Diplomteilprüfung Vertiefungsgebiet Media Comuputing

Prüfer: Borchers
Dauer: 55 Minuten
Note: 1.0

DIS1

Jan: Beginnen wir einfach mit dem CMN-Model. Was ist denn das überhaupt?
Ich: Erkläre Prozessoren, Speicher, Geschwindigkeiten
Jan: Und wofür kann man dieses Modell dann verwenden?
Ich: Zum Abschätzen von Reaktionsgeschwindigkeiten, Fehlerraten von Menschen bei Benutzerinterfaces.
Jan: Wie kann man das denn im Versuch feststellen?
Ich: Erkläre den Versuch mit den Strichen zwischen den Linien. Erkläre, woran man die gesamten 240ms und die 70ms des Motor-Prozessors ablesen kann.
Jan: Was für andere quantitative Verfahren ähnlicher Art hatten wir denn noch besprochen?
Ich: Fitts Law, Hicks Law, GOMS
Jan: Was macht man denn mit Hicks Law?
Ich: Erklärt. Logarithmisches Wachstum beschrieben.
Jan: Kommen wir doch mal zu GOMS. Was für verschiedene Typen hatten wir denn da?
Ich: Insbesondere hatten wir das normale (CMN)-GOMS und KLM-GOMS besprochen.
Jan: Was kam denn zuerst?
Ich: Ich nehm an zuerst das CMN-GOMS, und dann die vereinfachte Version des KLM.
Jan: In der Tat war es andersherum. Zuerst wurde das KLM-Verfahren entwickelt, und das dann zum vollen GOMS ausgearbeitet und veröffentlich. Aber ist auch nicht so wichtig.
Wie wendet man das denn an, und was sind denn die Unterschiede dazwischen?
Ich: Erkläre GOMS. Erkläre Baumstruktur/Linear. Nur fünf Operatoren bei KLM. Beschreibe die Regeln für Hinzufügen und Entfernen von M.
Jan: Und wann wendet man das an?
Ich: Damit kann man die Länge von Routineaufgaben berechnen.
Jan: Also angenommen, ich hab jetzt meinen Taktstock des virtuellen Orchesters. Kann ich da mit GOMS etwas erkennen?
Ich: Nein, denn die Zeiten der Operatoren bei Taktstock Input sind ja nicht bekannt.
Jan: Ok, angenommen, wir führen jetzt Experimente durch, und bekommen Werte für die Operatoren. Kann ich es dann nutzen?
Ich: Nein, denn beim virtuellen Orchester hat man ja first-time, one-time, short-time users. Da ist das normale GOMS nicht für geeignet. Ich glaube NGOMSL könnte das...
Jan: Genau! Wie ist das denn, wenn ich jetzt mit KLM eine Handy-Wahl modellieren möchte. Mach das doch vielleicht mal vor.
Ich: Naja, ich würde sagen, da man beim Handy ja mit dem Daumen suchen muss, ist das für jeden Tastendruck so eine Art Pointing-Funktion. Das dauert auf bestimmt länger als nur die 0,2 Sekunden vom K, aber nicht so viel wie die 1.1 Sekunden vom P. Weiterhin würde ich sagen, dass, wenn man eine Nummer wählt, die man gut kennt, nicht vor jeder Ziffer nachdenken muss, da die Nummer ein Block ist.
Jan: Ja, das mag sein. *Nach etwas hin und her haben wir uns dann auf etwas angepasste Zeiten geeinigt)*.
Ich: Modelliere die 10 Tastendrücke als (M + 10 * (P+K)). Dazu dann noch ein P + K für die Wahltaste. Da muss kein M vor, da es ein konstanter Terminationsausdruck ist, wie in den Regeln beschrieben.
Jan: Ja, ist ok so.
Ich: Ok, angenommen, wir möchten jetzt eine neue Eingabe-Funktion entwickeln, um T9 zu ersetzen. Etwas, dass man mit Neigung des Handys kontrollieren kann. Wie würdest du dafür

Prototypen bauen?

Ich: Naja, zuerst würde ich auf jeden Fall Storyboard erstellen und Benutzerbefragung durchführen. Insbesondere sollte da zwischen Gelegenheitsnutzen und häufigen Nutzern unterschieden werden.

Jan: Ja, das ist immer ein guter Anfang.

Ich: Weiterhin... Tja, ich denke, ein Papierprototyp bringt hier nicht so viel, da sich ja Navigationseigenschaften oder Dinge, die man dort sieht nicht wirklich ändern. Da müsste wohl wirklich ein Hardware-Prototyp her.

Jan: Ja, ein simpler Papier-Prototyp für die neuen Befehle würde wohl nicht schaden, aber um den Hardware-Prototyp kommt man wohl nicht herum. Jetzt hab ich also diesen Prototypen... Wie stell ich denn fest, ob das neue Verfahren besser ist als der alte?

Ich: Beschreibe quantitatives Testverfahren. Unterschied im Labor/beim User, Within-Users/Between-Users. (Hier hatte ich etwas missverstanden. Ich dachte, dass sich Lerneffekt auch auf Versuche der gleichen Art bezog. Es bezieht sich aber wohl nur auf Versuche unterschiedlicher Art, also wenn ich zuerst Eingabemethode A benutze, lerne ich dabei schon, wie Eingabemethode B funktioniert, und bin dann dort schneller. Der Fehler war aber wohl nicht so wild)

Jan: Was kann man denn sonst noch machen, wenn man gerade keine Benutzer parat hat?

Ich: Cognitive Walkthrough erklärt.

Jan: Ok, da muss man jetzt teure Experten anheuern. Gibt es nicht noch etwas, was preisgünstiger ist?

Ich: Literature Review.

Jan: Genau! Gerade im Handy-Bereich gibt es massig Paper, da lohnt es sich schon, einmal hereinzuschauen.
Wenn Leute jetzt Probleme mit der Eingabe haben, wie kann man das denn modellieren?

Ich: Beschreibe Norman's 7 Stages of Action

Jan: Genau! Betrachten wir mal die Evaluation-Seite. Nenn mir doch mal Beispiele, wie solche Probleme in den 3 Fällen aussehen könnten.

Ich: Perception: Es gibt keine Lampe, die eine erfolgreich Wahl anzeigt. Interpretation: Irgendwas blinkt, aber man weiß nicht, was es bedeutet. Comparison: (hier musste ich lange überlegen) Wenn mein Ziel es war, mich mit jemandem zu unterhalten, muss ich schauen, ob ich das erreicht habe... Vielleicht ist die Verbindung zu schlecht, um ein Gespräch zu führen, oder mir fehlt doch die direkte Kommunikation. Ist aber etwas abstrakt.

Jan: Ja, es ist schwierig zwischen Interpretation und Comparison zu unterscheiden. Norman hatte in seinem Buch ja das Beispiel mit der Lampe. Dort wird geprüft, ob die Lampe hell genug ist, um tatsächlich lesen zu können.
Wir hatten uns ja auch noch über die Geschichte von HCI unterhalten. Erzählt doch einfach mal kurz was über die wichtigsten Schritte.

Ich: Erzähle über x-dimensionale Interfaces, NLS, Apple II, Xerox Alto (Rasterbildschirm!)

DIS2:

Jan: Kommen wir mal zur technischen Seite... Vergleich doch mal Smalltalk mit Mac OSX

Ich: Erkläre Single-Process/Multi-Process, Virtualisierung.

Jan: Und sonst? Denk doch mal an das Grafikmodell...

Ich: Unterschied Raster- und Vektor-Operationen; Unterstützung durch GPU bei OSX. Core Image, Core Video.

Jan: Es gibt noch einen wichtigen Unterschied...

Ich: Nach einigen Hinweisen von Jan: Durch das Compositing (Schatten etc) ist es bei Mac OSX nicht mehr so leicht, bestimmte Widgets einem bestimmten Pixel-Haufen zuzuordnen. Der Schatten von einem Fenster zum Beispiel ist nicht Teil des Base-Window-System Window

Jan: Bei Smalltalk hatten wir ja über Morphic gesprochen... Was war denn daran so besonderes?

Ich: Erkläre Liveness, Directness.
Jan: Wen wir jetzt mal vom Desktop wegkommen wollen und an Post-Desktop UI denken... Was hatten wir da denn für Toolkits?
Ich: Erkläre iStuff.
Jan: Ok. Wenn wir das jetzt für das Prototyping unseres Handy mit der zusätzlichen Hardware nutzen wollten, wie sähe das denn aus?
Ich: Erkläre Hardware über Proxy an Event Heap, und von da weiter an Handy
Jan: Genau. Was für Probleme könnten den dabei entstehen?
Ich: Lag durch Netzwerk und Event Heap Verarbeitung. Responsiveness kann leiden.

CTHCI:

Jan: Genau. Kommen wir mal zu weiteren Modellierungsmöglichkeiten von Interfaces. Da hatten wir ja so was mit Matrizen gemacht. Ich hab da zu Hause so eine Lampe, die durch Drücken des Sockel steuerbar ist. (aus->dim->hell->superhell->aus). Modellier das doch mal.
Ich: Modelliere das.
Jan: Ja, das scheint richtig zu sein. Und wozu überhaupt der Aufwand?
Ich: Um durchführbare Aktionen genauer untersuchen zu können. Matrizen haben ausserdem ein paar schöne Eigenschaften, z.B. ist das hintereinanderausführen definiert, das invertieren ebenso.
Jan: Wie stellt man das denn fest und was bedeutet das?
Ich: Determinante ungleich 0. Das heisst, dass wenn das Gerät nach unserem Modell funktioniert, es einen Knopf haben könnte, der die entsprechende Aktion in jedem Fall rückgängig macht.
Jan: Ok. Wir hatten ja noch ein paar weitere Sachen, die man sich damit anschauen konnte...
Ich: Ja, zum Beispiel Anzahl der Schritte, die man braucht, um bei einem Fehler wieder zum vorherigen Zustand zurückzukommen (in diesem Fall 3 weitere Schritte). Und z.B. wie viel Operationen man durchführen muss, um zur am weitesten entfernten Funktion zu erlangen (ebenfalls 3 in diesem Fall).
Jan: Was für Vor- und Nachteile hat dieses Interface denn und was könnte man verbessern, ohne jetzt was an der Hardware zu ändern?
Ich: Naja, wenig Knöpfe ist auf jeden Fall immer gut, damit die Leute keine Angst bekommen. Gut ist auch, dass der ganze Sockel als An-Schalter funktioniert, dass man sich im Dunkeln nicht vertun kann. Verbesserungsmöglichkeiten fallen mir keine sinnvollen ein... Eine kontinuerliche Dim-Funktion durch kontinuierliches Drücken des Sockels wäre schlecht, weil dann eine Zeit-Komponente eingeführt wird.
Jan: Naja, was man besser machen könnte, wäre, die Reihenfolge der Dim-Schritte umzudrehen. Dann kann man zum Schlafengehen einfach einmal draufdrücken, und die Lampe ist aus, anstelle sich, wenn man müde ist, da noch mehrmals draufzudrücken.

HCI:DP

Jan: Kommen wir noch schnell zu Patterns. Wir hatten ja die Sprache von Jennifer Tidwell. Erzähl doch mal was über den Aufbau:
Ich: Erzähl was über die Form (explizit, auch mit Negativ-Beispielen), und die Struktur: Primärpatterns Content/Action (auch aufgezählt), davon weiterführende Patterns, mögliche Sub-Sprachen.
Jan: Genau. Wenn du das jetzt mal mit der Sprache vergleichst, die ich für Exhibits gemacht hab...
Ich: Deine ist ja von der Form her sehr an das Alexander-Format angepasst. Inhaltlich ist sie viel spezifischer auf ein Thema, während Tidwell ja Toolkit-Unabhängig, und sogar Interface-unabhängig ist.
Jan. Richtig. Wenn du jetzt mal die 9 Golden Rules, Tidwell, Borchers, und die Mac-HIG nach Spezifität sortieren müsstest, wie würdest du das machen? Also in welcher Reihenfolge würdest du die in den Stadien des Entwicklungsprozesses anwenden?
Ich: Zuerst 9 Golden Rules. Die kann man sehr schnell prüfen und sind sehr allgemein. Dann

Tidwell, da ja auch UI-unabhängig. Borchers und Mac-HIG würde ich wahrscheinlich nie zusammen benutzen, weil bei Exponaten das OS ja eh verschwindet, und die Mac-HIG also nicht so wichtig sind. Alle allgemeingültigen Sachen daraus sind bestimmt auch bei Tidwell oder in den Golden Rules enthalten.

Jan: Ja, das stimmt, das macht Sinn. Wenn du sie jetzt nach Zeitlosigkeit sortieren müsstest?

Ich: 9 Golden Rules gelten immer, das hat viel mit menschlicher Physiologie zu tun. Tidwell als nächstes, wegen Toolkit-Unabhängigkeit. Mac-HIG nicht so sehr, da es ja auf Aqua bezogen ist. Da kann sich ja durchaus in der nächsten Version was ändern. Borchers auch nicht so sehr, da Exponate sich ja technisch auch ständig weiterentwickeln.

Jan: Ja, das sehe ich ähnlich. So, ich denke dann sind wir durch, warte doch bitte kurz draussen.