

# Usability Engineering

Jens Willmer<sup>1</sup> und Thomas Tigges<sup>1</sup>

DHBW Stuttgart Campus Horb, Florianstraße 15, 72160 Horb am Neckar

**Zusammenfassung.** Usability Engineering bezeichnet ein Gebiet, in dem es darum geht eine optimale Mensch-Maschinen-Schnittstelle zu entwickeln. Bei einer solchen Schnittstelle liegt das Hauptaugenmerk auf der Benutzerfreundlichkeit. Die entwickelte Schnittstelle soll möglichst effizient für die vorgesehenen Arbeitsaufgaben gestaltet sein und zudem ein ansprechendes, gewohntes Aussehen haben, damit der Benutzer gerne damit arbeitet und sich schnell zu Recht findet. Des Weiteren muss die Schnittstelle vorgegebene Standards und Guidelines einhalten.

## 1 Usability Engineering

Usability Engineering beinhaltet eine Vielzahl von Aufgaben, welche sich über den kompletten Zeitraum eines Projektes hinziehen. Die Bedienbarkeit kann nicht als eine isolierte Aufgabe während des Projekts betrachtet werden, vielmehr ist die Bedienbarkeit ein wichtiger Teil der Entwicklung. Diese verläuft parallel zu der Programmierung des Produkts und hat direkte Auswirkungen auf die Reputation einer Firma, denn die Bedienbarkeit des zu verkaufenden Produkts wirkt sich auf die Produktivität und Zufriedenheit der Anwender aus.

Um die Entwicklung der Oberfläche und die damit verbundene Bedienbarkeit so effizient wie möglich zu gestalten wird das Usability Engineering in verschiedene Prozessphasen eingeteilt, welche in den kommenden Abschnitten näher betrachtet werden. [3,4]

### 1.1 Kenne den Anwender

Der erste Schritt im Prozess des Usability Engineerings ist es die späteren Anwender kennen zu lernen und diese zu definieren. Es ist hierbei darauf zu achten, dass die späteren Anwender nicht nur aus dem Kreis der Personen stammen, welche später das Produkt benutzen werden, sondern dass ebenso die IT-Abteilung des Unternehmens sich mit Teilen des Programms auskennen muss, um es in ihre Infrastruktur zu integrieren und bei Problemen über mögliche Ursachen Bescheid weiß.

Die Benutzer der späteren Anwendung zu kennen ist die Grundvoraussetzung für ein effizientes Design und oft sehr schwer zu erreichen. Die Entwickler der Software werden üblicherweise von dem Kunden fern gehalten um ihre Produktivität nicht zu beeinflussen und ihren Fokus auf die Entwicklung zu richten, wodurch ihnen jedoch der Bezug zum Kunden fehlt. Des Weiteren stellt der Auftraggeber meistens wenig Zeit zur Verfügung um die späteren Benutzer zu

beobachten und von ihrem bisherigen Arbeitsverhalten zu lernen. Entweder weil die Anwender gut bezahlte Fachkräfte sind, welche viel kosten oder aber auch weil die Anwender es nicht mögen, dass man ihnen über die Schulter schaut.

Es ist wichtig die Komplexität der Oberfläche an der Erfahrung der Anwender auszurichten. Wird die Anwendung zum Beispiel für Kinder entwickelt sind Bilder für die Navigation besser geeignet als Text, weil davon ausgegangen werden muss, dass die Kinder noch nicht gut lesen können und sie deswegen erst trainiert werden müssen, um die Navigation zu verstehen. Desto schneller ein Anwender die Funktionsweise der Anwendung versteht und sich in der Navigation zurechtfindet, desto schneller kann er produktiv arbeiten, wodurch Zeit und teure Schulungen eingespart werden können.

Bei der Betrachtung von Anwendern sollte besonderes Augenmerk auf die effektivsten Anwender gelegt werden. Die Informationen über die Vorgehensweise dieser Anwender zur Bewältigung ihrer Arbeitsaufgaben kann dazu genutzt werden das zu entwickelnde System dahingehend zu optimieren. Dafür ist es besonders nützlich zu betrachten bei welchen Aufgaben am meisten Zeit benötigt wird, welche nicht funktionieren und welche in der Bedienung unkomfortabel sind, um dies in der Entwicklung zu berücksichtigen und besser zu machen.

Bei der Betrachtung des Anwenders ist es wichtig zu verstehen, dass dieser mit dem neu entwickelten System in der Zukunft eine Evolution vom Anfänger zum professionellen Anwender durchmachen wird. In der Phase des Anfängers wird er dankbar sein über ausführliche Fragekataloge, die ihn durch die Anwendung begleiten und ihm jeden Schritt erklären oder ihm für den jeweiligen Systemzustand mehrere Optionen vorschlagen wie er fortzufahren hat. Ist der Anwender dann aber soweit, dass er keine geführte Bedienung mehr benötigt und sich selbstsicher durch das Programm bewegen kann, wird ihn die ausführliche Anleitung zu jedem Programmschritt nerven. Für diese Personengruppe sollte in Betracht gezogen werden, dass die Hilfe abschaltbar ist oder der Anwender sich auch durch Tastenbelegungen im Programm fortbewegen kann.

Ein weiterer Punkt, der bei der Entwicklung beachtet werden sollte, ist, dass das neue System die Arbeitsweise des Anwenders verändert und dieser wiederum neuen Anwendungsmöglichkeiten für das Programm finden wird. Zum Beispiel kann es vorkommen, dass Kalkulationstabellen von den Anwendern auch zur Datenablage benutzt werden.

Um auf diese Anwendungsveränderungen seitens der Nutzer reagieren zu können ist ein flexibles Programmdesign erforderlich.

## **1.2 Zielsetzung für die Bedienbarkeit**

Da die Bedienbarkeit eines Programms ständig verbessert werden kann und ihre Gestaltung auch stark von den Anwendern und dem Einsatzzweck abhängt, ist es erforderlich die Qualität der Bedienbarkeit fest zu legen. Besteht im Betrieb des Auftraggebers eine hohe Fluktuation an Arbeitskräften, die das Programm bedienen, ist es erforderlich viel Wert auf eine kurze Einarbeitungsphase zu legen und dafür in anderen Bereichen Einschnitte hin zu nehmen. Abbildung 1 zeigt

hierzu eine mögliche Einteilungsskala, mit welcher die gesetzten Ziele visualisiert werden.

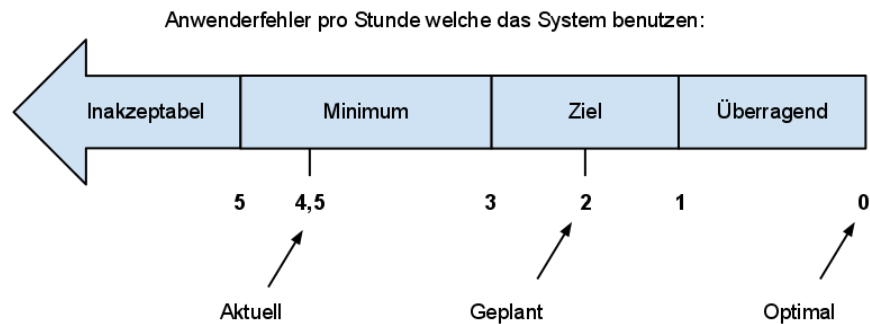


Abb. 1.

Ein weiterer Punkt, der die Zielsetzung maßgeblich beeinflusst, ist das Budget, das zur Verfügung steht. Hierbei sind auch finanzielle Vorteile zu berücksichtigen, die nicht direkt in Verbindung mit dem Verkaufserlös des Produkts in Zusammenhang stehen. Dazu gehört zum Beispiel die Berechnung der Effizienz und die dadurch beeinflussten Kosten.

Um diese zu berechnen wird eine Auflistung erstellt, die die Anzahl an Mitarbeitern enthält, welche das Produkt benutzen, die des Weiteren auch festhält wie lange diese Mitarbeiter das Produkt im Jahr benutzen (365 Tage x 8 Stunden) und wie viele Jahre das Produkt voraussichtlich genutzt wird. Auf Basis dieser Auflistung ist dann auch u.a. eine Berechnung möglich wodurch sich kalkulieren lässt welche Kosten eingespart werden können falls die Einarbeitungszeit um einen Tag kürzer ausfällt oder die durchschnittliche Produktivität um einen bestimmten Prozentsatz fällt beziehungsweise steigt.

### 1.3 Designphase

Eine bewährte Methode den Designprozess einer Anwendung zu starten ist es die Designer einzeln an einem möglichen Konzept arbeiten zu lassen, und möglichst ohne dass die Designer sich gegenseitig absprechen. Am Ende dieser ersten Designphase werden die unterschiedlichen Konzepte zusammengeführt und die besten Ansätze der einzelnen Designer in das weiter zu entwickelnde Konzept übernommen (Abbildung 2).

Die besten Ergebnisse entstehen dabei, wenn die verschiedenen Entwürfe möglichst unterschiedliche Lösungsansätze verfolgen. Eine Variante um dies zu bewerkstelligen nennt sich „diversified parallel design“. Hierbei bekommen die

Designer verschiedene Problemstellungen zu einer bestimmten Funktion. Zum Beispiel soll sich ein Designer damit befassen wie eine bestimmte Funktion am besten für neue Anwender umgesetzt werden kann und ein andere Designer soll sich überlegen wie diese Funktion am besten für geschultes Personal gestaltet werden kann.

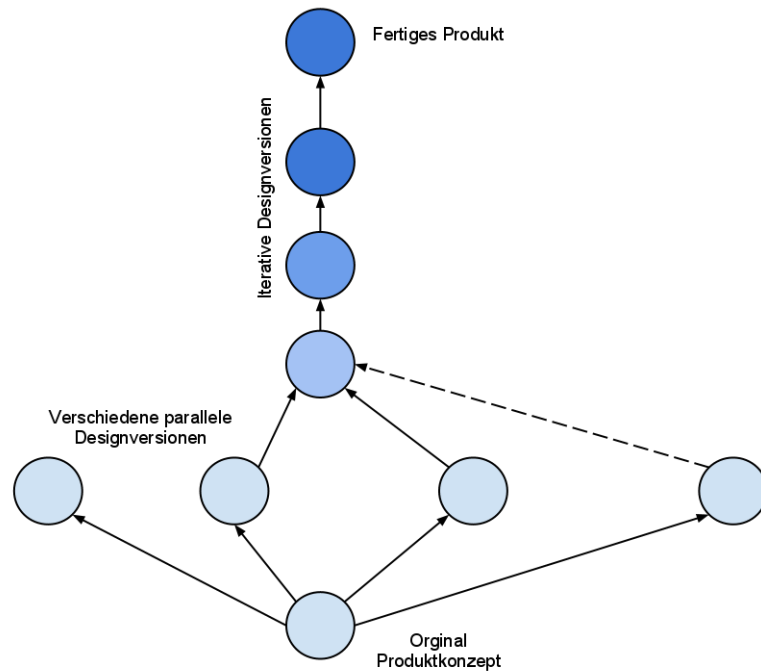


Abb. 2.

Während der Designphase ist es wichtig, dass die Designer Zugriff auf eine Auswahl von Personen haben, welche später mit dem Produkt arbeiten sollen. Dies ist deshalb so wichtig, weil nur die späteren Anwender genau beschreiben können welchen Problemen sie bei ihrer derzeitigen Arbeit gegenüberstehen. Des Weiteren ist darauf zu achten, dass nicht immer die gleichen Anwender zum Studium herangezogen werden, da diese sonst mit der Zeit nicht mehr als repräsentativ betrachtet werden können. Das Arbeiten als Testanwender verändert mit der Zeit das Verständnis über die Funktionsweise der Anwendung, was dann die Testergebnisse verfälscht.

#### 1.4 Guidelines und Standards

Um den Aufwand bei den anstehenden Tests zu minimieren ist es empfehlenswert Standards für das Aussehen der Anwendung fest zu legen. Damit müssen

Anleitungen oder Trainingsvideos für die Tester nicht jedes Mal neu erstellt werden, sondern können ergänzt werden. Zudem erleichtert es das Wiedererkennen und hat den Effekt, dass sich die Tester schneller in ein gewohntes Umfeld einarbeiten können. Standards beschreiben Methoden, welche sich gegenüber anderen Vorgehensweisen als die populärsten und anerkanntesten herausgestellt haben.

Dem gegenüber stehen die Guidelines und Stylesheets. Sie bauen auf den Standards auf und erweitern diese durch genauere Beschreibungen. Ein Standard könnte zum Beispiel sein, dass es in einem Programm immer einen Weg zurück zum Startzustand geben soll. Eine Guideline die darauf aufbaut würde zum Beispiel festlegen, dass dieser Weg immer oben links in Form eines Pfeils vorhanden sein muss. Guidelines und Stylesheets entstehen meistens aus den gesammelten Erfahrungen von vorangegangenen Projekten.

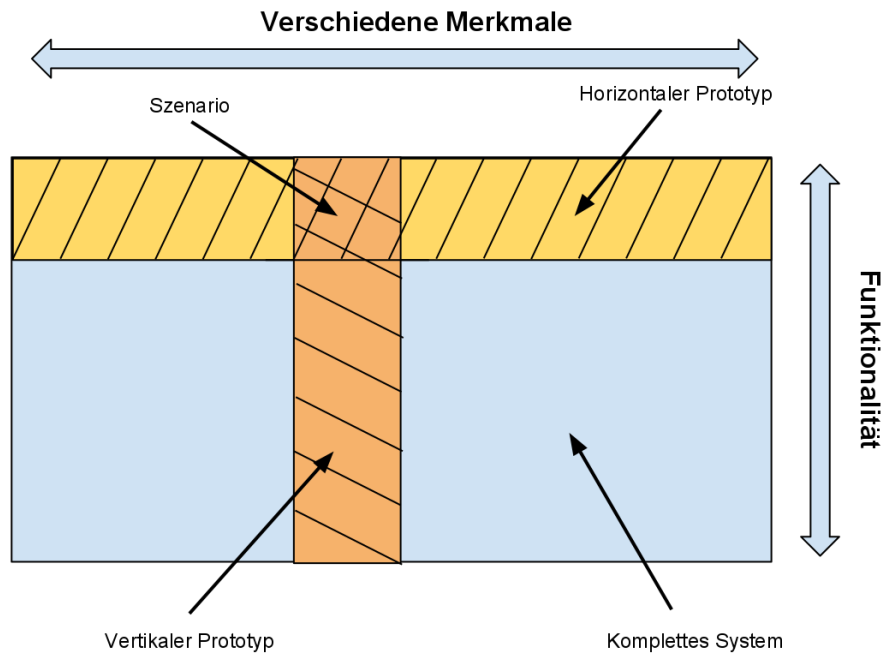


Abb. 3.

### 1.5 Prototyp

Um die Usability testen zu können bevor das Produkt seinen endgültigen Zustand erreicht hat sind Prototypen das Mittel der Wahl. Mit ihnen lassen sich Funktionen schon am angestrebten Design testen und der Entwickler muss keine

Anpassungen mehr am Design vornehmen um neue Funktionen zu implementieren. Ein Prototyp lässt sich hierbei schnell aufbauen.

Dies wird erreicht, indem der Prototyp nur wenig Funktionalität bietet und diese ohne Rücksicht auf Geschwindigkeitseinbußen oder Rechenaufwand implementiert werden. Des Weiteren gibt es spezielle Software, welche dafür entwickelt wurde Prototypen sehr schnell aufzubauen, indem häufig benötigte Elemente schon als Bausteine vorgehalten werden. Ebenso ist es möglich für unterschiedliche Anwendungsbereiche unterschiedliche Prototypen zu erstellen, wodurch keine Rücksicht auf das Zusammenspiel untereinander genommen werden muss.

Es gibt hierbei drei verschiedene Herangehensweisen bei der Konzeptionierung eines Prototyps (Abbildung: 3). Bei der Ersten wird auf die Funktionalität verzichtet, dies nennt man den horizontalen Prototyp. Er beinhaltet das komplette User-Interface, verzichtet aber auf die darunterliegenden Funktionen, wodurch keine wirkliche Arbeit simuliert werden kann.

Bei einem vertikalen Prototyp ist dies genau umgekehrt. Bei ihm wird auf ein Teil des Interfaces verzichtet, um die dadurch gesparte Zeit in die Funktionalität zu investieren. Damit ist es diesem Prototyp möglich für einen limitierten Bereich die angestrebte Funktionalität ab zu bilden.

Die dritte Variante besteht darin die Funktionalität und die Oberflächennavigation soweit einzuschränken, dass der Testanwender lediglich einen vorher definierten Testablauf abarbeiten kann, da alles andere noch nicht implementiert wurde.

## 1.6 Oberflächen Evaluation

Um Fehler in der Oberfläche zu priorisieren und festzustellen welche Auswirkungen diese auf die Funktionalität des Programms haben, sollten diese von mindestens vier unabhängigen Personen bewertet werden. Eine einfache Art der Bewertung besteht darin eine Tabelle auf zu bauen, über welche das Problem bewertet wird (Abbildung4).

## 1.7 Iteratives Design

Um das Design weiter zu verbessern kann es hilfreich sein das Logfile eines Tests zu untersuchen, um festzustellen bei welchen Aufgaben die Probanden am meisten Zeit benötigt haben. Mit dieser Methode ist es möglich Designprobleme zu entdecken, die den Anwendern nicht bewusst auffallen. Ein hoher Zeitaufwand für eine Aufgabe deutet meist auf einen nicht weit genug automatisierten Arbeitsablauf oder Probleme bei der Steuerung hin. Um weitere solche Probleme zu entdecken kann es von Vorteil sein die Logfiles der abzulösenden Anwendung, sofern vorhanden, zu sichten und mit den eigenen Daten zu vergleichen. So lassen sich Arbeitsabläufe finden, die im früheren Design effizienter gestaltet wurden und daraus lassen sich Verbesserungen ableiten.

Werden Probleme gefunden geht es an erster Stelle darum wie viel Anwender davon betroffen sind und wie viele Anwender die Lösung des Problems beeinflussen oder sogar stören wird. Es gilt daraufhin mehrere Lösungen zu testen und zuletzt die effektivste davon aus zu wählen.

		Wieviel Anwender werden auf das selbe Problem stoßen?	
		Wenige	Viele
Behinderung für den Anwender der das Problem entdeckt hat	Klein	Wenig Gewichtung	Mittlere Gewichtung
	Groß	Mittlere Gewichtung	Hohe Gewichtung

Abb. 4.

## 1.8 Feedback

Letztlich ist es für alle Mitarbeiter wichtig zu verstehen wieso etwas genau auf diese Art und Weise umgesetzt wurde, damit zukünftig eingebrachte Änderungen nicht den Sinn dahinter zerstören.

## 2 Usability Testing

Die grundlegendste Testmethode, welche durch keine andere ersetzbar ist, ist der Anwendertest. Sie liefert direkte Informationen darüber wie die Anwender das System nutzen und wo ihre Probleme liegen. Wie bei jeder Testmethode ist allerdings auf die Zuverlässigkeit sowie die Aussagekraft der Daten zu achten.

Zuverlässigkeit bedeutet hierbei, dass mit denselben Ergebnissen gerechnet werden kann wenn der Test wiederholt wird. Dies ist ein großes Problem, denn die Testanwender unterscheiden sich oft sehr voneinander. Es gibt Anwender, die in den Tests 10-mal schneller sind als der langsamste Benutzer. Zudem sind die besten 25 % der Nutzer durchschnittlich doppelt so schnell wie die langsamsten 25 %. Deshalb ist es schwer, zuverlässige Resultate aus kleinen Gruppen zu gewinnen.

Bei der Aussagekraft wird die Frage gestellt ob das Resultat des Tests tatsächlich das zu testende Problem betrifft. Oft entstehen diese Probleme wenn die falschen Nutzer als Testanwender herangezogen werden, diesen die falschen Aufgaben

zugewiesen werden, zu wenige Zeit zur Verfügung stand oder nicht auf den sozialen Hintergrund geachtet wurde. Zum Beispiel macht es einen Unterschied ob ein System, das für Banker entwickelt wird, mit Studenten getestet wird oder tatsächlich mit ausgebildeten Fachkräften aus dem Banksektor. Eine noch größere Abweichung ist zu erwarten wenn die Tester aus Studiengängen stammen, die mit der Banklehre garnichts zu tun haben.

## 2.1 Testplan

Bevor ein Testplan aufgestellt wurde, welcher die Ziele des Tests definiert, sollten keine Tests vorgenommen werden. Ein Testplan sollte folgende Fragestellungen beantworten:

- Ziel des Tests?
- Wo und wann findet der Test statt?
- Wie lange dauert jede Testeinheit?
- Was für Zubehör wird benötigt?
- Welche Software muss für den Test vorbereitet werden?
- Wie sollte der Zustand des Systems aussehen bevor der Test startet?
- Wie schnell sollte das Netzwerk, auf zum Beispiel Datenbankabfragen, reagieren?
- Was für Personen werden als Tester benötigt?
- Wer überwacht den Test?
- Wie viele Testanwender werden benötigt?
- Welche Tests sollen die Anwender testen?
- Welche Kriterien werden angelegt um zu bestimmen ob eine Aufgabe erfolgreich war?
- Werden Hilfsmittel zur Verfügung gestellt und welche?
- In wie weit darf die ausrichtende Person Hilfestellungen geben?
- Welche Daten werden gesammelt und wie werden diese später analysiert?
- Wie sehen die Bedingungen aus um den Test als Erfolg zu werten?

## 2.2 Budget

Außerdem sollte der Testplan ein Budget für die anstehenden Tests enthalten. Einige Dinge werden bar bezahlt werden müssen. Andere Kosten werden verursacht bei der Benutzung der firmeneigenen Güter. In wie weit diese mit dem bestehenden Budget verrechnet werden hängt von der jeweiligen Firma ab. Typische Kosten die anfallen werden sind:

- Spezialisten zur Planung, Durchführung und Analyse der Tests
- Assistenten welche Terminabsprachen festlegen oder die Testresultate zusammentragen
- Softwareentwickler welche die Quellcode für die Tests anpassen
- Die Arbeitszeit der Testanwender
- Computer die während dem Test und der Analyse benötigt werden



- Die Räumlichkeiten
- Videomaterial und andere Verbrauchsmaterialien

Das Testbudget sollte hierbei in variable und fixe Kosten aufgeteilt werden. Fixkosten sind hierbei die anfallenden Kosten für die Planung und Installation der Testumgebung. Diese Kosten verändern sich nicht mit der Anzahl von eingeladenen Testern. Variable Kosten hingegen bezeichnen die Kosten welche für jeden einzelnen Anwender anfallen.

Wurde das Budget in diese zwei Kategorien aufgeteilt kann berechnet werden wie viel für Fix- und Variable-Kosten ausgegeben werden kann, um darauf aufbauend die Anzahl von Testanwendern zu bestimmen.

### 2.3 Pilottest

Pilottests finden vor den eigentlichen Tests statt. Es werden hierfür Testanwender genutzt, welche gerade im näheren Umfeld verfügbar sind. Sie dienen dazu um herauszufinden, ob es eine Diskrepanz zwischen der veranschlagten Zeit für eine Testaufgabe und der eigentlich benötigten Zeit für die Abarbeitung der Aufgabe gibt. Des Weiteren können hier Mängel in der Testumgebung entdeckt werden, welche vor dem eigentlichen Test noch behoben werden können.

### 2.4 Testanwender

Am repräsentativsten sind Anwender welche später auch mit dem zu testenden System arbeiten. Stehen zu wenige Testanwender zur Verfügung kann auch im Umfeld der pensionierten Mitarbeiter gesucht werden. Sie besitzen andere Charakteristiken, wodurch andere Designfehler aufgedeckt werden können. Bei einer kleinen Gruppe sollte darauf geachtet werden, dass möglichst nur durchschnittliche Testanwender für den Test ausgewählt werden um die Ergebnisse nicht zu verfälschen. Wird mit einer großen Personengruppe gearbeitet ist dieser Faktor zu vernachlässigen, da große Gruppen in der Regel sowohl sehr schnelle als auch sehr langsame Anwender enthalten.

Sollen allerdings aus dieser Gruppe nicht alle alles testen, muss diese Gruppe anhand von Merkmalen in Untergruppen eingeteilt werden, um homogene Kleingruppen zu erhalten. Übliche Merkmale zur Einteilung in Gruppen sind hierbei das Alter, die Abteilung, das Geschlecht sowie die Fertigkeiten der Person.

Ein weiteres Problem, das bei Testanwendern auftritt, ist dass diese von Testaufgabe zu Testaufgabe das System besser kennenlernen und nach einigen Aufgaben nichtmehr als Anfänger eingestuft werden können. Um diesen Faktor zu minimieren sollten die einzelnen Gruppen die angesetzten Tests in unterschiedlicher Reihenfolge durchführen.

Wenn sich die Testanwender unter Druck gesetzt fühlen werden sich die Fehler häufen und das Lernverhalten wird signifikant abnehmen. Um diesem Verhalten vorzubeugen ist es empfehlenswert den Testern klar zu machen, dass nicht sie auf ihre Leistungsfähigkeit geprüft werden, sondern es um die Erfassung von

Daten zur Verbesserung des angestrebten Produktes geht und keine Daten erfasst werden, welche später den Anwender identifizieren könnten. Des Weiteren sollte die erste Aufgabe, die ein Testanwender bekommt, eine Aufgabe sein, bei der der Erfolg garantiert ist, um den Tester zu motivieren. Die letzte Aufgabe die der Benutzer zu tätigen hat, sollte hingegen so anspruchsvoll sein, dass der Nutzer danach das Gefühl hat etwas erreicht zu haben und mit einem positiven Gefühl aus den Tests geht, um auch für kommende Tests mit einer positiven Grundhaltung zur Verfügung zu stehen.

## 2.5 Testphasen

Ein Usability Test besteht typischerweise aus folgenden Phasen:

**Vorbereitung** In der Vorbereitung geht es darum sicherzustellen, dass die Räumlichkeiten vorbereitet wurden, die Systeme in einem einheitlichen, im Testplan beschriebenen Zustand sind und alle Materialien für den Test vorhanden sind, sowie Systemdaten aus früheren Tests wieder auf ihren Ursprung zurückgesetzt wurden.

**Einführung** In der Einführung der Testanwender sollten diese über die Absichten der Tests aufgeklärt werden. Ihnen sollte eine kurze Einführung in das System gegeben werden sowie eine Einführung über das Vorgehen bei den einzelnen Tests. Des Weiteren sollte darauf hingewiesen werden, dass die aus dem Test bezogenen Daten keine Identifikation der Testanwender möglich machen und die Anwender freiwillig an dem Test teilnehmen, also auch jederzeit abbrechen können.

**Test** Während des Tests sollte die Aufsichtsperson davon Abstand nehmen mit den Testanwendern zu interagieren und versuchen keine Meinung darüber ab zu geben ob ein Anwender die Tests gut bearbeitet oder nicht. Wenn die Aufsichtsperson jedoch feststellt, dass ein Testanwender sichtbar enttäuscht ist und nicht weiter kommt, kann sie eingreifen damit die weiteren Tests davon nicht beeinflusst werden.

**Abschlussbesprechung** Nach den Tests werden die Anwender gebeten ihre Meinung über das getestete System ab zu geben und ihre persönlichen Erfahrungen sowie die ihnen aufgefallenen Probleme und Verbesserungsmöglichkeiten zu beschreiben. Nachdem die Anwender das Testlabor verlassen haben ist es wichtig, dass alle erhobenen Daten eingesammelt werden und mit einer Anwendernummer versehen werden um bei der späteren Analyse der Daten diese zuordnen zu können.

## 2.6 Weitere Test-Methoden

Es gibt verschiedene Methoden mit welchen Anwendertests durchgeführt werden können. Jede besitzt hierbei unterschiedliche Vor- und Nachteile. Vier dieser Methoden werden in den kommenden Abschnitten nun näher betrachtet.

**Laut denken** Bei der 'Laut denken'-Methode bekommt der Anwender die Aufgabe - während er die Tests bearbeitet - seine Gedanken laut mit zu teilen.

Dies hilft Missverständnisse in der Anleitung oder der Bedienung auf zu decken. Es ist hierbei allerdings darauf zu achten, dass die Aussagen des Anwenders kritisch bewertet werden bevor sie vermerkt werden. Eine Aussage des Anwenders über die Anordnung eines Feldes, das er übersehen hat, muss nicht unbedingt bedeuten, dass es Sinn macht, das Feld neu zu platzieren. Es muss immer darauf geachtet werden ob dies auch im Gesamtkonzept einen Sinn ergibt. Viel stärker zu bewerten ist die Vorgehensweise des Anwenders und das Verstehen weshalb er etwas so macht und nicht anders.

Bei dieser Methode ist darauf zu achten, dass sie bei der Zeitmessung andere Ergebnisse liefert als alternative Methoden. Der Benutzer benötigt für das laute Denken mehr Konzentration, was ihn entweder ausbremst oder dazu führt, dass der Anwender ein verbessertes Lernverhalten an den Tag legt, wodurch er an Geschwindigkeit zunimmt.

**Konstruktive Interaktion** Bei dieser Methode arbeiten zwei Anwender zusammen an einem Testfall. Der Vorteil hierbei ist, dass die Kommunikation zwischen den Testpersonen bessere Erkenntnisse hervorbringt. Dieses Vorgehen wird bei Kindern bevorzugt, da es einfacher als die ?laut denken?-Methode zu erklären ist.

Es kann hierbei allerdings vorkommen, dass die Probanden verschiedene Strategien der Problemlösung verfolgen, wodurch sie sich gegenseitig ausbremsen. Zudem wird eine große Anzahl an Testpersonen benötigt, was nur gelingt, wenn Testpersonen in großer Anzahl zur Verfügung stehen und relativ günstig sind - bevorzugt Kinder.

**Retrospektives testen** Beim retrospektivem Testen wird den Benutzern ein Video vom vorhergehenden Test gezeigt, welches diese kommentieren sollen. Der Vorteil hierbei ist, dass das Video pausiert werden kann und dadurch Zeit für Zwischenfragen zur Verfügung steht. Des Weiteren besteht kein Zeitdruck und die Anwender können sich für ihre Antworten Zeit lassen, wodurch qualifiziertere Aussagen getroffen werden können.

Diese Methode eignet sich gut wenn es nur sehr wenige Testanwender gibt, da mit dieser Methode mehr Informationen aus den einzelnen Probanden zu gewinnen sind. Der Nachteil ist, dass für die Befragung der Testanwender weitere Zeit benötigt wird und die Zeit der Anwender meist sehr teuer ist.

**Coaching Methode** Bei dieser Methode darf der Anwender einen Experten über das System befragen, wodurch er auf einfache Weise Fachwissen erwirbt und einen Einblick in das Modell des Experten bekommt. Hierdurch wird die Testperson schnell zu einem geschulten Anwender, wodurch er für kompliziertere Tests zur Verfügung steht. Diese Methode wird sehr häufig bei sehr teuren Testbenutzern angewendet, da sie den eigentlichen Test beschleunigt. Zudem wird diese Methode häufig in Japan angewendet, da andere Methoden oft schwer zu benutzen sind. Grund dafür ist, dass es in der japanischen Kultur Normen gibt, welche es den Menschen erschweren einen Widerspruch ein zu legen.

**Zusammenfassung.** Der folgende Abschnitt beschäftigt sich mit den physischen Grundlagen des Menschen und daraus abzuleitenden Anforderungen an Benutzeroberflächen. Beginnend mit dem Gedächtnis, als Speicher des aktuell Wahrgenommenen, führt der Text weiter zu Themen die sich mit der Erfahrung des Anwenders beschäftigen. Nach der Behandlung von Fehlerarten und deren Vermeidung werden anschließend grundlegende Themen der Gestaltung von Benutzerschnittstellen besprochen.

### 3 Gedächtnis

Das Gedächtnis unterteilt sich in mehrere Bereiche, die im Folgenden behandelt werden sollen. Dabei handelt es sich um das sensorische Kurzzeitgedächtnis, das Kurzzeitgedächtnis und das Langzeitgedächtnis. Von ihnen werden verschiedene Aufgaben übernommen: Das Verarbeiten der Sinneseindrücke, das bewusste Denken und das Speichern von Informationen. [1]

#### 3.1 Sensorisches Kurzzeitgedächtnis

Die von den verschiedenen Sinnen des Menschen gelieferten Signale werden im sensorischen Kurzzeitgedächtnis, nach dem Prinzip First In First Out, gespeichert. Seine Kapazität umfasst ungefähr 1 Sekunde. Bei der schnellen Abfolge von Reizen, zum Beispiel der Vergleich zweier fast identischer Ausdrücke, können in den sich ändernden Daten schnell Unterschiede gefunden werden. Nachvollziehen lässt sich dies auch durch einen schnellen Blick auf eine analoge Uhr. Wird der Blick sofort wieder abgewandt, kann die Uhrzeit aus dem sensorischen Gedächtnis trotzdem noch abgelesen werden. [1]

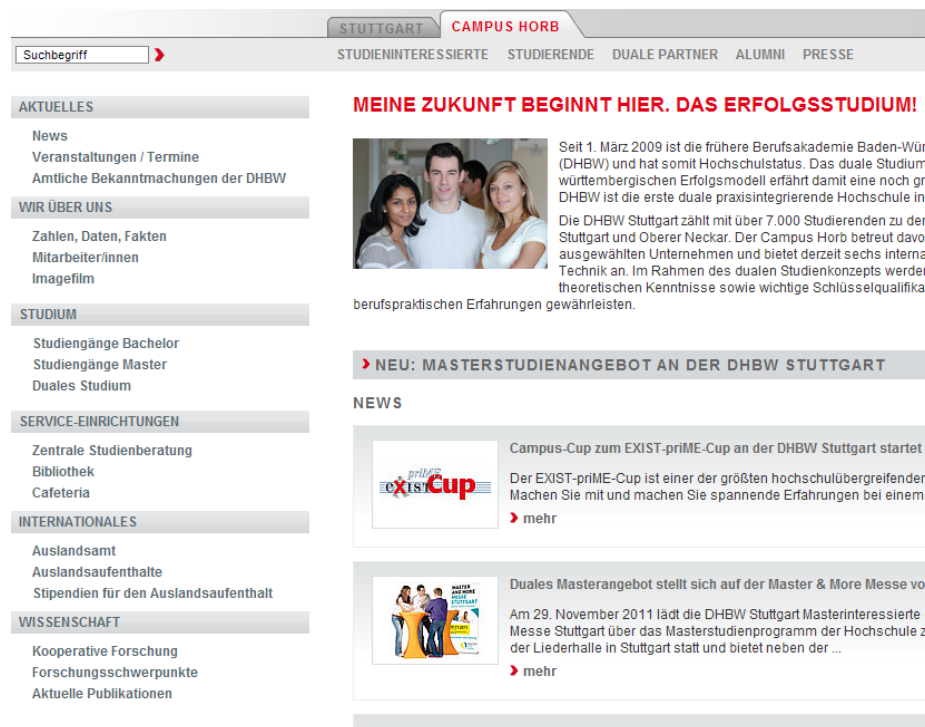
#### 3.2 Kurzzeitgedächtnis

Quasi der Arbeitsspeicher unseres Gedächtnisses ist unser Kurzzeitgedächtnis. Alles mit dem wir uns zu einem bestimmten Zeitpunkt bewusst beschäftigen wird im Kurzzeitgedächtnis gespeichert. Es enthält den Kontext in dem wir uns in jedem Augenblick befinden.

Der aktuelle Kontext grenzt das Kurzzeitgedächtnis zum sensorischen Kurzzeitgedächtnis ab. Das sensorische Kurzzeitgedächtnis verarbeitet Reize auch unterbewusst. Wird zum Beispiel während eines Gespräches von einem Dritten der eigene Name gerufen, wird dies dort wahrgenommen und tritt von dort in das Bewusstsein, dem Kurzzeitgedächtnis.

Vom aktuellen Kontext wird unsere Wahrnehmung stark bestimmt. Bei neu hinzukommenden Signalen wird zunächst versucht diese im Rahmen des aktuellen Kontextes einzuordnen. Sprunghafte Wechsel sind nicht vorgesehen und können zu Verwirrung führen. Die Frage nach „Was bewirkt das Drücken einer Taste?“ wird beim Gespräch über ein Klavier anders beantwortet als beim Gespräch über eine Fernbedienung oder einen Fahrstuhl.

Beim Entwurf einer Benutzerschnittstelle ist es daher sehr wichtig den aktuellen Kontext des Benutzers zu beachten. Zum Einen kann dieser Effekt dazu genutzt werden den Anwender in eine gewünschte Denkrichtung zu lenken. Zum Anderen muss aber auch jederzeit der aktuelle Kontext des Anwenders in Betracht gezogen werden. Die angebotenen Funktionen und Rückmeldungen müssen sich immer am aktuellen Kontext des Anwenders orientieren. Andernfalls können diese zu Verwirrung und schlimmstenfalls zu Frust führen.



**Abb. 5.** Im oberen Bereich gliedert sich die Navigation in fünf Punkte. Auf der linken Seite müssen wesentlich mehr Navigationselemente gleichzeitig dargestellt werden. Durch die Gruppierung wird auch dort die  $7 \pm 2$  Regel eingehalten.

Die Kapazität des Kurzzeitgedächtnisses ist begrenzt. Da dieses immer den aktuellen Kontext des Anwenders bestimmt, muss darauf geachtet werden dieses nicht unnötig zu belasten. Für die Kapazität des Kurzzeitgedächtnisses wird angenommen, dass es 7 Informationseinheiten  $\pm 2$  aufnehmen kann. In der Psychologie werden diese Informationseinheiten auch als Chunks bezeichnet. Je nach Kontext können solche Chunks zum Beispiel einzelne Icons, Namen oder Begriffe darstellen.

Wichtig ist diese Begrenzung der Kapazität für den Entwurf von Navigationselementen, für dessen Gestaltung ein Beispiel in Abbildung 5 zu sehen ist.

Mehr als sieben Menüpunkte sollte eine Navigation nicht umfassen, besser sogar nur fünf. Andernfalls droht die Gefahr, dass der Anwender die Navigation nicht mehr auf einen Blick erfassen kann. Müssen mehr als 7 Elemente dargestellt werden, bietet es sich an diese in kleineren Gruppen zusammenzufassen. Hierdurch wird die Menge an Informationen pro Ebene begrenzt. [1]

### 3.3 Langzeitgedächtnis

Im Langzeitgedächtnis wird alles gespeichert das wir während unseres Lebens gelernt haben. Es speichert folgende Informationen:

**Fakten, Daten und Konzepte** die explizit formuliert sind, wie einzelne Wörter, Vokabeln, Gerüche, Gefühle und Regeln wie die Grammatik.

**Bilder und Vergleiche** mit denen ein Gegenstand oder eine ganze Situation erfasst und erkannt werden können.

**Zusammenhänge und Schlussfolgerungen** die aus der Kombination von Beobachtetem, Gelerntem und Erfahrenem gezogen werden.

**Abläufe** die eine Reihe von komplexen Handlungen umfassen und die immer das gleiche Ziel erreichen. Einmal gestartet, müssen diese Abläufe nicht mehr bewusst gesteuert werden. Nur wenn der Ablauf nicht dem Gewohnten entspricht, schaltet sich wieder das bewusste Überlegen ein.

Aus all diesen Daten wird im Langzeitgedächtnis ein Wissen gebildet. Im Bereich der Mensch-Computer-Interaktion tritt dieses Wissen unter anderem als Aufgabenwissen und Bedienungswissen auf. Das Aufgabenwissen bezieht sich auf die Art wie eine Aufgabe erledigt werden kann. Das Bedienungswissen bezieht sich auf die in einer Oberfläche verwendeten Ausdrücke, den Aufbau von Menüs oder das Aussehen und die Bedeutung von Icons.

Dieses Vorwissen und die Erwartungen müssen beim Entwurf einer Benutzeroberfläche beachtet werden. Zum Einen muss der Anwender Funktionen an der von ihm erwarteten Stelle finden. Zum Anderen ist beim Gestalten der Oberfläche auf Konsistenz zu achten. Windows gewohnte Anwender werden die Einstellungen unter dem Menüpunkt *Extras* suchen, während Anwender von Linux diesen eher unter dem Menüpunkt *Datei* erwarten; und wäre eine Anordnung unter dem Menüpunkt *Bearbeiten* nicht die intuitivste?

Werden die Erfahrungen und Erwartungen des Anwenders berücksichtigt, findet dieser sich in einer neuen Benutzeroberfläche schneller zurecht und kann Aufgaben effizienter erfüllen. [1]

## 4 Assoziationen

Beim Ausführen und Lernen neuer Handlungen versucht das Gehirn diese mit bereits Bekanntem zu verknüpfen. Können solche Assoziationen gebildet werden, ist es wesentlich leichter etwas Neues zu erlernen und zu verstehen. Anstatt sich zunächst ein neues mentales Modell einer Sache aneignen zu müssen, kann auf

ein bereits vorhandenes zurückgegriffen werden. Tastenkombinationen fangen beispielsweise, insbesondere im englischsprachigen Raum, oft mit dem Wort der gewünschten Aktion an: Strg + A markiert alles, Strg + S speichert das aktuelle Dokument. [1]

#### 4.1 Mentale Modelle

Während des Lernens baut der Mensch mentale Modelle des zu Lernenden auf. Dies gilt sowohl beim Erlernen von Fakten und Konzepten als auch von Fähigkeiten und Abläufen.

Ein Beispiel hierfür ist das Schreiben eines Textes auf einem Tablet-PC. Wie beim Computer auch wird der Text in der Regel über eine QWERTZ-Tastatur, allerdings meist einer virtuellen, eingegeben. Diese funktioniert genauso wie die physische die vor dem Bildschirm auf dem Schreibtisch steht. Allerdings gibt es auch Abweichungen die im mentalen Modell für das Tablett Berücksichtigt werden müssen. Zum Beispiel beim Verschieben und Markieren von Text. Durch Klicken und Ziehen wird auf dem Computer Text markiert. Auf einem Tablett hat diese Geste in der Regel die Bedeutung, dass die Seite verschoben wird.

Zum Lernen gehört auch das Wiederholen und Üben des zu Erlernenden. Je häufiger und konzentrierter geübt wird, desto größer ist auch der Lernerfolg. Durch das Wiederholen des Erlernenden kann nach und nach ein mentales Modell aufgebaut und vervollständigt werden.

Erheblich beschleunigt werden kann das Lernen wenn Verknüpfungen zu schon Bekanntem aufgebaut werden können. Je mehr solcher Verknüpfungen aufgebaut werden können, desto leichter fällt es ein neues mentales Modell aufzubauen. [1]

#### 4.2 Metaphern

Durch die Verwendung von Metaphern kann gut an bereits erlerntes angeknüpft werden. Dateien werden beispielsweise durch entsprechende Symbole repräsentiert, die in Ordner gezogen werden können. Durch Aufnehmen und Verschieben, können sie wie in der realen Welt in beliebige Ordner verschoben werden. Dabei müssen dem Anwender aber auch die Einschränkungen und Erweiterungen bewusst sein. Eine digitale Datei kann, im Gegensatz zu ihrem realen Gegenstück, problemlos sofort und beliebig oft kopiert werden.



**Abb. 6.** Die Funktionen der Symbole in der Symbolleiste sind dank Metaphern schnell zu erlernen.

Bei der Verwendung von Metaphern muss stets darauf geachtet werden, dass die Metaphern konsistent verwendet werden, um Verwirrung beim Anwender zu

vermeiden. Das Ablegen einer Datei sollte entweder zum sofortigen Löschen von dieser führen, oder zur „Ablage“ im Papierkorb. Wenn sie einmal nur „Abgelegt“ und ein andermal direkt gelöscht wird führt dies zu Verwirrung. Ein Beispiel für die Verwendung von Metaphern in Benutzeroberflächen ist in Abbildung 6 zu sehen. [1]

## 5 Erfahrung

Jeder Anwender bringt einen unterschiedlichen Erfahrungsschatz mit. Dieser drückt sich durch das Vorwissen des Anwenders aus und die Art und Weise wie er eine Aufgabe löst. Die beiden Extreme des Erfahrungsschatzes sind auf der einen Seite der Anfänger und auf der anderen der Experte. Meist wird der Anwender einen Erfahrungsschatz zwischen diesen beiden Extremen mitbringen.

Ein Anfänger verfügt über wenig Wissen und Fertigkeiten. Zur Lösung einer Aufgabe muss dieser seine Aufmerksamkeit oft bewusst auf die Bedienung richten. Das heißt, er kann seine Aufmerksamkeit nicht voll auf die Bewältigung der Aufgabe richten, sondern muss sie immer wieder mit der Bedienung des Gerätes oder Software teilen. Als Folge dessen benötigt ein Anfänger meist mehr Zeit als ein Experte um eine Aufgabe zu erledigen und macht dabei zudem häufiger Fehler. [1]

## 6 Aufmerksamkeit

Die Aufmerksamkeit kann, wie bereits in Abschnitt 3.2 dargelegt, in einem Moment nur bewusst auf eine Aktivität gerichtet werden. Dies wirkt sich auch auf die Durchführung von Aufgaben aus.

Obwohl es durchaus möglich ist mehrere Dinge parallel durchzuführen, ist die Aufmerksamkeit normalerweise nur auf eine Handlung gerichtet. Beim Autofahren konzentriert sich der Fahrer in der Regel voll darauf sich (weitgehend) an die Verkehrsregeln zu halten und dabei nichts zu überfahren. Das Gas geben, Bremsen und Kuppeln läuft dabei automatisch ab. Muss der Fahrer, zum Beispiel durch eine Frage von einem Mitfahrer, nachdenken, ist die Aufmerksamkeit nicht mehr voll auf den Straßenverkehr gerichtet. Das Steuern des Autos wird dabei in aller Regel automatisch fortgeführt. Passiert jedoch etwas unvorhergesehenes, dauert es einen kurzen Moment bis die Situation aus dem sensorischen Kurzzeitgedächtnis ins Kurzzeitgedächtnis gelangt und bewusst darauf reagiert werden kann.

Deckt sich eine Anwendung mit der Erfahrung und den Erwartungen des Anwenders, so fällt diesem die Bedienung der Benutzeroberfläche leichter. Im Idealfall kann sich der Anwender voll auf die zu erledigende Aufgabe konzentrieren und die Bedienung der Anwendung tritt in den Hintergrund, da sie ohne Nachdenken automatisch erfolgt.

Aus diesem Grund kann es besser sein die Erledigung einer Aufgabe in mehrere Masken, mit einem klar definierten Ziel, aufzuteilen, anstatt eine große Maske,



die alles enthält, zu verwenden. Durch diese Zerlegung der Aufgabe in Teilaufgaben wird es dem Anwender erleichtert sich auf einen bestimmten Aspekt zu konzentrieren. Als Resultat daraus kann sich der Anwender besser auf die Teilaufgaben konzentrieren und muss sich nicht auch noch in einer überladenen Oberfläche zurechtfinden.

Trotz aller Bemühungen muss die Bedienung von neuen Oberflächen meist zunächst erlernt werden. Ist die Benutzerführung aber logisch und in sich konsistent, ist es sehr wahrscheinlich, dass die Bedienung sehr schnell in den Hintergrund tritt und sich der Anwender voll auf die zu erledigende Aufgabe konzentrieren kann. [1]

## 7 Fehler

Mit Fehlern sind hier keine technischen Fehler gemeint, sondern solche die in Handlungen vorkommen. In der Regel lassen sich diese auch nicht vermeiden, sondern es kann ihnen höchstens vorgebeugt werden.

Jeder Fehler beeinträchtigt die Zufriedenheit des Anwenders. Außer dieser hat absichtlich auf diesen hingewirkt, um zum Beispiel die Pflichtfelder eines Formulars herauszufinden. Unterteilen lassen sich die Fehler in zwei Klassen: Schwere Fehler die das Erfüllen der Aufgabe verhindern und leichte Fehler nach deren Korrektur die Aufgabe trotzdem erfüllt werden kann. [1]

### 7.1 Intellektuelle Fehler

Handelt es sich bei dem intellektuellen Fehler um einen Denkfehler, ist es Softwaretechnisch quasi nicht möglich diesen zu verhindern. Soll beispielsweise ein Logo als Vektorgrafik erstellt werden, wird aber stattdessen als Pixelgrafik erstellt, kann auch die beste Software nicht helfen – sie hat ihre Aufgabe wie vorgesehen erfüllt.

Anders sieht es bei intellektuellen Fehlern aus, die auf einem falschen mentalen Modell des Anwenders basieren. Zu allererst sollte eine Anwendung, wie bereits in Abschnitt 4.1 besprochen, so entworfen werden, dass sie das mentale Modell der Mehrheit der Anwender berücksichtigt. Je nach Zielgruppe kann dieses mentale Modell jedoch erheblich variieren. Eine Bildbearbeitung für Heimwender wird ein anderes Bedienkonzept aufweisen als das für professionelle Anwender. In Ersteren wird zum Ausschneiden eines Bildausschnittes eine eigene Funktion angeboten, während in der Professionellen zunächst eine Auswahl erstellt und anschließend, „versteckt“ in einem Menü, ausgewählt werden muss, dass das Bild auf diese Auswahl zugeschnitten werden soll. Um auch Umsteigern das Verstehen der Funktionen zu ermöglichen, sollten diese in der Dokumentation berücksichtigt werden. Auch die gruppierte Anordnung von ähnlichen Steuerelementen kann zum schnelleren Aufbau eines korrekten mentalen Modells beitragen. [1]

## 7.2 Fehler in flexiblen Handlungsmustern

Solche Fehler basieren darauf, dass der Anwender Ausnahmen ignoriert und wie gewohnt weiterarbeitet. Oft wird diese Ignoranz beim Anwender durch die Ignoranz des Anwendungsentwicklers verursacht!

Werden auch kleine Probleme, die auf das Handeln des Anwenders keine direkten Auswirkungen haben, gemeldet und das schlimmstenfalls mit einem ähnlichen Aussehen wie bei schwerwiegenden Ausnahmen, führt dies zwangsläufig zu einer Abstumpfung. Rein informative Meldungen sollten auf ein Minimum reduziert werden, um einer solchen Abstumpfung vorzubeugen. Außerdem sollten verschiedene Klassen von Meldungen auf einen Blick klar voneinander abzugrenzen sein.

Gängig ist einer Aufteilung in vier Klassen, die sich, für den Anwender klar erkennbar, in ihrer Gestaltung unterscheiden.

**Fehler** informiert den Anwender über ein bereits eingetretenes Problem. In der Regel wird eine solche Meldung mit Rottönen gestaltet.

**Warnung** informiert den Anwender über einen Zustand der, sofern er nicht behoben wird, zu einem Problem führen kann. In der Regel wird eine Warnung durch Gelb und Orangetöne gekennzeichnet.

**Frage** fordert vom Anwender eine weitere Bestätigung an, ob er eine Aktion tatsächlich durchführen will. In der Regel wird eine Frage durch einen blauen Farbton gekennzeichnet.

**Information** informiert den Anwender lediglich. Sie sind immer nützlich und relevant, aber niemals kritisch und können vom Benutzer deshalb problemlos ignoriert werden. Wie Fragen werden Informationen häufig mit einem blauen Farbton gestaltet. Je nach Anwendungsfall kann davon aber auch abgewichen werden, im Falle einer Bestätigung zum Beispiel zugunsten eines Grüntones.

Je nach Software richtet sich diese an Anwender mit einem unterschiedlichen Erfahrungsschatz. Hier kann auf ein Verfahren zurückgegriffen werden, bei dem der Anwender selbst entscheiden kann welche Meldungen er angezeigt bekommen möchte. Realisiert werden kann dieses durch ein Auswahlkästchen über das der Anwender die Anzeiger dieser Meldung für die Zukunft verhindern kann. Sinnvoll ist dies beispielsweise beim Speichern eines Bildes in einem anderen Format. Ein Erfahrener Anwender möchte hier nicht ständig von Meldungen genervt werden, dass diese Formatänderung zu einer Verschlechterung der Bildqualität führen kann, während diese Meldung bei unerfahrenen Anwendern durchaus ihre Berechtigung haben kann. [1]

## 7.3 Sensomotorische Fehler

Hierbei handelt es sich um Fehler, die sich aus einer mangelnden Koordination zwischen den Sensoren und der Motorik ergeben. Informationen werden falsch wahrgenommen und deshalb an einer falschen Stelle geklickt, oder die Durchführung der Bewegung selbst erfolgt unkonzentriert.

Verhindert werden können solche Fehler unter anderem dadurch, dass Interaktionselemente eine angemessene Größe erhalten. Der zur Verfügung stehende Platz sollte auch tatsächlich verwendet werden. Dies gilt insbesondere im Hinblick auf den Trend, dass immer mehr berührungsempfindliche Bildschirme im Alltag Einzug halten. Zu kleine Elemente können hier mit den, im Vergleich zum Mauszeiger, wuchtigen Fingern nicht sicher getroffen werden. Aber auch mit der Maus ist es schwierig zu kleine Elemente zu treffen. Hierdurch wird der Anwender gezwungen seinen Kontext auf die Benutzeroberfläche zu richten, anstatt diese, gemäß Abschnitt 3.2, in den Hintergrund treten zu lassen. [1]

## 8 Antwortzeiten

Bei der Kommunikation mit einem Gerät treten unweigerlich Verzögerungen auf. Diese Resultieren aus der Bearbeitungszeit die ein Gerät zur Verarbeitung der Nutzereingaben benötigt. Sie setzt sich zusammen aus der Bearbeitung der Nutzereingabe durch das Gerät und der anschließenden Darstellung des Ergebnisses. Je nach Aufgabe kann diese Wartezeit variieren. Die Antwortzeiten lassen sich grob in folgende Geschwindigkeitsklassen einteilen:

**Bis ca. 1 Sekunde** wird die Antwort als unmittelbar empfunden.

**Bis ca. 5 Sekunden** wird die Antwort als verzögert empfunden.

**Bis ca. 10 Sekunden** wird die Antwort als stark verzögert empfunden.

**Ab ca. 10 Sekunden** wird davon ausgegangen, dass keine Antwort mehr erfolgt.

Zu beachten ist, dass es sich hier lediglich um Richtwerte handelt. Je nach Anwendungsfall, Benutzergruppe und der Vorerfahrung des jeweiligen Nutzers, können diese Werte variieren. [1]

Allgemein gilt: Ist die zu erwartende Wartezeit groß, sollte der Anwender durch einen entsprechenden Hinweis darauf hingewiesen werden. Idealerweise enthält dieser Hinweis auch eine Information darüber wann mit der Antwort zu rechnen ist. Trotzdem kann hierdurch das Problem des Zeitverlustes für den Anwender nicht kompensiert werden. Benötigt eine Internetseite sehr lange zur Antwort, ist die Konkurrenz meist nur einen Klick entfernt. Reagiert eine Anwendung nur langsam, wird beim nächsten Mal eben das Produkt eines anderen Herstellers gekauft.

Zum Vergleich wie wichtig die Ladezeiten sind, seien hier zwei Beispiele genannt. Für die Firma Google bedeutet eine Zunahme der Ladezeit ihrer Internetseiten von 0,4 auf 0,9 Sekunden eine Verringerung der Besuche um 20%. Für die Firma Amazon bedeutet eine Zunahme der Ladezeit um 0,1 Sekunden einen Umsatzverlust von 1%.

## 9 Layoutgestaltung

Wichtig für die Übersichtlichkeit und leichte Erfassbarkeit der Inhalte einer Anwendung ist ein klares und einheitliches Layout. Innerhalb von diesem Layout

werden alle Elemente einer Benutzeroberfläche angeordnet – Texte, Überschriften, Bilder, Tabellen, usw. Ein Layout mit einem festen Raster gibt den Elementen einer Anwendung eine feste Struktur. An dieser können sich die Entwickler beim Entwurf orientieren. Wird diese Struktur vom Entwickler einheitlich und konsequent verwendet, bietet diese dem Anwender eine erhöhte Übersichtlichkeit – die Navigation, Überschrift und die Inhalte befinden sich immer an der erwarteten Stelle und müssen nicht erst gesucht werden.

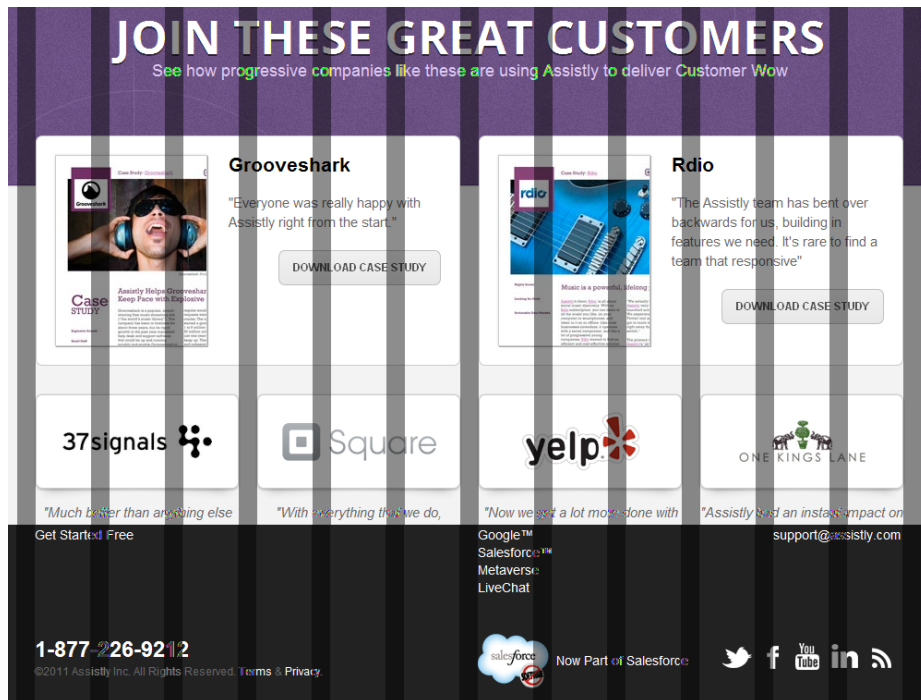


Abb. 7. Die Elemente der Seite Assystly.com sind an einem Raster ausgerichtet. Mehrere Spalten können beliebig zu breiteren zusammengefasst werden.

Ein festes Raster bewahrt den Entwickler ebenfalls davor für jede Seite ein neues Layout zu entwerfen. Stattdessen wird auf bestehendes zurückgegriffen. Dies mag zunächst einschränkend wirken, erhöht aber die Produktivität, die Wartbarkeit und die Zufriedenheit des Anwenders. Der Entwickler muss das Rad nicht jedes Mal neu erfinden, die Struktur ist immer ähnlich, sowohl für andere Entwickler als auch den Anwender. Entwickler können, unter Beachtung des Rasters und der Grundstruktur, trotzdem verschiedene und interessante Oberflächen und Inhalte entwerfen. [2]

## 9.1 Spaltenbreite und Abstände

Die Spaltenbreite und davon indirekt abhängig auch die Anzahl der Spalten, kann prinzipiell je nach Anforderung der Anwendung gewählt werden. Allerdings sollten dabei einige Punkte beachtet werden.

- Die Breite einer einzelnen Spalte sollte höchstens 80 Zeichen betragen. Viele schmale Spalten sind möglich, da diese miteinander verbunden werden können und damit eine Größere bilden.
- Zwischen Spalten sollte ein Abstand vorgesehen werden. Dieser sollte sich nach der gewählten Schriftart und -größe richten. Als Faustregel gilt: Ein Zwischenraum in der Größe des optischen Zeilenabstandes ist optimal, sollte aber mindestens die Breite der beiden Buchstaben „mi“ betragen.

Eine solche Aufteilung ermöglicht es dem Auge sicher in die nächste Zeile zu Springen. Hierdurch ist Text leichter und ermüdungsfreier zu lesen. Andere Elemente wie Text sind davon völlig unberührt, sie müssen sich lediglich in das Raster einpassen. Ihre Größe kann auch mehrere Spalten umfassen, da mehrere Spalten zu einer zusammengefasst werden können. Ein Beispiel für die beschriebenen Techniken ist in Abbildung 7 zu sehen. [2]

## 10 Farbgestaltung

Farbe hat auf den Menschen verschiedene Wirkungen. Sie ist nicht nur ein Gestaltungselement, sondern kann auch verschiedene Stimmungen und Assoziationen beim Betrachter hervorrufen. Wichtig ist: Farbe ist immer abhängig von der menschlichen Wahrnehmung. Verschiedene Menschen eines Kulturkreises werden Farben zwar ungefähr gleich wahrnehmen und deuten – die Interpretation eines jeden kann aber durch die individuelle Erfahrung abweichen. [2]

### 10.1 Farbwirkung

Die Wirkung von Farben wird stark durch kulturelle Einflüsse geprägt. Was im Westen als Farbe der Reinheit gilt, ist im asiatischen Raum die Farbe der Trauer. Außerdem wird die Farbpfindung noch einmal durch die individuelle Erfahrung geprägt. So wird der eine die Farbe Rot mit der Liebe verbinden, während ein anderer an Blut denkt. In der westlichen Welt haben die Farben grob die folgende Bedeutung:

**Blau** Beruhigend und entspannend.

**Gelb** Neid, Verrat und Feigheit.

**Grün** Farbe der Mitte. Beruhigend ohne zu ermüden.

**Orange** Vitale Stärke und Aktivität. Kraftspender.

**Rot** Erhöht Energiepegel und seelische Kraft.

**Violett** Geist und Spiritualität.

**Weiß** Vollkommene Farbe. Reinheit.

**Schwarz** Tod und Trauer.

Daneben gibt es außerdem eine Einteilung in warme und kalte Farben:

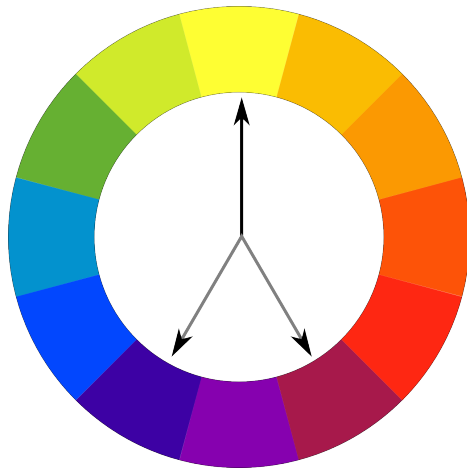
**Warme Farben** sind alle Farben von Gelbgrün bis Violett-Rot. Sie werden mit der warmen Jahreszeit verbunden, die von Gelb- und Rottönen bestimmt wird.

**Kalte Farben** sind alle Farben die auf dem Farbkreis gegenüber der warmen Farben liegen. Sie erinnern an das Kalte, blaues Wasser, Eisberge und das blaue Schimmern von Eis im Winter. [2]

## 10.2 Farbwahl

Für neue Publikationen, Internetseiten, Logos, usw. stellt sich immer wieder die Frage nach den richtigen Farben. Erleichtert werden kann diese Wahl durch Kenntnisse in der Farbwahrnehmung und der in Abschnitt 10.1 besprochenen Farbwirkung.

Zunächst muss eine Grundfarbe gefunden werden, auf der Aufbauend die weitere Farbwahl stattfindet. Für eine harmonische Farbwahl bietet es sich an, zu der gewählten Grundfarbe zwei ihrer Nachbarkomplementärfarben zu verwenden, wie in Abbildung 8 zu sehen. Der Kontrast einer Grundfarbe mit ihren beiden Nachbarkomplementären wird als besonders harmonisch empfunden. Die Nachbarkomplementäre sind die Farben, die links und rechts gleich weit von der Komplementärfarbe entfernt liegen. Als Unterstützung bei dieser Wahl kann zum Beispiel auf die Webanwendung [colorshemedesigner.com](http://colorshemedesigner.com) verwendet werden.



**Abb. 8.** Zu der Grundfarbe Gelb bieten sich als harmonische Nachbarkomplementärfarben Dunkelblau und Magenta an.

Bei der Wahl von Farben für Texte sollte beachtet werden, dass diese einen starken Kontrast zu ihrem Hintergrund aufweisen. Bei schwachen Kontrasten fällt es schwer den Text vom Hintergrund zu unterscheiden. Sollte es trotzdem notwendig sein eine Schriftfarbe mit schwachem Kontrast zu verwenden, sollten die Buchstaben durch eine kontraststarke Umrandung oder Schatten deutlich vom Hintergrund abgehoben werden. [2]

## 11 Typographie

Wichtig für die Lesbarkeit von Texten ist deren Typographie. Insbesondere für Internetseiten, die in der Regel einen hohen Anteil von Fließtext enthalten, ist diese wichtig.

Für Text sollte eine maximale Breite von 80 Zeichen vorgesehen werden. Dies ist jedoch nur ein grober Richtwert der von vielen anderen Faktoren, insbesondere auch der Schriftgröße, beeinflusst wird. Gerade auf Internetseiten sind oft auch Texte mit einer größeren Breite zu finden. Es sollte jedoch vermieden werden, dass der Anwender beim Lesen außer den Augen auch den gesamten Kopf selbst mit drehen muss. Als Richtwert sollte der Text nur so breit sein, wie er auch auf einen Blick überschaut werden kann.

Absätze erleichtern den Augen die Orientierung, da es sich um aus dem reinen Fließtext hervorstechende Merkmale handelt. Sie ermöglichen es den Augen auch bei größeren Zeilenlängen sicher zum nächsten Zeilenanfang zu springen. [2]

### 11.1 Schriftgröße

Die Schriftgröße ist ein wichtiges Element der Benutzerfreundlichkeit. Sowohl zu kleine als auch zu große Schriftgrößen sind schlecht für die Benutzerfreundlichkeit. Als Orientierung für Schriften auf Bildschirmen kann folgende Einteilung dienen:

**Konsultationsgrößen** sind alle Schriftgrößen bis 7 pt. Sie sollten nicht für Haupttexte verwendet werden, sondern nur für zum Beispiel Fußnoten.

**Lesegrößen** sind Schriftgrößen von 8–12 pt. Sie sind optimal für Fließtexte, Meldungen und andere wichtige Elemente einer grafischen Benutzeroberfläche.

**Schaugrößen** sind Schriftgrößen von 13–48 pt. Sie werden oft für Überschriften und Titel verwendet.

**Plakatgrößen** sind alle Schriftgrößen größer als 48 pt. Sie kommen überall dort zum Einsatz, wo der Text auch noch aus größerer Entfernung gelesen werden können soll (Plakate, Messestände, usw.).

Bei den Lesegrößen sollte beachtet werden, dass größere Schriftarten Brillenträgern und anderen mit einer Sehschwäche zugutekommen. Bestenfalls können diese einen Text mit 12 pt noch lesen, während sie bei einem Text mit 8 pt eine Sehhilfe benötigen. Von einer Schriftgröße über 12 pt sollte für Fließtexte aber ebenfalls abgesehen werden, da hier für Menschen ohne Sehschwäche die Lesbarkeit langer Texte zu leiden beginnt. [2]

## 12 Usability in Open Source Software

Für so gut wie jeden Anwendungsfall existiert heutzutage eine Open Source Software (OSS). Betrachtet man jedoch den Rechner eines typischen Anwenders mit einem Microsoft Windows, wird man kaum Open Source Software auf dessen Rechner finden. Dies mag zum Einen an der Popularität und besseren Werbung von kommerziellen Anwendungen liegen, liegt aber oft auch daran, dass die Anwender die Bedienung einer OSS nicht verstehen.

### 12.1 Probleme für die Usability in der Open Source Entwicklung

Zentrales Problem bei der Entwicklung von OSS ist, dass die Entwickler meist Programmierer sind. Profis für Usability und Design sind in den Entwicklungsprozess meist nur wenig eingebunden. Bei den Entwicklern handelt es sich im Normalfall um Experten und Poweruser – deren Erfahrung und Erwartungen sich meist kaum mit der der Zielgruppe deckt.

Hinzu kommt, dass es kaum einen Ansporn dafür gibt auf Usability zu achten. Im Gegensatz zu Bugreports ist sie nur schlecht messbar und Preise werden damit (explizit) auch nicht gewonnen. Meist ist OSS Codezentriert. Mit schönem Code können aber leider nur die Programmierer etwas anfangen, dem Endanwender ist dies aber völlig egal. Ist die Software nicht bedienbar hilft ihm auch der schönste Code nichts.

### 12.2 Integration von Usability in den Entwicklungsprozess

Wichtig ist es ein Bewusstsein für die Notwendigkeit des Usability Engineerings zu schaffen. Zwar werden damit keine Preise gewonnen, wenn eine Anwendung aber wegen mangelnder Usability niemand verwendet, wird sie auch keiner für einen Preis vorschlagen.

Um den Entwicklern einen Orientierungspunkt zu geben, sollten Gestaltungsrichtlinien erstellt werden an denen sich diese Orientieren können. Dabei sollte immer beachtet werden, dass allein das Befolgen dieser Richtlinien nicht sofort auch zu Software mit einer perfekten Usability führt. Wäre dies der Fall, könnten auch große Unternehmen völlig auf Tests mit Anwendern verzichten. Der Weg zur perfekten Usability ist ein iterativer Prozess, der in der Regel perfekt zum ebenfalls iterativen Vorgehen bei der Entwicklung von OSS-Projekten passt.

Einfache und billige Usability-Testverfahren können auch von einer Open Source Community etabliert werden. Dazu zählt beispielsweise das Remote Testing. Schön ist es, wenn hierfür echte Anwender gewonnen werden können. Stehen diese (noch) nicht zur Verfügung, ist aber auch eine abgewandelte Form möglich: Aus dem Remote Testing wird ein Local Testing mit eventuell technisch weniger begabten Familienmitgliedern oder der Freundin.

Bei der Entwicklung von OSS sollten, wie es bei jeder Closed Source Software auch sein sollte, schon früh Usability-Experten hinzugezogen werden. Sinnvoll ist ebenso die Gründung einer Gruppe die für Usability und deren Test zuständig



ist. Wichtig ist hier eine klare Entscheidungsstruktur mit einem hauptverantwortlichen Designer zu etablieren.

Wichtig ist sich der Zielgruppe bewusst zu werden. Wer ist meine Zielgruppe? Welche Erfahrung hat meine Zielgruppe? Was möchte meine Zielgruppe mit der Verwendung meiner Software erreichen? Nützlich ist hierfür die Definition verschiedener fiktiver Anwender.

**Ernst Fröhlich** 56 Jahre; Unternehmensberater; Grundlegende Kenntnisse mit Office Software; Leidet unter Kurzsichtigkeit; Verbringt seine Freizeit gern im eigenen Schrebergarten; Hobbys: Aktienkurse beobachten.

**Julia Heiter** 19 Jahre; Auszubildende zur Industriekauffrau; Kenntnisse in Office und Finanzsoftware; Verwendet den PC hauptsächlich für Facebook und trifft sich in ihrer Freizeit oft mit Freunden; Hobbys: Tanzen, Reiten, Feiern.

Durch diese fiktiven Anwender gelingt es leichter sich mit den Anwendern zu identifizieren. Außerdem schaffen diese eine gemeinsame Basis auf deren Grundlage diskutiert werden kann.

### 12.3 Usability Engineering in OpenSource-Projekten

Bei der Blogplattform Wordpress wird seit der Entwicklung zur Version 2.7 die Benutzerfreundlichkeit der einzelnen Module standardmäßig getestet. Es fiel hierbei auf dass das Testen der Softwarekomponenten mit kleinen Benutzerzahlen keine adäquaten Resultate liefert da die Benutzer aus unterschiedlichsten Regionen der Welt stammen und schon auf Basis der Sprachpakete unterschiedliche Ansichten auf das zu testende Produkt bekamen.

Um auf diese Probleme einzugehen entwickelt die Wordpress-Gemeinschaft ein eigenes System über welches sich Benutzer anmelden können um an den Testfällen teil zu nehmen. Entwickler können in diesem System Testfälle einstellen, die anschließend von allen angemeldeten Benutzern bearbeitet, bewertet und kommentiert werden können.

Das Resultat der herausgebrachten Version 2.7 war ein großer Erfolg und fand viel Zuspruch bei den Anwendern, weshalb der Prozess des Usability Engineerings, bei Wordpress nun standardmäßig zu der Entwicklung neuer Programmversionen dazugehört.

## 13 Literatur

1. Dahm Markus: Grundlagen der Mensch-Computer-Interaktion. Pearson Studium, 2006
2. Norbert Hammer: Mediendesign für Studium und Beruf. Springer-Verlag, Berlin, Heidelberg, 2008
3. Jakob Nielsen: Usability Engineering. Morgan Kaufmann Verlag, USA, 1993
4. Jane Wells: Contributing to WordPress, Part III: Usability Testing.  
<http://wordpress.org/news/2009/05/testing-ops/>, Abgerufen am 06.12.2011