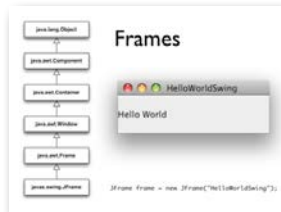


Benutzeroberflächen

Software-Praktikum
Andreas Zeller, Universität des Saarlandes
mit Christian Lindig

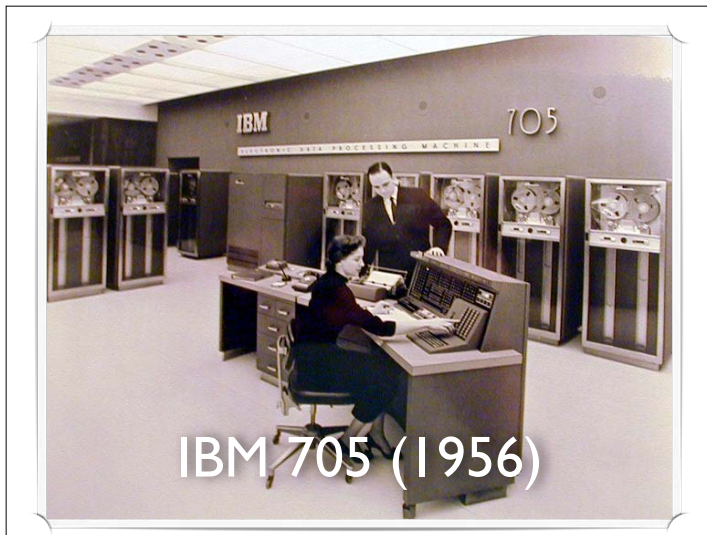


Benutzeroberflächen

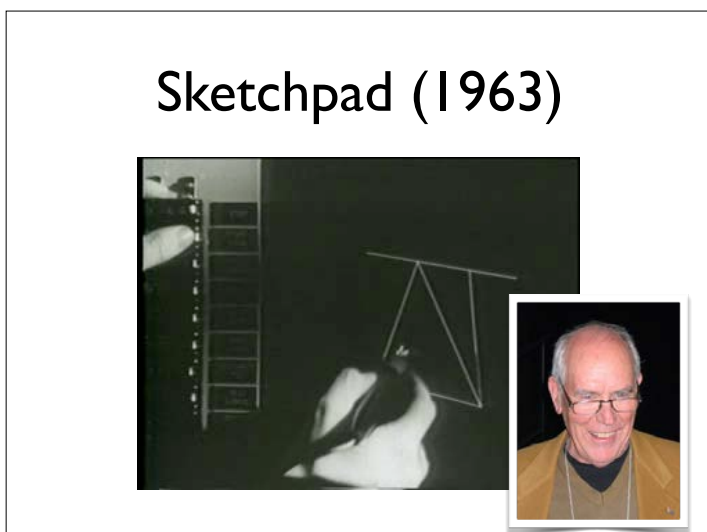


Lochkarten (1725–)





http://www-03.ibm.com/ibm/history/exhibits/mainframe/mainframe_PP705.html



Thesis by Ivan Sutherland; origin of CAD, GUI, OO; Turing Award in 1988

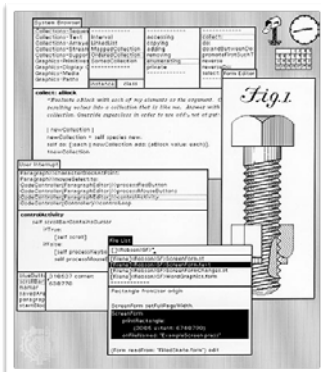
<http://en.wikipedia.org/wiki/Sketchpad>

Terminals (1970–)



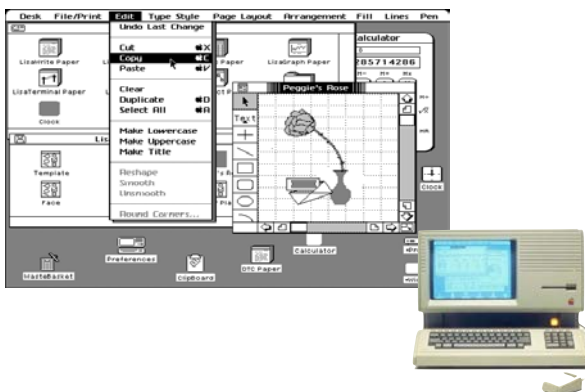
WIMP – „Windows“, „Icons“, „Menus“, and „Pointer“

Xerox Alto (1973)

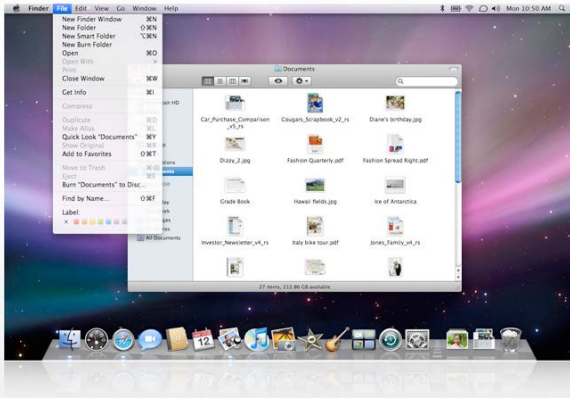


Alle Eigenschaften moderner Benutzeroberflächen

Apple Lisa (1983)



Mac OS X (2008)



... eigentlich bis heute unverändert ...

Windows 8.1 (2014)



... wenn man von ein paar Spielereien absieht.

Oberflächengestaltung

EN ISO 9241 – Ergonomie der Mensch-System-Interaktion



<http://kommdesign.de/texte/din.htm>

Aufgabenangemessenheit

Einfach und direkt zum Ziel

"Ein Dialog ist *aufgabenangemessen*, wenn er den Benutzer unterstützt, seine Arbeitsaufgabe effektiv und effizient zu erledigen."

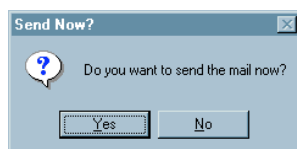
Aufgabenangemessenheit

einfach und direkt zum Ziel

- Eine Website verzeichnet Ansprechpartner.
- Formulare verlangen nur solche Angaben, die für den Vorgang relevant sind.
- Die Wartezeiten sind minimal.
- Bei Fehlern wird der Cursor gleich auf das zu korrigierende Feld gesetzt.
- Zwischenergebnisse einer längeren Online-Transaktion können gespeichert werden.
- Es gibt Shortcuts zu den wichtigsten Seiten.

Beispiele für mangelnde Effizienz

Jedesmal, wenn man in *Lotus Notes* eine e-mail absenden will, verlangt Notes eine Bestätigung. Das kann auf Dauer recht anstrengend werden:

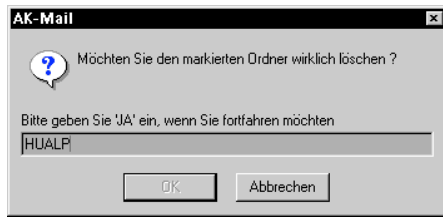


Wenn der Benutzer schon einmal auf *Absenden* gedrückt hat, warum nicht einfach anerkennen, daß er wohl *Absenden* meint?

Aufgaben-
angemessenheit

...

Noch schlimmer treibt es da *AK-Mail*:

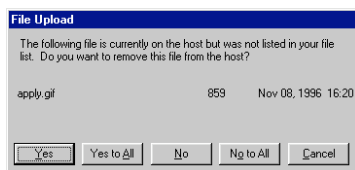


Warum eigentlich *JA*? Warum nicht gleich *JAAA*, *VERFLUCHT!*?

Aufgaben-
angemessenheit

...

Microsoft's *Web Wizard* ist ein Programm zum Verwalten von Dateien auf WWW-Servern. Während des Ablaufs erstellt Web Wizard eine Liste der Dateien auf dem Server und fragt ab, für jede Datei einzeln, ob sie gelöscht werden soll:

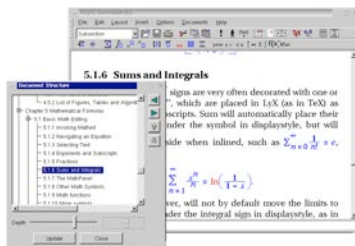


Wenn man etwa die Datei *zeller.gif* löschen wollte, müsste man zunächst 400mal auf *No* klicken – für jede der Dateien – die im Alphabet davor stehen.

Aufgaben-
angemessenheit

Gute Effizienz

In den Textverarbeitungsprogrammen LyX und KLyX gibt es einen *Formeleditor*, mit dem Anfänger Formeln über ein Menü zusammensetzen können:



Aufgaben-
angemessenheit

...mit der Tastatur

Der fortgeschrittene Benutzer kann aber auch direkt über die Tastatur TeX-Kommandos eingeben – etwa

```
\sum_{n = 1}^{\infty} \frac{x^n}{n} =
```

```
\ln\left(\frac{1}{1 - x}\right)
```

für

$$\sum_{n=1}^{\infty} \frac{x^n}{n} = \ln\left(\frac{1}{1-x}\right) .$$

Bereits mit der ersten Eingabe von \ schaltet KLyX in den TeX-Eingabemodus.

Flexible Programme sind meist auch effizient!

Aufgaben-
angemessenheit

Selbstbeschreibungsfähigkeit

Wenn Intuition ausreicht

"Ein Dialog ist *selbstbeschreibungsfähig*, wenn jeder einzelne Dialogschritt durch Rückmeldung des Dialogsystems unmittelbar verständlich ist oder dem Benutzer auf Anfrage erklärt wird."

Selbstbeschreibungsfähigkeit

wenn Intuition ausreicht

- Links sind so formuliert, dass man sicher vorhersagen kann, wohin sie führen.
- Der Umfang einer Treffer-Liste kann gleich am Tabellenanfang abgelesen werden.
- Eine Anwendung hat eine Online-Hilfe mit kontextspezifischen Bedienhinweisen.
- Nachdem eine Anfrage gesendet wurde, erscheint eine Meldung "Anfrage wird bearbeitet, bitte warten".

Selbsterklärende Dialoge

Selbsterklärende Dialoge steigern die Handlungskompetenz, indem sie das Wissen über das Software-System fördern.

Ein Dialog ist selbsterklärend, wenn

- dem Benutzer auf Verlangen Einsatzzweck sowie Leistungsumfang des Dialogsystems erläutert werden können und wenn
- jeder einzelne Dialogschritt
 - unmittelbar verständlich ist oder
 - der Benutzer auf Verlangen dem jeweiligen Dialogschritt entsprechende Erläuterungen erhalten kann.

Selbst-
beschreibungs-
fähigkeit

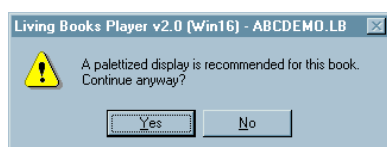
Selbsterklärende Dialoge (2)

Konkrete Maßnahmen für selbsterklärende Dialoge:

- Der Benutzer muß sich zweckmäßige Vorstellungen von den Systemzusammenhängen machen können
- Erläuterungen sind an allgemein übliche Kenntnisse der zu erwartenden Benutzer angepaßt (deutsche Sprache, berufliche Fachausdrücke)
- Wahl zwischen kurzen und ausführlichen Erläuterungen (Art, Umfang)
- Kontextabhängige Erläuterungen

Selbst-
beschreibungs-
fähigkeit

Schlechte Erläuterungen



Was ist ein „palettized display“?

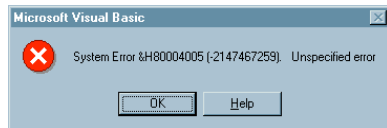
Ein PC-Experte mag diese Meldung vielleicht verstehen. Diese Warnung entstammt aber *Dr. Zeuss's ABC*, ein Alphabet-Lern-Programm für 3- bis 5jährige Kinder.

Noch bemerkenswerter: Die Warnung ist völlig überflüssig, denn auch ohne „palettized display“ funktioniert das Programm einwandfrei.

Selbst-
beschreibungs-
fähigkeit

Schlechte Erläuterungen (2)

Diese höchst aussagekräftige Fehlermeldung ist Microsoft *Visual Basic 5.0* zu entnehmen:



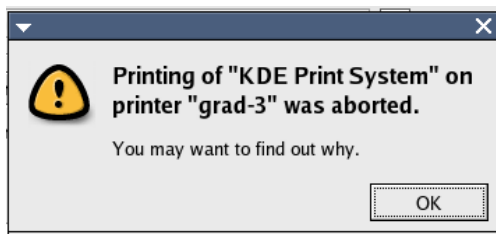
Nach dem Klicken auf *Help* erhalten wir:

Visual Basic encountered an error that was generated by the system or an external component and no other useful information was returned. The specified error number is returned by the system or external component (usually from an Application Interface call) and is displayed in hexadecimal and decimal format.

Lösung des Problems: Neu booten?

Selbst-
beschreibungs-
fähigkeit

Gibt es auch in KDE



Selbst-
beschreibungs-
fähigkeit

503 höfliche Leute



Selbst-
beschreibungs-
fähigkeit

“503 höfliche Leute sagen erstmal Hallo.”

Type mismatch



Selbst-
beschreibungs-
fähigkeit

Erschwerend kam hinzu, dass die arme Sekretärin, mit dieser Meldung konfrontiert, ein ums andere Mal "mismatch" eintippte ("type" = tippen), ohne dass irgendetwas passierte.

Unix kann das auch...

Zum Schluß eine unfreundliche Fehlermeldung der *Secure Shell*:

```
$ ssh somehost.foo.com
You don't exist, go away!
$ _
```

Diese Fehlermeldung erscheint etwa, wenn der NIS-Server gerade nicht erreichbar ist. Nicht, daß man den Benutzer darüber aufklären würde...

Selbst-
beschreibungs-
fähigkeit

Steuerbarkeit

Wenn alles unter Kontrolle ist

"Ein Dialog ist *steuerbar*, wenn der Benutzer in der Lage ist, den Dialogablauf zu starten sowie seine Richtung und Geschwindigkeit zu beeinflussen, bis das Ziel erreicht ist."

Steuerbarkeit

Wenn alles unter Kontrolle ist

- Eine Tabelle hat Buttons zum spaltenweisen Sortieren.
- Eine Suchmaschine bietet die Möglichkeit, die Zahl der Treffer pro Seite einzustellen.
- Ein Tool ermöglicht es, einen Download zu unterbrechen und später fortzusetzen.
- Große Grafiken werden als "Thumbnails" dargestellt, die bei Bedarf vom Benutzer vergrößert werden können.

Flexibilität

Eine Anwendung ist dann *flexibel*, wenn

- der Benutzer mit einer *geänderten Aufgabenstellung* seine Arbeit noch effizient mit demselben System erledigen kann,
- eine Aufgabe auf alternativen Wegen ausgeführt werden kann, die dem Benutzer entsprechend seinem *wechselnden Kenntnisstand* und seiner aktuellen Leistungsfähigkeit wählen kann,
- *unterschiedliche Benutzer* mit unterschiedlichem Erfahrungshintergrund ihre Aufgaben auf alternativen Wegen erledigen können.

Steuerbarkeit

Flexibilität (2)

- *Makrobildung* ermöglichen, d.h. Operationen bei wiederkehrenden Abläufen können zu einer einzigen Operation zusammengefaßt werden.
- *Mengenbildung* ermöglichen, d.h. Objekte, auf die die gleichen Operationen angewendet werden sollen, können zu größeren Einheiten zusammengefaßt werden
- Soweit wie möglich *nicht-modale Dialoge* verwenden.
Ein *modaler Dialog* schränkt im Rahmen einer Ausnahmesituation die Handlungsflexibilität ein (z.B. zur Fehlerbehebung, wenn erst danach weitergearbeitet werden kann).
- *Parallele Bearbeitung* mehrerer Anwendungen mit gegenseitigem Informationsaustausch vorsehen.

Generell: *Alternative Benutzeroperationen anbieten!*

Steuerbarkeit

Beispiele für geringe Flexibilität

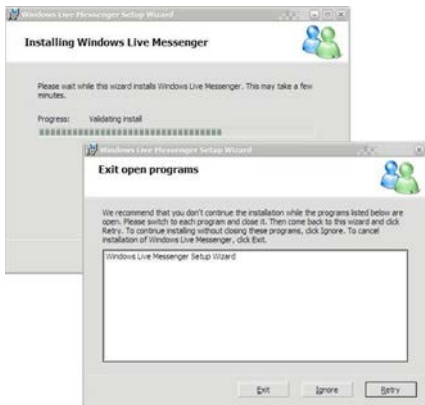
eZip ist ein Programm zum Entkomprimieren von Dateien. Die einzelnen Schritte sind genau vorgegeben:



- Was möchten Sie tun?
- Welche Optionen wünschen Sie?
- Welchen Namen möchten Sie angeben?
- usw. usf.

Steuerbarkeit

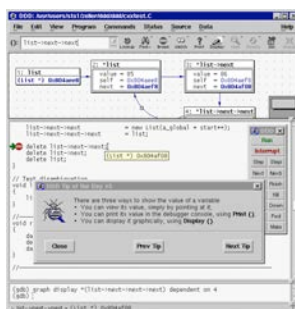
Modale Dialoge



Steuerbarkeit

Beispiele für große Flexibilität

DDD ist ein graphischer *Debugger*, mit dem der Ablauf eines Programms Schritt für Schritt untersucht werden kann.



Steuerbarkeit

Erwartungskonformität

Wenn es zur Gewohnheit wird

"Ein Dialog ist *erwartungskonform*, wenn er konsistent ist und den Merkmalen des Benutzers entspricht, z.B. seinen Kenntnissen aus dem Arbeitsgebiet, seiner Ausbildung und seiner Erfahrung sowie den allgemein anerkannten Konventionen."

Erwartungskonformität

wenn es zur Gewohnheit wird

- Der Link zur Startseite ist unter dem Firmenlogo oben links platziert.
- Der "Warenkorb" in einem Online-Shop heißt immer "Warenkorb".
- Unterstrichene Wörter sind immer Hypertext-Links.
- Beim Drücken der Tabulator-Taste springt der Cursor auf das nächste Eingabefeld.

Regelwerke für die Dialoggestaltung

Das wichtigste Hilfsmittel, einheitliche Programmbedienung zu erhalten, sind *Regelwerke* (*style guides*) für die Oberflächengestaltung.

Beispiel: Dialoggestaltung mit der GNOME-Bibliothek, einem Linux-Standard:

- *All dialogs should have at least one button that is labeled "Close", "Cancel", "OK", or "Apply".*
- *Modal dialogs should be avoided wherever possible.*
- *The default highlighted button for a dialog should be the safest for the user.*
- *All dialogs should default to a size large enough to display all of the information in the dialog without making a resize necessary.*
- *All dialogs should set the titlebar with an appropriate string.*

Regelwerke für die Dialoggestaltung (2)

- *A dialog which consists of a single entry box shall have its "OK" button be the default (which is to say that ENTER shall accept the entry), and the ESCAPE key shall be bound to the Cancel button.*
- *In a dialog the "OK" or "Apply" button should not be highlighted until the user has made a change to the configurable data in the dialog.*
- *If a dialog contains more than one button for the destruction of that dialog, (for example, an "OK" and a "Cancel"), the affirmative button should always fall to the left of the negative.*

Style Guides können mehrere hundert Seiten umfassen!

Erwartungs-
konformität

Bekannte Style Guides

Verbreitete Stile und Regelwerke der Dialoggestaltung sind

- *Windows* (Windows Interface Application Design Guide) – 580 Seiten
- *Motif* (OSF/Motif Style Guide) – 400 Seiten
- *MacOS X* (Aqua Human Interface Guidelines) – 310 Seiten

Manche Betriebssysteme (z.B. X Window System/UNIX), Programme (z.B. StarOffice), und Bibliotheken (z.B. Swing, Qt) unterstützen mehrere Stile – entweder wahlweise oder gleichzeitig.

Manche Widget-Sets sehen auf allen Plattformen gleich aus – ein Verstoß gegen die Erwartungen des Benutzers.

Generell nähern sich Aussehen und Verhalten der Oberflächen (*look and feel*) einander an.

Erwartungs-
konformität

Inkonsistenzen - Beispiele

Den Sinn einheitlicher Regeln für Benutzungsschnittstellen erkennt man am besten, wenn man *Verstöße* betrachtet.

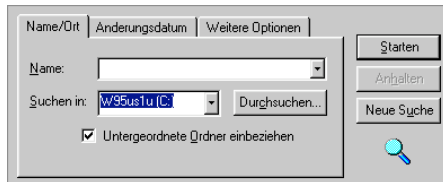
Auf einem *Macintosh*-Mülleimer kann man beliebige Objekte löschen, indem man sie auf den Mülleimer zieht.



Zieht man jedoch eine Diskette auf den Mülleimer, wird sie nicht gelöscht, sondern ausgeworfen. Ein „magischer“ Mülleimer, der neue Benutzer verwirrt.

Erwartungs-
konformität

Suchen in Windows (seit WIN95)

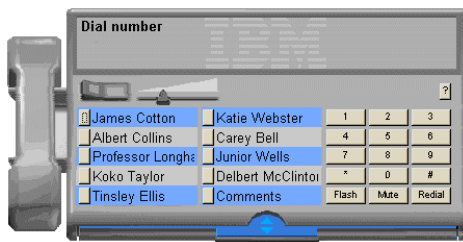


Originellerweise wird die Suche mit *Starten* aktiviert – der *Durchsuchen*-Knopf erlaubt nur die Auswahl eines Startverzeichnis.

Erwartungs-
konformität

Das IBM RealPhone

Das *IBM RealPhone*, ein Telefonprogramm, kommt ganz ohne Standard-Bedienungselemente aus:



In der wirklichen Welt mag es wichtig sein, anders auszusehen. Was Bedienung angeht, ist das Befolgen von Standards der richtige Weg.

Standardelemente

Benutzer erwarten standardisierte Bedienelemente, auch wenn es für sie keine physikalische Entsprechung gibt.



Erwartungskonforme Dialoge

Die Handlungskompetenz wird durch *erwartungskonforme Dialoge* unterstützt.

Ein Dialog ist erwartungskonform, wenn er den Erwartungen der Benutzer entspricht,

- die sie aus Erfahrungen mit bisherigen Arbeitsabläufen oder aus der Benutzerschulung mitbringen sowie
- den Erwartungen, die sie sich während der Benutzung des Dialogsystems und im Umgang mit dem Benutzerhandbuch bilden.

Erwartungs-
konformität

Erwartungskonforme Dialoge (2)

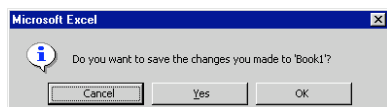
Konkrete Maßnahmen für erwartungskonforme Dialoge:

- Das Dialogverhalten ist einheitlich (z.B. konsistente Anordnung der Bedienungselemente).
- Bei ähnlichen Arbeitsaufgaben ist der Dialog einheitlich gestaltet (z.B. Standard-Dialoge zum Öffnen oder Drucken von Dateien)
- Zustandsänderungen des Systems, die für die Dialogführung relevant sind, werden dem Benutzer mitgeteilt.
- Eingaben in Kurzform werden im Klartext bestätigt.
- Systemantwortzeiten sind den Erwartungen des Benutzers angepaßt, sonst erfolgt eine Meldung.
- Der Benutzer wird über den Stand der Bearbeitung informiert.

Erwartungs-
konformität

Unerwartetes Dialogverhalten

Gewöhnlich schreiben Standards vor, wie Bedienungselemente anzuordnen sind – etwa *Bestätigen* vor *Abbrechen*, oder *Ja* vor *Nein*, oder *Hilfe* ganz rechts. *Microsoft Excel* macht alles anders:

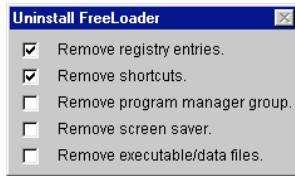


Wo ist der Unterschied zwischen Yes und OK?

Erwartungs-
konformität

...bei FreeLoader

Wenn man *FreeLoader*, einen WWW-Browser deinstalliert, erscheint dieses Fenster:



Der Benutzer mag denken, er könne auswählen, was denn nun tatsächlich deinstalliert werden soll. Weit gefehlt! Dies ist eine *Statusmeldung*, die anzeigt, was bereits deinstalliert wurde.

Originellerweise ändern die Knöpfe ihren Zustand, wenn man auf sie klickt – aber diese Änderung hat keine Auswirkungen.

Fehlertoleranz

Wenn unliebsame Überraschungen ausbleiben

"Ein Dialog ist *fehlertolerant*, wenn das beabsichtigte Arbeitsergebnis trotz erkennbar fehlerhafter Eingaben entweder mit keinem oder mit minimalem Korrekturaufwand seitens des Benutzers erreicht werden kann."

Fehlertoleranz

Wenn unliebsame Überraschungen ausbleiben

- Beim Rückwärtsbrowsen auf einer Webseite wird die Information aktualisiert damit nicht fälschlicherweise der Eindruck entsteht, Bearbeitungsschritte seien rückgängig gemacht worden.
- Über ein Skript werden die Daten eines Formulars auf Plausibilität geprüft bevor sie abgesendet werden.
- Fehlermeldungen werden in der Sprache der Benutzer formuliert.

Fehlerrobuste Dialoge

Ein Dialog ist *fehlerrobust*, wenn trotz erkennbar fehlerhafter Eingaben das beabsichtigte Arbeitsergebnis mit minimalem oder ohne Korrekturaufwand erreicht wird.

Dem Benutzer müssen Fehler verständlich gemacht werden, damit er sie beheben kann.



Fehlertoleranz

Fehlerrobuste Dialoge (2)

Konkrete Maßnahmen für fehlerrobuste Dialoge:

- Benutzereingaben dürfen nicht zu Systemabstürzen oder undefinierten Systemzuständen führen.

Aus der GCC-Dokumentation:

If the compiler gets a fatal signal, for any input whatever, that is a compiler bug. Reliable compilers never crash.

- Automatisch korrigierbare Fehler können korrigiert werden. Der Benutzer muß hierüber informiert werden.
- Die automatische Korrektur ist abschaltbar.
- Korrekturalternativen für Fehler werden dem Benutzer angezeigt.

Fehlertoleranz

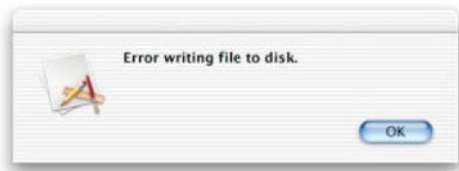
Fehlerrobuste Dialoge (3)

- Fehlermeldungen weisen auf den Ort des Fehlers hin. z.B. durch Markierung der Fehlerstelle.
- Fehlermeldungen sind
 - verständlich,
 - sachlich und
 - konstruktiv

zu formulieren und sind einheitlich zu strukturieren (z.B. Fehlerart, Fehlerursache, Fehlerbehebung).

Fehlertoleranz

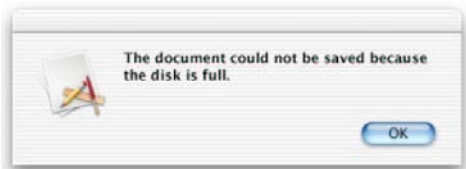
Eine schlechte Fehlermeldung



Eine Fehlermeldung, wie sie jeder von uns schon geschrieben hat.
Das Problem: wie soll der Benutzer darauf reagieren?

Fehlertoleranz

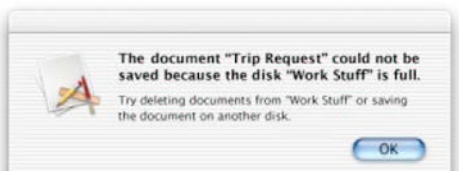
Eine bessere Fehlermeldung



Diese Fehlermeldung verrät die Ursache des Fehlers und gibt damit zumindest indirekt einen Hinweis, wie man ihn vermeiden könnte.

Fehlertoleranz

Eine gute Fehlermeldung



Eine Fehlermeldung, die keine Fragen offen lässt.

Fehlertoleranz

Auch schlecht

Diese Meldung erscheint in IBM's *Aptiva Communication Center*, wenn der Benutzer einen leeren Datensatz ausgewählt hat und auf den *Change*-Knopf klickt:



- Niemals eine Funktion anbieten, die in einer Fehlermeldung endet
- Stattdessen Funktionen *ausblenden*, die nicht angewandt werden können.

Denn Benutzer machen alle dummen Fehler die sie finden können!

Feedback



Individualisierbarkeit

Wenn jeder arbeitet, wie es ihm beliebt

"Ein Dialog ist *individualisierbar*, wenn das Dialogsystem Anpassungen an die Erfordernisse der Arbeitsaufgabe sowie an die individuellen Fähigkeiten und Vorlieben des Benutzers zulässt."

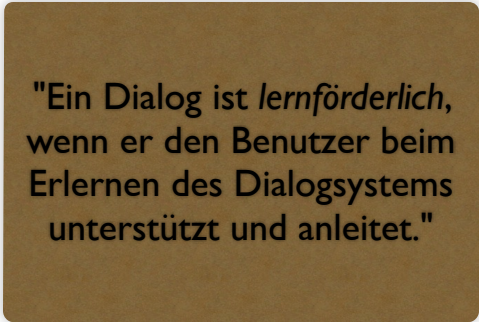
Individualisierbarkeit

Wenn jeder arbeitet, wie es ihm beliebt

- In einem personalisierten Web-Portal kann man festlegen, welche Fenster wo angezeigt werden.
- Ein editierbares Profil ermöglicht es anzugeben, welche News man lesen möchte.
- Ein Online-Shop erkennt Kunden wieder und füllt Formularfelder selbständig aus.
- Auf einer Startseite kann man zwischen Mobile- oder Desktop-Version wählen und bookmarken.

Lernförderlichkeit

Wenn man schlauer wird



"Ein Dialog ist *lernförderlich*, wenn er den Benutzer beim Erlernen des Dialogsystems unterstützt und anleitet."

Lernförderlichkeit

Wenn man schlauer wird

- In einer "Guided Tour" werden die Benutzer mit besonderen Tricks in der Bedienung einer Applikation vertraut gemacht.
- Im Buchungs-System eines Reiseanbieters besteht die Möglichkeit einer Probebuchung.
- In einer Sitemap kann man sich ansehen, nach welcher Logik eine Website strukturiert ist.

Konsistenz und Kompetenz steigern

- Objekt-Aktivierung und -Bearbeitung *einheitlich, übersichtlich und durchschaubar* darstellen. Wenig syntaktische Fehler zulassen.
- Nur *im Kontext anwendbare Funktionen* darstellen; sinnlose Funktionen blockieren (z.B. grau darstellen)
- *Letzte Parametereinstellung* beim Aufruf einer Menüoption anzeigen (z.B. Formatieren einer Diskette)
- *Sicherheitsabfragen* bei Operationen mit schwerwiegenden Folgen. Folgen verdeutlichen.
- *Undo/Redo*-Funktion anbieten, d.h. alle durchgeführten Operationen sollten storniert werden können.
Auf Wunsch muß die Stornierung wieder aufgehoben werden (*Redo*).
Mindestens einstufig. Wunsch mehrstufig.

Lernförderlichkeit

Konsistenz und Kompetenz steigern (2)

- *Inkrementelle Aufgabenbearbeitung* ermöglichen, d.h. kleine, unabhängige, nichtsequentielle Teilschritte mit jeweiliger Ergebnismeldung.
Keine Operation darf in einer „Sackgasse“ enden.
- *Rückmeldungen* auf alle Benutzeroperationen. Anzeigen, ob Eingabe erwartet wird oder gerade eine Verarbeitung stattfindet. Überdurchschnittliche Verarbeitungszeiten anzeigen (Art, Objekt, Umfang oder Dauer).
Systembedingte Verzögerungen, Unterbrechungen oder Störungen explizit anzeigen.

Lernförderlichkeit

Oberflächengestaltung

EN ISO 9241 – Ergonomie der Mensch-System-Interaktion



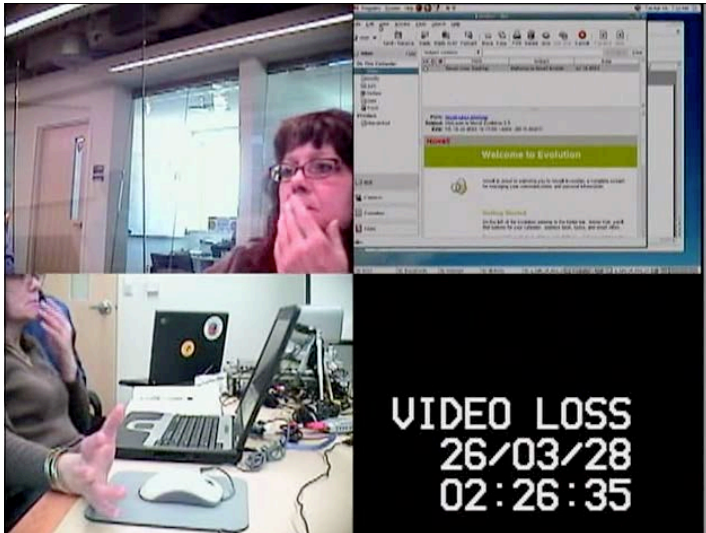
<http://kommdesign.de/texte/din.htm>

Benutzerfreundlichkeit prüfen

Um die Benutzerfreundlichkeit zu prüfen, gibt es nur ein Mittel: *Das System mit echten Benutzern testen!*

Typische Vorgehensweise: Benutzer sollen mit dem System eine bestimmte Aufgabe erledigen – und halten anschließend fest, was sie gestört hat.

Die Benutzer werden dabei (auf Video) aufgezeichnet.



Task: Email A Tale of Two Cities to arthur@ximian.com; Subject14
<http://www.betterdesktop.org/wiki/index.php?title=Data>

Ergebnisse

80% sind erfolgreich

Probleme:

- "Send/Receive"-Schaltfläche ist verwirrend
- Evolution ist schwer als Mail-Anwendung zu finden
- Liste der Anhänge ist versteckt, auch wenn Anhang vorhanden

Benutzerfreundlichkeit prüfen

Beispiel: In Linux-Maschine einloggen (Studie, 2001)

(von Suzanna Smith et al., *GNOME Usability Study Report*,
http://developer.gnome.org/projects/gup/ut1_report/)



Probleme bei der Benutzung dieses Dialogs

- Was ist ein „Login“? Der Benutzername? Oder das Paßwort? (Konsistenz und Kompetenz)
- Username und Paßwort werden nicht gemeinsam abgefragt (Konsistenz und Kompetenz)
- Was soll der Benutzer nach der Eingabe des Namens tun? (Erwartungskonforme Dialoge)
- Bei falschem Benutzernamen/Paßwort erscheint kein Dialog (Erwartungskonforme Dialoge)
- Ist Klein-/Großschreibung relevant? Was passiert, wenn die Caps-Lock-Taste gedrückt ist? (selbsterklärende Dialoge)
- Was ist „marge-hci“? (selbsterklärende Dialoge)

Vorschlag für (leicht) verbesserten Dialog



Typically, when project managers observe their design undergoing a usability test, their initial reaction is:

Where did you find such stupid users?



Checkliste: Benutzungsoberfläche

Ist die Oberfläche konsistent gestaltet?

Dies wird in der Regel durch das Verwenden von Standard-Dialogen erreicht. Wo immer möglich, sollte man sich an Standards orientieren – Dialoggestaltung, Menüstruktur, Tastaturbelegung usw.

Sind die Dialoge selbsterklärend?

Sind die Dialoge erwartungskonform?

Auch dies wird durch Standard-Dialoge erreicht.

Checkliste: Benutzungsoberfläche (2)

Sind die Dialoge fehlerrobust?

Fehleingaben dürfen in keinem Fall zu Abstürzen oder undefiniertem Verhalten führen.

Meldungen müssen positiv sein. Keine Anklagen! Keine Beleidigungen! Keine Witze! Keine merkwürdigen Geräusche auf dem Lautsprecher!

Stets sollte das System vermitteln, daß es zu dumm ist, den Benutzer zu verstehen, und nicht, daß der Benutzer zu dumm ist, das System zu bedienen.

Checkliste: Benutzungsoberfläche (3)

Ist die Anwendung flexibel?

Nach Absprache mit dem Auftraggeber sollte die Anwendung möglichst so gestaltet werden, daß Aufgaben auf alternativen Wegen ausgeführt werden können.

Modale Dialoge sollten, wo immer möglich, vermieden werden.

Unterstützt die Oberfläche effizientes Arbeiten?

Das entscheidet der Auftraggeber.
