

## **Gedächtnisprotokoll**

### **SoftwareEngineering II (Methodische Entwicklung von Web-Applikationen) - 1794**

**Prof. Six**

**Sommersemester 2009**

**Dauer: ca. 30 min**

**Note: 3**

Die Prüfungsfragen aus den schon vorhandenen Protokollen möchte ich nicht wiederholen. Allerdings hat mir Prof. Six diese nur in sehr geraffter Form gestellt und dafür neue Aspekte beleuchtet die ich leider nur sehr zögerlich und mit Hilfen beantworten konnte – deshalb die obige Note. Dennoch würde ich den Kurs und Prof. Six als Prüfer empfehlen.

Die relativ neuen Aspekte waren:

1. Zweite Möglichkeit (neben dem Session Scope) einen Warenkorb zu speichern (statefull Session Beans). Wann verwendet man den Scope und wann die Beans?
2. Was sind gute und was sind schlechte Architekturen? Was beeinflusst neben den funktionalen und nichtfunktionalen und ... Anforderungen noch die Architektur? (hier wollte er auf AWF-unabhängige Komp. hinaus
3. Überraschende Thematik „Verfeinerung und Ergänzung des Entwurfsklassenmodells“: Hier wollte Prof. Six wissen was man in den einzelnen Schichten noch machen muss – also nochmals nachlesen.

Viel Glück

**Prüfungsprotokoll**  
**Software Engineering II (Kurs 01794)**  
**13.03.2008**

Fachprüfung MCompScience

Prof. Dr. Six

Note: 1.3

Die Prüfung fand in einer netten, entspannten Atmosphäre statt. Prof. Six ist ein angenehmer Prüfer, der viel nachfragt und auch selber relativ viel redet. Längere Ausführungen von Seiten des Prüflings sind also nicht gefragt.

Bei mir ging es in der Prüfung eigentlich nur um KE 1&2 sowie KE 6, was natürlich nicht viel heißen muß.

Ich wurde ziemlich ausführlich zu den Themen Session, Scopes, JavaBeans, Requestablauf (nicht detailliert) sowie anforderungsorientierte Aktivitäten (aoA) befragt. Ich habe teilweise die „A“-Wörter (Aktivität, Aktion, Anforderung...) etwas durcheinandergeschmissen, was aber nicht so schlimm gewesen zu sein scheint.

Hier die Fragen, soweit ich mich erinnere:

- Erzählen sie mir mal was über Sessions – Wozu, wie realisiert?
- Wie funktioniert das mit den Cookies und den zirkulierenden Session-ID's?
- Cookies erscheinen natürlicher – warum werden nicht nur cookies benutzt?
  
- Beispiel warenkorb – wo werden daten abgespeichert? (scope)
- welche möglichkeit gibt es noch?
- was ist ein scope, welche arten von scopes gibt es?
- Mit welcher art java-klassen werden scopes realisiert?
- Was ist das besondere an java beans?
- Haben sie nur getter und setter methoden oder auch andere?
- Warum werden sie so einfach gehalten? (jeder soll nur so viel wissen, wie er für seine aufgabe wissen muss)
  
- request-ablauf mit detailierungsgrad mvc-pattern (request-controller-model...)?
- jsp und servlet – wer benutzt getter, wer setter methoden der javabeans?
  
- aoA <->uml-aktivitäten: welche unterschiede, warum?
- Was sind szenen? Welche information enthalten sie? Berücksichtigen sie das UI-Design?
- Wann ist die zusammenfassung mehrerer systemtätigkeiten zwischen 2 akteuraktionen eine systemaktion, wann eine systementscheidung?
- Beispiel für systemaktion ohne anwendungslogik?
- Warum wird uml-verträglichkeit aufgegeben?

Ich kann den Kurs und auch Prof. Six als Prüfer sehr empfehlen.

## Gedächtnisprotokoll SW Engineering 2

=====

Dauer: ca. 30-40 Min (wir sind öfters abgeschweift)

Note: 1.0

Version des Scripts: aktueller Stand zum WS 2007/2008

### \* Gesamteindruck

Herr Prof. Six ist ein sehr netter; er formuliert Fragen gut und ruhig, sodass man genug Zeit zum Mitdenken hat. Auch, wenn man mal eine Frage nicht ganz korrekt verstanden hat und die Antwort leicht daneben liegt, bekommt man das später wieder hin. Die Prüfung läuft sehr ruhig ab und zwischen den einzelnen Fragen erzählt Prof. Six noch etwas ausführlicher Ergänzendes zu der Antwort und/oder Frage (auch als Überleitung). Mir ist allerdings aufgefallen, dass es manchmal einen nicht zu Wort kommen lässt und man daher die Frage nicht vollständig beantworten kann (mir ging das 2mal so); also: Beeilung ;-)

### \* Themen

Folgende Themen wurden angesprochen:

#### 1. Session

- Wozu - wegen Verbindungslosigkeit, Zustand wird aber bei einigen Anwendungen benötigt
- Wie realisiert man das - es liegen dazu Beans im Session-Scope, auf die man innerhalb des Webcontainers zugreifen kann
- Stichwort Client/Session-ID, wie wird die übermittelt (die 3 Verfahren)
- Warum wird die Verwaltung der ID nicht dem Client überlassen

#### 2. Servlets/JSPs

- Request-Ablauf eines Servlets
- Was macht ein Servlet
- Was sollte und was kann ein JSP alles tun (sind ja zwei unterschiedliche Sachen!)
- Welche Mittel gibt es, in einer JSP auf ein Bean im Scope zuzugreifen (Scriptlets, EL - EL ist die bessere Variante)
- Warum ist die EL besser als Scriptlets
- JSTL sollte man kennen (die kann man auch statt EL verwenden); was bietet die zusätzliche zu EL
- JSPs sind auch nur Servlets, können also genau das gleiche tun - sollten aber nur Anzeige übernehmen

#### 3. Architektur

- Wie sahen in ABE-Web die Feinstrukturierungen der Architektur aus?
- Vielleicht man ein Beispiel zu übergreifenden und gemeinsamen AWF-Komponenten geben

#### 4. Softwarespezifikation

- Könnte man die SW-Spezi auch entfallen lassen? - Im Grunde ja, kann ja eh alles automatisch generiert werden; für didaktische Zwecke aber trotzdem guter Zwischenschritt

So, mehr weiß ich nicht mehr. Vielleicht als Schlusssatz: Kleinigkeiten auswendig lernen ist unnötig, so was wird nicht gefragt. Der Zusammenhang ist wichtig. Viel Erfolg

Prüfungsprotokoll Software Engineering II - 01794 Januar 2007

Prüfer: Prof. Six

Beisitzer: Alexander Lorenz

Note: 1.0

Architektur:

F: Was verstehen wir unter Softwarearchitektur?

(Zwei Sachen sollen genannt werden: Aufteilung in substantielle Teile, etc. und wie diese Komponenten miteinander interagieren ("Die Festlegung der Interaktion wird gern vergessen."). Architektur ist eine grobgranulare Aufteilung, Abgrenzung zum Entwurf, Wichtig: Architektur ist "global" vs. Entwurf der lokal ist)

F: Was heisst global?

(Arch. lässt sich nur als ganzes betrachten und festlegen. Entwurf kann man dagegen iterativ in Teilen bearbeiten)

F: Welche Architektur haben wir im Kurs gewählt?

(Schichtenarchitektur überlagert mit einer MVC Architektur)

F: Wie sah das dann aus?

(V und C in der Webschicht, das M wird unterteilt in die Anwendungslogik und die Anwendungsobjekte)

F: Warum haben wir diese Architektur gewählt? Warum reichte uns nicht einfach die 5-Schichtenarchitektur?

(Feinere Aufteilung, bessere Abgrenzung der Aufgaben der Komponenten)

F: Wie haben wir diese Aufteilung dann praktisch realisiert?

(Über Aufteilung in Pakete. Die Paketaufteilung sollte man kennen.)

Aktivitäten:

F: Wozu haben wir nach der normalen UML-Aktivität denn interaktionsorientierte Aktivitäten eingeführt und worin besteht der Unterschied?

(Szenen und Stereotypen; Szenen: gemeinsame Modellierung von Kontrollfluss und graph. Benutzeroberfläche (daher "interaktionsorientiert"), Stereotypen: "Wer macht was?", Sichtbarmachung der Verantwortlichkeiten von Akteur vs. System)

F: Und die anforderungsorientierten Aktivitäten? Was ist dann bei denen anders? Und warum?

(Vereinfachte Modellierung durch gemeinsame Modellierung von Aktion und Entscheidungsknoten und Unterdrückung bzw. Weglassen von Systemaktionen; Lesbarkeit für den Anwender, Grundlage der Validierung)

F: Warum modelliert man Aktionen und Entscheidungsknoten in einem Knoten?

(Verständlicher für den Anwender; "Man hat einfach weniger Modellelemente im Diagramm bei gleicher Ausdruckskraft.")

F: Und warum belässt man es später nicht bei dieser Modellierung?

(Man will später die UML-Konformität wiederherstellen.)

F: Welche Systemaktionen lässt man denn bei den anforderungsorientierten Aktivitäten weg und welche modelliert man?

("Ergibt sich aus der Diskussion mit den Anwendern." Eine Systemaktion die z.B. ein für den Anwender ein wichtiges Ergebnis hat, darf natürlich nicht unterdrückt werden.

Systemaktionen wie "Eingaben speichern"

gehören aber nicht ins anforderungsorientierte Aktivitätsdiagramm)

Entwurf:

F: Wie kommen wir von der Softwarespezifikation zum Entwurfsmodell?

(Abbildung der Aktivitäten in das Klassenmodell des Entwurfs anhand systematischer Regeln)

F: Welche Regeln sind denn das?

(Die 11 Transformationsregeln erklärt; es kamen Zwischenfragen zum Verständnis, reines Auswendiglernen reicht also nicht.)

Da wir am Anfang noch über den Kurs allgemein, die Strukturierung und die Modellierung der komplexen Systemaktionen diskutiert haben, war die Zeit damit auch schon um. Die Antworten in Klammern oben sind nur als Hinweis zu verstehen und müssen nicht richtig bzw. vollständig sein.

Der Schwerpunkt der Fragen lag deutlich auf dem Software-Engineering, also auf der systematischen, modellbasierten Vorgehensweise und der Modellierung, und nicht auf den technischen Voraussetzungen (KE 1+2, KEs zu Struts und EJBs). Trotzdem würde ich die technischen KEs nicht ignorieren. Ich denke auch dazu können durchaus Fragen kommen, besonders wenn man bei den damit verbundenen Fragen zum Software-Engineering Lücken zeigt.

# Prüfungsprotokoll Diplomprüfung Software Engineering II

Prüfer Prof. Dr. Six  
21.01.2002 11.30 Uhr ca. 30 min  
Note 1.0

## Prüfungseindruck

Prof. Six ist ein sehr guter Prüfer. Er versucht eine angenehme Prüfungsatmosphäre zu schaffen, indem er zuerst nach dem Background zu SE fragt, also über Beruf, Studium, nach der Motivation, warum man in diesen Kurs eine Prüfung ablegt. In der Prüfung führt er sehr genau in jedes angeschnittene Thema ein, und formuliert sehr detaillierte Fragen. Zusätzlich diskutiert er nach Beantwortung noch praktische Aspekte, so dass sich ein sehr lockeres Gespräch über SE II ergibt. Er setzt Schwerpunkte, somit werden in der Prüfung nicht alle Themen angeschnitten, sondern er geht auch mal in die Tiefe. Wichtige Themen waren Anforderungsermittlung, Generalisierung und man sollte auch ein paar Parallelen zu SEI parat haben. Zu allen Punkten bietet es sich an, Beispiele anzugeben, ruhig auch aus dem Kurs, manchmal fragt er explizit danach. Ich habe versucht die Fragen nicht nur knapp zu beantworten, sondern auch den Zusammenhang herzustellen. Seine Benotung ist sehr fair, zudem ich bei ein paar Punkten unsicher war, und er die Frage neu formulieren musste (v.a. bei Generalisierung), bzw. auch mal kein Beispiel parat hatte (bei *extend* von Anwendungsfällen).

## Prüfungsverlauf

### Einstieg mit Anforderungsermittlung

Nennen und erläutern der drei Modelle  
(funktionales M. – Anwendungsfalldiagramm  
strukturelles M. – Klassendiagramm  
verhaltensorientiertes M. – Interaktionsdiagramm  
Zustandsdiagramm)

### Anwendungsfälle

was beschreiben diese  
(Teilfunktionalität des Anwendungssystems)

wie sind sie textuell spezifiziert  
(*use cases, actors, main flow, exeptional flow*,  
genau erläutern mit Beispiel)

welcher Umfang  
(ca. eine DinA 4-Seite)

wie formal  
(nicht zu formal, da Anwender, die Anwendungsfälle noch verstehen soll)

## Akteure

was wird mit Akteuren modelliert  
(Abstraktion der Benutzer, die gegenüber dem Anwendungssystem in verschiedenen Rollen auftreten können)

was kann sich noch dahinter verbergen  
(externe Systeme)

welches analoges Modellkonstrukt gibt es in SE I  
(Begrenzer)

## Beziehungen zwischen Anwendungsfällen

welche gibt es  
(*include*, *extend*, genau erläutern mit Beispiel für *extend*)

warum diese Modellierung mit Beziehungen  
(Komplexität ?, Wiederverwendung ?)

## Klassenmodell

was beinhaltet eine Klasse  
(Attribute, als Namen der Eigenschaftswerte  
Operationen)

Sichtbarkeit  
(*public*  
*protected*, Zugriff nur in *packages*  
*private*, alle Begriffe erläutern)

sollte in der Anforderungsermittlung schon die Sichtbarkeit festgelegt werden  
(gehört eher zum Entwurf, da *private* innere Funktionalität beschreibt  
in Anforderungsermittlung die externe Benutzung des Anwendungssystems  
beschrieben wird)

welche Beziehungen zwischen Klassen gibt es  
(Assoziationen, Navigierbarkeit, Multiplizität,...  
Aggregation und Komposition genau erläutern mit Beispielen,  
z.B. aus dem Kurs)

## Generalisierung

### Definition

(formal über Konformität)

wie kann sie im ‚Alltag‘ des Programmierers ohne die formale Prüfung eingesetzt werden

(hier hatte ich etwas Probleme eine passende Antwort zu geben)

wozu wird sie eingesetzt, welche Vorteile bietet Generalisierung

(Entwurf mit Verträgen

Test, hier gab es einen sehr kurzen Ausflug zum Thema Test)

wobei noch

(Wiederverwendung)

welches Prinzip gilt hier

(Substituierbarkeitsprinzip, mit Erläuterung)

wann erkennt Programmierer dass er Generalisierung und nicht das allgemeinere Konzept der Vererbung einsetzt

(hier wieder etwas Probleme,

im Wesentlichen, z.B. keine Operationen überschreiben, etc.)

(Die Fragen wurden viel detaillierter gestellt, als ich sie hier wiederzugeben versuchte)

Viel Glück in deiner Prüfung

Gedächtnisprotokoll mündliche Prüfung 1794

Datum: 13.06.01

Prüfer: Prof. Dr. Six

Vergleich modulare Architektur zu oo-Architektur:

- welche Architektur ist durchgängiger über die Phasen des Entwurfs
  - modular: aus Datenfluss müssen die Operationen der Datentypen zusammengesucht werden, ansonsten kann das ER-Modell in ADT umgesetzt werden. Das Datenflussmodell kann nicht durchgängig auf funktionale Moduln abgebildet werden
  - ooEntwurf: das Klassenmodell kann direkt auf die Datenhaltungsschicht abgebildet werden, das Anwendungsfalldiagramm wird auf Kontrollklassen und Schnittstellenklassen verteilt und ist deshalb nicht direkt abbildbar.
- Insgesamt Vorteile für oo, aber auch hier keine optimale Durchgängigkeit
- welche Modelle verwendet die Anforderungsspezifikationen im modularen Entwurf und im oo-Entwurf

Welche Schichten gibt es in der drei Schichten Architektur

Wo finden sich die einzelnen Modelle aus der Anforderungsspezifikation wieder (Klassenmodell in Datenhaltungsschicht, Anwendungsfälle in Kontrollklassen und Benutzerschnittstellen)

Nach welchen Kriterien werden Anforderungsfälle zu Paketen in der Anforderungsspezifikation zusammengefasst (Anforderermittlung richtet sich nach dem realen Leben!)

Welche weiteren Gliederungen/Beziehungen für Anwendungsfälle gibt es (include/extend)

Wie wird das Zustandsdiagramm der Anforderungsspezifikation im Feinentwurf umgesetzt (gibt es dazu eine Klasse?)

Wann ist Vererbung sinnvoll (gebundenesBuch/Taschenbuch, Zeitschrift/Buch)

Herr Six legt viel Wert auf einen Überblick über Softwareengineering, dabei wird auch (allgemeines) Wissen aus SE I vorausgesetzt. Wenig konkrete Fragen und wenig Details aus dem Kurstext. (Das mag aber auch daran liegen, dass ich zeitgleich das vom Lehrstuhl angebotene Softwarepraktikum besucht habe) Die Benotung ist sehr fair.

## **Kurs Software Engineering II (1794) Fassung vom WS 1995/1996**

**Dauer 25 min**

Prüfer Prof. Six

Von den Gebieten Anbindung externer Systeme, Anbindung relationaler Datenbanksysteme, Ada/C.

Objektorientierte Analyse, graphische Benutzungsoberflächen. Programm-Verifikation. Wiederverwendung und Prototyping konnten aus Zeitgründen nur OOA und GUI geprüft werden.

Wodurch unterscheidet sich die OOA von der MSA ?

MSA : drei Modelle (ER-Diagramm, Datenflußmodellierung und Zustandsübergangdiagramm), OOA ein integriertes Modell.

Was sind die Primitive der OOA ?

Objekte, Klassen. gerichtete und ungerichtete Objektbeziehungen, Vererbungsbeziehungen. Nachrichtenkanäle und Themenbereiche.

Was ist der Unterschied zwischen Klassen und Objekten ?

Objekte können anonym oder ansprechbar sein. Hier sagte ich, daß dies den Realisierungs- bzw. Architektur-ADOs entspräche, was dazu führte, daß ich aus Nervosität die A&D-Phase der OOA mit der Entwurfs-Phase der MSA durcheinanderwarf. Dies hatte allerdings keine weitere Auswirkung auf den Verlauf der Prüfung.

Was entspricht den Klassen der OOA in der MSA ?

Den Entitäten bis auf die Tatsache, daß dort keine Funktionalitäten mitmodelliert werden. Ebenso werden die Kassendienste nicht dargestellt. Dies führte zu einer Diskussion über Standard- und Klassendienste.

Welche Objektbeziehungen gibt es ?

Konstruktions-. Behälter- und logische Zusammenschlüsse. Für diese wurden Beispiele erwartet.

Was charakterisiert gerichtete Objektbeziehungen gegenüber ungerichteten Objektbeziehungen ?

Bei den gerichteten Objektbeziehungen trägt das übergeordnete Ganze Verantwortung für die untergeordneten Teile. Dies drückt sich beim Löschen des übergeordneten Teiles aus.

Was für Beziehungen gibt es auf Klassenebene ? Vererbung.

Was ist das allgemeinere Konzept Generizität oder Vererbung ? Vererbung.

Ist Generizität aus der MSA nicht dasselbe wie Vererbung bei der OOA ?

Nein, denn Generizität erlaubt nur das Erstellen einer Schablone, die nicht dynamisch erweitert werden kann. Hieran schloß sich eine Diskussion über dynamische und statische Aspekte der OO an.

Wie werden komplexe Situationen bei den Nachrichtenkanälen dargestellt ?

Durch Situationsbeschreibungen.

Wo sind die Zustandsübergangdiagramme der MSA geblieben ?

Sie werden jetzt auf Klassen- bzw. Objektebene zur Zustandsbeschreibung einge-

setzt.

Wozu wird Aufgabenmodellierung betrieben ?

Erste Strukturierung der erwarteten Funktionalitäten der Benutzungsoberfläche.

Was ist das Ergebnis der Aufgabenmodellierung ?

Eine strukturierte natürlich-sprachliche Beschreibung der Hauptfunktionalitäten des Systems.

Wozu ist Abstimmung mit dem statischen Applikationsmodell notwendig ?

Um zu gewährleisten, daß die Funktionalitäten der Klassen auch alle erforderlichen Benutzungsoberflächen-Operationen unterstützt.

Welcher Aspekt der OOA wird durch die Aufgabenmodellierung unterstützt?

Die Modellierung der Nachrichtenkanäle (Szenarien).

Auf was setzt die UI-Analyse auf ?

Auf den Ergebnissen der Aufgabenmodellierung und dem dynamischen Applikationsmodell.

Was ist das Ergebnis der UI-Analyse ?

Formale Beschreibung der Benutzungsoberfläche.

Welches sind die Primitive der UI-Analyse ? Szenen, Anker, Sichten und Benutzeraktionen.

Was wird später aus den „Szenen“ ?

Daraus werden im wesentlichen die Fenster.

Warum ist neben der Klassensicht noch der Klassenanker notwendig ?

Der Klassenanker verbindet das Applikationsmodell mit der Benutzungsoberfläche. Die Klassensicht gibt an, welche Attribute und Objektbeziehungen der referenzierten Klasse dargestellt werden sollen.

Wozu gibt es multiple Klassenanker?

Um die Bearbeitung der gleichen Objekte in verschiedenen Fenstern sicher zu stellen.

Wie werden die Ergebnisse der UI-Analyse in der Entwurfsphase weiterverwendet?

Erstellung der Dialogkontrolle, der Präsentationskontrolle und der Applikationsschnittstelle. (Beschreibung der zugehörigen Event-Response-Systeme).

Hier war die Zeit zu Ende. Themen aus dem Bereich der Verifikation wurden daher nicht mehr abgehandelt, womit ansonsten zu rechnen gewesen wäre.

Fazit: Faire und detaillierte Prüfung. Gut vorbereitet ist eine gute Note erreichbar.