Prüfungsprotokoll Praktische Informatik

Prüfer:	Prof. Dr. J. Borchers Alexander Nyßen (in Vertretung von Prof. Dr. H. Lichter)
Inhalt:	Designing Interactive Systems I (Borchers), WS 2004/05 Designing Interactive Systems II (Borchers), SS 2005 Software-Qualitätssicherung und Projektmanagement (Nyßen), SS 2005
Datum:	Juni 2006
Dauer:	45 Minuten
Vorbereitungszeit:	Drei Monate
Note:	1.0

1 Bemerkungen

Die Fragen in diesem Prüfungsprotokoll sind nur sinngemäß wiedergegeben. Ich denke, dass das Protokoll die geprüften Themen ganz gut abdeckt, allerdings kann ich die Vollständigkeit (oh Wunder) nicht garantieren. Außerdem sind sowohl die Fragen als auch die Antworten stark gekürzt. Ich hoffe, dass das Dokument dennoch für die Vorbereitung hilfreich ist. Falls Fragen aufkommen, bitte einfach per Mail an malteweiss@gmx.de. Zu Beginn ein paar Anmerkungen zu den einzelnen Fächern. Jedes Fach wurde ziemlich genau 15 Minuten lang geprüft.

1.1 Designing Internative Systems I+II

Obwohl DIS sicherlich zu den Vorlesungen gehört, die leicht verständlich sind und den gesunden Menschenverstand ansprechen, sollte man sich nicht davon abbringen lassen, den Stoff **detailliert und vertieft** zu lernen! Prof. Borchers erwartet nicht nur, dass das Wissen komplett bis zu einem hohen Detailgrad auf Abruf bereit steht, sondern auch, dass man es verinnerlicht hat und es bei konkreten Situationen **anwenden** kann. Desweiteren kann ich nur empfehlen, die **Vorlesungsvideos anzuschauen**, da dort einige Aspekte deutlich umfangreicher erläutert werden.

Was DIS I betrifft, halte ich es für sinnvoll, zwei Themen genauer zu erarbeiten, da sie in den Folien nur komprimiert Erwähnung finden, aber definitiv prüfungsrelevant sind:

- **Die Geschichte von HCI** vom Abakus über Sketchpad bis zum Apple Macintosh. Dieses Thema kommt in den Folien nur zusammengefasst vor, wurde aber in der Vorlesung *ausführlich* behandelt (Vorlesungsvideos anschauen!). Ich denke, dass die GUI-Computer (Xerox Alto, Xerox Star, Apple Lisa, Apple Macintosh) besonders wichtig sind.
- **Die mathematischen Modelle**, die vorgestellt worden, sprich: CMN-Modell, Fitts' Law, GOMS. Diese Modelle sollte man an Beispielen *anwenden* können.

Ein Blick in das Buch *The Design of Everyday Things* kann sicher nicht schaden, um gute Beispiele zu finden.

Bei DIS II gilt eigentlich das gleiche: es ist sicher hilfreich, ausgewählte Vorlesungsvideos nochmal anzuschauen. Zum SubArtic-Toolkit und den Multimedia Interfaces (in der Vorlesung von SS 2006 nicht mehr enthalten) gibt es desweiteren gute Paper. Ich wurde bei DIS II über den Stoff vom SS 2005 geprüft.

Im darauffolgenden Jahr haben sich ein paar Themen geändert, z.B. sind Symbian OS und Web 2.0 hinzugekommen, dafür ist das Kapitel über Multimedia Interfaces herausgenommen worden.

Wesentlich ist also: Wissen verinnerlichen, Querverweise bilden und Stoff anwenden können. Prof. Borchers bezeichnet dies als *Toolbox*-Prinzip. Nachdem ich jetzt jede Menge Angst verbreitet habe, hier noch ein paar beruhigende Gedanken: Der Stoff lässt sich gut vorbereiten und Prof. Borchers ist ein fairer Prüfer. Die Atmosphäre war sehr angenehm und man lernt auch noch einiges dabei. Da ich in meinem Redefluss das ein oder andere englische Wort falsch aussprach und damit den Sinn veränderte, gab es auch einiges zu lachen :)

1.2 Software-Qualitätssicherung und Projekt-Management

Dieses Fach wurde von Alexander Nyßen geprüft, da Prof. Lichter kurzfristig erkrankt war. Herr Nyßen stellte klar verständliche Fragen, die sich nach intensivem Lernen des Stoffs problemlos beantworten ließen. Auch bei diesem Prüfungsteil war die Atmosphäre angenehm und entspannt.

Zum Lernen empfehle ich die Vorlesungsfolien. Sie sind umfassend, nahezu fehlerfrei und gut ausgearbeitet, so dass eigentlich kaum Fragen offen bleiben. Bei ausgewählten Kapiteln bin ich dann aber doch in die Vorlesung gegangen, wodurch bestehende Unklarheiten ausgeräumt wurden. Wer Lust hat, sich noch etwas Hintergrundwissen anzueignen, kann *Software-Qualität. Testen, Analysieren und Verifizieren von Software.* von *Peter Liggesmeyer* lesen, allerdings war das Buch für die Prüfung nicht relevant. Neben den Folien waren die **Prüfungsprotokolle** äußerst hilfreich. Es gab ehrlich gesagt keine Frage, die mich wirklich überrascht hat.

2 Prüfungsfragen

Normal gedruckte Sätze sind Aussagen oder Fragen vom jeweiligen Prüfer. Meine Antworten und Gedanken sind *kursiv* gedruckt.

2.1 Designing Interactive Systems I

Diesen Prüfungsteil konnte ich leider nicht sonderlich gut rekonstruieren. Prof. Borchers fragte nicht sequentiell. Es entstand eher eine Art Gespräch. Desweiteren stellte er oft mehrere Fragen gleichzeitig.

- Wie wir ja schon in unserem Vorgespräch besprochen haben, kommt es mir nicht nur darauf an, Wissen abzuprüfen. Sie sollen auch Querverweise herstellen und das Gelernte anwenden können.
  Äh, also eigentlich hatten wir gar kein Vorgespräch ...

- (Borchers grinst) Okay, dann war das jetzt die Standardflosskel zum Beginn einer DIS-Prüfung.

- Wir haben in der Vorlesung einige Modelle kennengelernt. Welche waren denn eher mathematischer und formaler Natur?
  Das CMN-Model, Fitts' Law, und am Ende der Vorlesung noch GOMS. Die statistische Analyse beim kontrollierten Experiment ist streng genommen auch mathematischer Natur.

- Dann erklären Sie doch mal das GOMS-Modell.
  Ich erläuterte nun ausführlich das CMN-GOMS (Nachfolger vom KLM). Ich betonte dabei, dass es routinemäßige/alltägliche Aufgaben abbildet und die Zeitwerte, die den Operatoren zugeordnet werden, Durchschnittswerte darstellen.

- Welche Modelle gibt es denn noch?
  Hier habe ich dann KLM, NGOMSL, CPM und EPIC erläutert.

- Wie würden Sie denn nun mit GOMS das Wählen einer Telefonnummer auf einem Handy modellieren? Wie gehen Sie denn da vor.
  Das lief nicht ganz so flüssig. Ich bezog die Frage erst auf CMN-GOMS (hierarchische Goal-Struktur), aber Prof. Borchers wollte es wohl am sequentiellen KLM gezeigt haben. Nach einigen Rückfragen über die konkrete Situation (z.B. dass die konkrete Telefonnummer vorgegeben ist), kreierte ich dann eine Liste von Operatoren. Prof. Borchers wollte dann, dass ich die Werte für die Operatoren „Finger bewegen", „Knopf drücken" schätzte. Habe ich dann auch gemacht. Es waren einige Rückfragen meinerseits nötig und die Aufgabenstellung hat mich ein wenig irritiert.

- Dann erläutern Sie mal Fitts-Law.
  Gesagt, getan (Ausführungszeit für zielgerichtete 1D-Bewegungen abhängig von Größe und Distanz, Formel aufgeschrieben).

- Welche Variationen gibt es denn da?
  Welfords und Shannons Law erläutert und Vorzüge erklärt (bessere Kurvenform, bei Shannon keine Konvergenz ins Negativ-Unendliche).

- Wie würde man das denn graphisch darstellen?
  Habe dann wie in den Folien den Index of Difficulty gegen die Mean Time abgetragen und eine idealisierte Linie eingezeichnet.

- Jetzt nehmen wir mal an, wir verdoppeln die Größe des Ziels. Wo liegen denn dann die Punkte auf dem Graphen?
  Hier hat mir Prof. Borchers eine Falle gestellt und ich bin mit Freuden reingetappt. Die Punkte liegen natürlich da, wo sie vorher lagen. Warum das so ist, überlasse ich dem Leser ;) Ich dachte jedenfalls, dass sie darunter liegen würden. Nachdem ich Alles nochmal durchdacht hatte und Prof. Borchers mich zweifelnd ansah, habe ich dann „über der ursprünglichen Kurve" geraten. Das dritte Raten war dann natürlich erst richtig. Keine Glanzleistung.

- Die amerikanische Tastatur sieht im Gegensatz zu unserer eine horizontale Enter-Taste vor. Welches Layout denken Sie ist denn besser unter Anwendung von Fitts' Law?
  Knifflig. Ich modellierte das Tippen als Bewegung, die vom Zentrum der Tastatur ausging. Da Fitts' Law eindimensionale Bewegungen zu Grunde legt, ist die deutsche Enter-Taste in der Projektion größer und damit schneller zu erreichen.

- Jetzt finden Sie aber auch eine Argument für die amerikanische Variante. Denken Sie mal daran, wie Sie tippen ...
  Mit dem Hinweis konnte ich absolut nichts anfangen.

- Na ja, es ist ja eine Bewegung von oben nach unten auf die Tastatur.
  Nach ein bisschen Grübeln habe ich mir dann überlegt, dass die Bewegung auf einer YZ-Ebene aus Sicht des Betrachters abläuft, also keine seitliche Bewegung stattfindet. Damit ist die vertikale Enter-Taste in der Projektion größer als die amerikanische Variante. Das war richtig.

  Ich war bis zu dem Zeitpunkt nicht wirklich zufrieden mit der Prüfung. Obwohl ich Alles gelernt hatte, lief es nicht so flüssig. Aber ich denke, dass Prof. Borchers wissen will, wie kreativ man unter dem Druck arbeitet. Die Fehler, die ich bis hierhin machte, fielen bei der Bewertung nicht so stark ins Gewicht.

- Wir haben ja über die Geschichte von HCI gesprochen. Welcher GUI-Rechner denken Sie hat den größten Entwicklungssprung gemacht?
  Ich habe den Xerox Alto von 1973 ausgewählt und im Detail erläutert. Habe am Ende noch Smalltalk erwähnt und das MVC-Modell als Vorteil herausgestellt. War ganz gut, denn danach wurde ich dann in DIS II gefragt :-)

- Danach kam ja der Xerox Star. Hat ja auch lange gedauert, den zu entwickeln...
  Acht Jahre.

- Was hat ihn denn ausgezeichnet?
  Habe zunächst mal ein paar Key-Features erwähnt: Desktop-Metapher, Netzwerkumgebung, Property Sheets usw. und bin dann im Detail auf den Design-Prozess eingegangen.

2.2 Designing Interactive Systems II

- Sie sprachen eben Smalltalk an. Was ist das denn?
  In erster Linie eine objektorientierte Programmiersprache. Allerdings auch ein vollständiges Window System. Virtual Machine und Virtual Image (Smalltalk-Klassen) erläutert. Habe noch Morphic eingestreut, um später danach gefragt zu werden ;)

- Was können Sie denn über das darunter liegende Betriebssystem sagen?
  Single process structure, single address space, Kommunikation über Prozedur-Aufrufe (koorperatives Multitasking).

- Was macht Morphic denn so besonders im Vergleich zu anderen Toolkits?
  Ich habe erstmal die zentralen Begriffe genannt: Structural Reification, Layout Reification, Directness, Liveness und Ubiquitous Animation. Dann fragte ich erstmal, wie man Reification ins Deutsche übersetzt. Offenbar gibt's da keine passable Übersetzung zu :)
  Ich erläuterte dann die Structural Reification und Layout Reification, allerdings bat mich Prof. Borchers darauf hin, dies mit Java Swing zu vergleichen. Dabei stellte ich dann heraus, dass diese beiden Technologien letztendlich Morphic nicht wirklich auszeichnen. Sie sind heute in fast jedem Toolkit üblich.
  Daher erklärte ich die Directness, Liveness und die Animationstechniken, die ja nun wirklich einzigartig sind.

- Beschreiben Sie mal die Klassenhierarchie von Mac OS X im Hinblick auf Objective-C? Was macht die Sprache so besonders z.B. im Vergleich zu C++?
  Ich habe erstmal ein paar einleitende Worte zu Objective-C gesagt (Ähnlichkeit zu Smalltalk usw.). Die Klassenhierarchie ist insbesondere sehr flach, was an drei Konstruktionen liegt, die sich miteinander kombinieren lassen: Delegates, Categories und Protocols.

- Erläutern Sie das mal.
  Das habe ich dann auch ausführlich getan. Die Erörterung hat insgesamt bestimmt 5 Minuten gedauert. Am Ende wurde ich noch gefragt, worin der Unterschied von Interfaces bei Java und bei Objective-C besteht. Die Interfaces von Java entsprechen den Protocols von Objective-C. Bei Objective-C gibt es aber auch Interfaces, die jedoch den Header-Dateien von C++ entsprechen :)

- Können Sie sich auch Nachteile vorstellen, die diese Techniken mit sich bringen könnten?
  Ich hatte mir im Vorfeld überlegt, dass dieser Ansatz zu extremer Unübersichtlichkeit und schlechter Wartbarkeit führen kann, besonders dann, wenn Programmierer willkürlich Klassen durch Delegates erweitern.

- Eine letzte Frage: Wir haben ja über Display devices gesprochen. Wie funktioniert eigentlich Electronic Inc? Was sind die Vorteile dabei?
  Ich habe die Technik dann erörtert: es gibt positiv/negativ geladene Kapseln in einem Substrat, die jeweils weiß oder schwarz darstellen. Ein Pixel wird dadurch gesetzt, dass Spannung induziert wird und sich dadurch entweder weiße oder schwarze Kapseln nach oben drängen (habe das an einer Zeichnung erläutert). Ich vergaß dabei zu erwähnen, dass die Kapseln ihrerseits in Kapseln (nämlich pro Pixel) liegen, war aber wohl nicht schlimm. Vorteile:

  - *Flexible Substrate möglich, dadurch kleinere einrollbare/ausziehbare Geräte konstruierbar.*
  - *Keine Energie notwendig, um Bild aufrecht zu erhalten.*

  Die Technik ist aber noch nicht ganz ausgereift.

2.3 Software-Qualitätssicherung und Projekt-Management

Software-Qualitätssicherung

- Die Vorlesung heißt ja Software-Qualitätssicherung und Projekt-Managment. In welcher Beziehung stehen denn diese beiden Begriffe? Ordnen Sie die doch mal ein.
  Hier war das Diagramm aus dem ersten Kapitel gefragt, dass Projekt-Management, Qualitätssicherung, Systemerstellung und Konfigurationmanagement in Beziehung setzt. Habe es aufgezeichnet und die Beziehungen zwischen den Komponenten erklärt.

- Was ist denn Qualität?
  Ich denke, die Frage kommt immer. Habe die ISO-Definition genannt und nochmal auf die entscheidenen Punkte hingewiesen: Qualität ist Zielerfüllung, muss geschaffen und in Form quantifizierbarer Anforderungen festgelegt werden.

- Nennen Sie mal ein paar Qualitätsmerkmale!
  Ich fing an, den Baum von Boehm aufzumalen. Irgendwann sollte ich damit aufhören ;-)

- Mit welchen Maßnahmen kann man denn gewährleisten, dass bestimmte Qualitätsmerkmale erreicht werden? Nennen Sie mal ein paar Beispiele.
  Organisatorische (Verantwortung defininieren, Richtlinien, Standards), konstruktive (Ausbildung, Hochsprachen), analytische Maßnahmen (Softwareprüfung, Metriken)

- Wie kann ich denn überprüfen, ob die Anforderungen hinreichend sind? Welche Verfahren haben wir kennengelernt?
  Hier habe ich zunächst mal den linguistischen Ansatz erläutert (Sprachtransformationen wie Tilgung, Generalisierung und Verzerrung, Anwendung von Regeln zum Aufspüren dieser Transformationen usw.)
  Dann habe ich die Reviews erwähnt und alle Varianten genannt.

- Welche Rollen gibt es denn bei Reviews?
  Habe die Rollen (Manager, Moderator, Autor, Gutachter, Schriftführer) genannt und ihre Aufgaben erläutert.

- Könnte man sich denn auch vorstellen, dass sich da Rollen überschneiden?
  Hmm, nein. Genau das will man ja ausschließen.

- Na gut, der Autor könnte noch den Schriftführer ersetzen. Aber ja, sie haben da schon Recht. Überschneidungen will man vermeiden.
  Wie könnte man denn sonst noch die Anforderungen überprüfen? Fällt Ihnen da noch eine Methode ein?
  Äh ... nein.
  Man könnte einen Prototypen bauen und ihm den Kunden vorführen.
  Borchers (grinst): Ja, sowas machen wir die ganze Zeit in DIS.

- Kennen Sie noch anderen Methoden der Software-Prüfung?
  Habe erstmal einen Überblick über die Software-Prüfung gegeben: Einteilung in dynamische und statische (mit/ohne Rechner) Prüfung. Ich wählte dann das Testen als Beispiel aus.

- Was ist denn Testen?
  Ausführung eines Programms mit der Absicht, Fehler zu finden. Ausnahme: Abnahmetest (erläutert).

- Wie sollte denn so ein Testfall sein?
  Repräsentativ, fehlersensitiv, redundanzarm, ökonomisch. Im Grunde geht es darum, mit möglichst wenigen Testfällen möglichst viele Fehler zu finden, und am besten solche, die hohe Fehlerfolgekosten verursacht hätten.

- Man kann ja nicht erschöpfend testen. Wie wählt man denn Testfälle aus?
  Durch ein Testauswahlkriterium (erläutert).

- Wann hört man denn mit dem Testen auf?

  1. *Beim Testauswahlkriterium genau dann, wenn es keine Fehler mehr aufdeckt.*
  2. *Wenn die relativen Fehlerkosten zu groß werden. Hier gibt es diese schöne Grafik mit den steigenden Fehlervorbeuge- und den fallenden Fehlerfolgekosten. Die sollte man drauf haben.*
  3. *Wenn die Wirksamkeit einer Prüfung einen gewissen Grad unterschreitet?*

- Wie kann ich denn feststellen, wie wirksam meine Methode ist?
  Nochmal kurz Wirksamkeit definiert (Anzahl der gefunden Fehler durch Anzahl aller Fehler) und bin dann im Detail auf Defect Seeding (und die Nachteile der Methode) eingegangen. Ich habe im Anschluss mal gefragt, ob die Technik überhaupt schon mal effektiv eingesetzt wurde. Ist offenbar keine gängige Praxis.

- Wir haben dann auch über Metriken geredet. Was ist eigentlich eine Metrik?
  Zuordnung einer Zahl oder eines Symbols zu einer Entität, um sie nach bestimmten Attributen zu charakterisieren. Wichtig ist, dass dies nach einer klar definierten Vorschrift vonstatten geht und man stets eine Interpretation braucht, um eine Metrik benutzen zu können.

- Wie lassen sich denn Metriken einteilen?
  Es gibt Produkt- und Prozess-Metriken. Danach fing ich an, die Attribute aufzulisten (einfach, berechnet, prognostisch usw.), die Liste wurde aber recht früh abgebrochen.

Projekt-Management

- Kommen wir nun zum Projekt-Management. Was ist denn eigentlich ein Projekt?
  Alle Tätigkeiten, Interaktionen und Resultate, um ein bestimmtes Ziel mit begrenzten Ressourcen in begrenzter Zeit zu erreichen. Ich wies insbesondere daraufhin, dass ein Projekt eine einmalige Tätigkeit ist.

- Was gehört denn so zum Projekt-Management?
  Planung, Monitoring, Reporting, Mitarbeiter-Motivation usw. Hatte Controlling nicht deutlich genug genannt, was Herr Nyßen nach Nachfrage dann selber nannte.

- Welche Stellschrauben gibt es denn beim Controlling?
  Entweder Steuermaßnahmen (Ist-Wert an Soll-Wert heranführen) oder Änderung des Projektplans (Soll-Wert an Ist-Wert heranführen).

- Ja, aber nehmen wir mal an, wir sind nicht im Zeitplan. Welche Möglichkeiten haben wir denn da?
  Da habe ich erstmal das Dreieck Leistung - Termine - Ressourcen aufgemalt und erläutert, dass sich diese drei Größen beeinflussen. Die Prüfung ging hier stärker ins Detail, was ich aber leider nicht mehr reproduzieren kann.

- Sie hatten eben Risikomanagement erwähnt. Wie läuft das denn ab?
  Risiken identifizieren, bewerten, Präventiv- und Notfallmaßnahmen planen, Maßnahmen überprüfen. Habe noch erwähnt, dass Risikomanagement eine kontinuierliche Tätigkeit ist.