht benannten Elementen. **(3 Punkte)**

Musterlösung

- $h \rightarrow size$
- $cap \rightarrow capacity$
- $elts \rightarrow elements$ (aber nicht stack)
- $c \rightarrow capacity$
- $remove \rightarrow pop$
- $add \rightarrow push$
- $IntCo \rightarrow Stack$ (oder IntCollection oder IntegerCollection)

Aufgabe 3.3: Verträge (10 Punkte, ca. 7,5 Minuten Bearbeitungszeit)

Für diese Verträge können Sie die `h`, `elts` and `cap` Attribute benutzen. Es sind Statusvariablen, aber sie sind sinnvoll für mögliche Benutzer der Klasse. Benutzen Sie die Java-Notation (mit @pre) für die Formulierung der Verträge.

a. Formulieren Sie die Voraussetzungen für die `remove()` und `add()` Methoden als logische Ausdrücke. **(3 Punkte)**

Musterlösung

- `!isEmpty()` für `remove()`.
- `!isFull()` für `add()`.

b. Formulieren Sie die Ergebniszusicherungen für die `remove()` und `add()` Methoden als logische Ausdrücke. Verträge für Variablen, die sich nicht ändern, müssen nicht spezifiert werden. **(5 Punkte)**

Musterlösung

- `result == elements[height] && height == (height@pre - 1)` für `remove()`.
- `elements[height@pre]==element && height == (height@pre + 1)` für `add()`.

c. Was passiert konkret, wenn ein Benutzer dieser Methoden die Voraussetzungen nicht erfüllt? **(2 Punkte)**

Musterlösung

Die Methode wirft eine OOB Exception (index out of bounds), und somit kann die Ergebniszusicherung nicht sichergestellt werden.

Aufgabe 3.4: Testen (20 Punkte, ca. 15 Minuten Bearbeitungszeit)

Die Klasse `IntCo` wurde um die Methode `areAllIntegersEven()` erweitert:

```
1  /**
2   * This query method verifies that there are
3   * only even integers in the stack.
4   *
5   * @return      false if there is any odd element,
6   *                      true otherwise.
7   */
8  public boolean areAllElementsEven()
9  {
10   boolean result = true;
11   for(int i = 0; i<h;i++)
12   {
13     if(elts[i]%2 == 0)
14     {
15       result=true;
16     }
17     else
18     {
19       result=false;
20     }
21   }
22   return result;
23 }
```

Sie erhalten den Auftrag, die neue Methode zu testen. *Hinweis:* In der Dokumentation Ihrer Testfälle benutzen Sie die folgende Darstellung für der Inhalt des Stapels: $\{n_1, n_2, \ldots, n_k\}$. Dabei ist n_1 das unterste und n_k das oberste (= zuletzt eingefügte) Element des Stapels.

a. Erstellen Sie Testfälle, welche die Ausgaben der Methode überdecken. Definieren Sie hierzu zwei Äquivalenzklassen und geben Sie für jede Äquivalenzklasse einen Testfall an. **(6 Punkte)**

Musterlösung

Klasse	Repräsentant	Erwarteter Output
Stacks ohne ungerade Elemente	$\{2, 4, 6\}$	true
Stacks mit einigen ungeraden Elementen	$\{2, 5, 4\}$	false

b. Geben Sie einen Testfall an, welcher einen Grenzwert für diese Methode testet. **(3 Punkte)**

Musterlösung

Grenze	Wert	Erwarteter Output
Leerer Stack	$\{\}$	true
Integer Repräsentation	$\{2^{32}\}$	true

c. Testen Sie diese Methode als Schreibtischtest mit den von Ihnen definierten Testfällen und dokumentieren Sie die Befunde. **(5 Punkte)**

Musterlösung

Die Methode liefert $false$ zurück, falls die letzte Nummer im Stack ungerade ist, ansonsten $true$. Zwei Antworten sind möglich für die Dokumentation: entweder der Code funktioniert für die Testfälle, oder nicht.

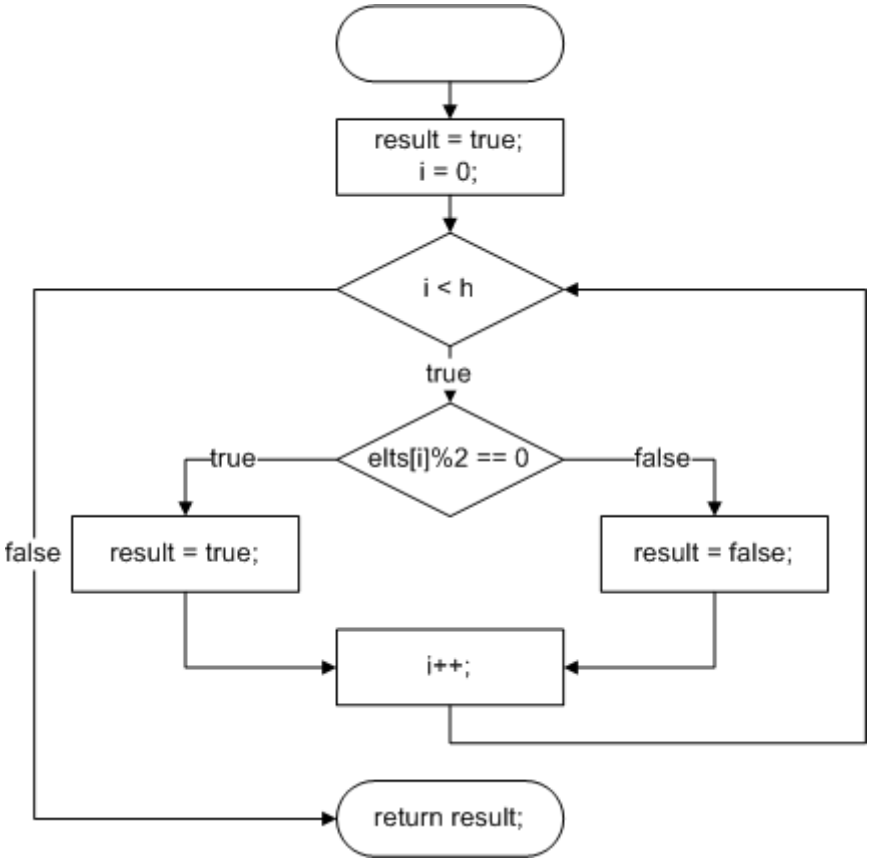


Abbildung 1: Flussgraph der Methode `areAllElementsEven()`

d. Abbildung 1 repräsentiert den Flussgraph der Methode `areAllElementsEven()`. Wie viele Zweige gibt es? Wieviel Prozent Zweigüberdeckung erreichen Sie mit dem in Teilaufgabe c. durchgeführten Test? **(2 Punkte)**

Musterlösung

4 Zweige. 100%.

e. Ist es möglich, 100% Zweigüberdeckung mit nur einem Testfall zu erreichen? Begründen Sie. **(2 Punkte)**

Musterlösung

100%. Ja, es ist möglich. Ein Stack mit zwei Elementen, einem ungeraden und einem geraden, erreicht 100% Zweigüberdeckung.

f. Berechnen Sie die zyklomatische Komplexität dieser Methode. **(2 Punkte)**

Musterlösung

$$v(G) = \#_{Kanten} - \#_{Knoten} + 2 \cdot \#_{Endpunkte} = 8 - 7 + 2 \cdot 1 = 3$$

oder

$$v(G) = \#_{If} + \#_{For} + 1 = 1 + 1 + 1 = 3$$