

Usability Engineering

Kapitel 3

Usability Engineering - Lebenszyklus

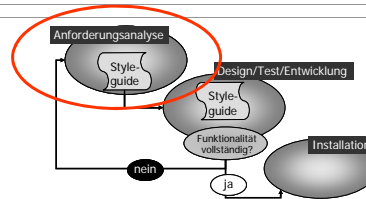
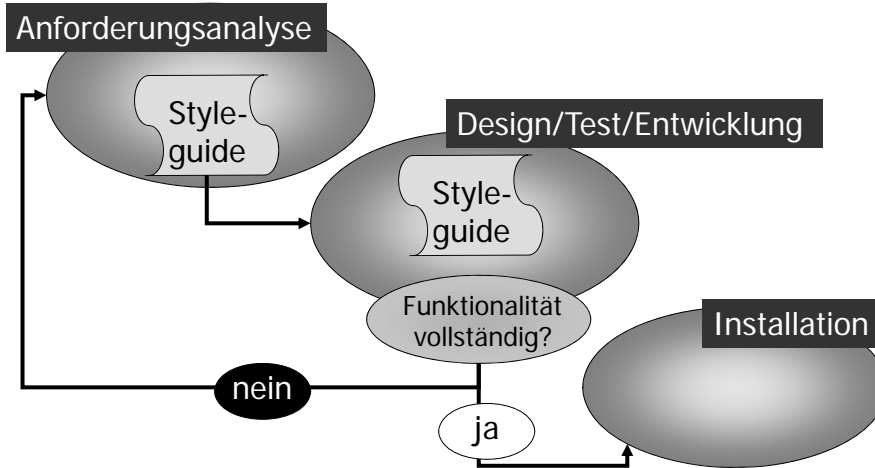
- 1 Usability Engineering – Einführung
- 2 Wahrnehmungspsychologie
- 3 *Usability Engineering – Lebenszyklus*
 - 3.1 *Anforderungsanalyse*
 - 3.1.1 *Nutzerprofile*
 - 3.1.2 *Aufgabenanalyse*
 - 3.1.3 *Plattform-Potentiale und -Beschränkungen*
 - 3.1.4 *Allgemeine Design-Prinzipien*
 - 3.1.5 *Usability-Ziele*
 - 3.2 *Design, Test und Entwicklung*
 - 3.2.1 *Niveau 1*
 - 3.2.1.1 *Re-engineering der Arbeitsabläufe*
 - 3.2.1.2 *Design des Konzeptuellen Modells*
 - 3.2.1.3 *Mock-ups des Konzeptuellen Modells*
 - 3.2.1.4 *Iterative Evaluierung des Konzeptionellen Modells*

- 3.2.2 *Niveau 2*
 - 3.2.2.1 *Design der Schnittstellenstandards*
 - 3.2.2.2 *Prototyping der Schnittstellenstandards*
 - 3.2.2.3 *Iterative der Evaluierung der Schnittstellenstandards*
 - 3.2.2.4 *Styleguide*
- 3.2.3 *Niveau 3*
 - 3.2.3.1 *Design konkreter Interfaces*
 - 3.2.3.2 *Iterative Evaluierung konkreter Interfaces*
- 3.3 *Installation und Nutzer-Feedback*
- 4 Methoden der Usabilityevaluation
- 5 Web Usability
- 6 Barrierefreiheit

Usability Engineering wird in drei Phasen implementiert:

- 3.1 Anforderungsanalyse
- 3.2 Design, Test und Entwicklung
- 3.3 Installation

Usability Engineering begleitet die Produktentwicklung durch ihren gesamten Lebenszyklus.



Aufgabe 1: Nutzerprofile (user profiles)

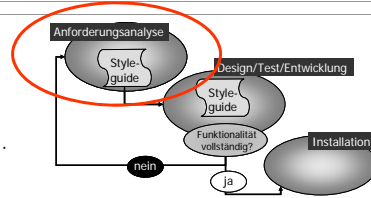
Aufgabe 2: Aufgabenkontext-Analyse (contextual task analysis).

Aufgabe 3: Definition der Usability-Ziele (usability goal setting).
Spezifizierung qualitativer und quantitativer Ziele.

Aufgabe 4: Plattform-Potentiale und -beschränkungen (platform capabilities and constraints).

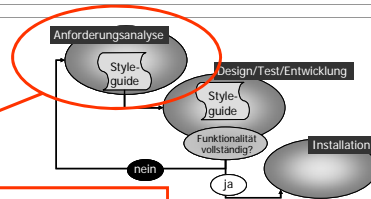
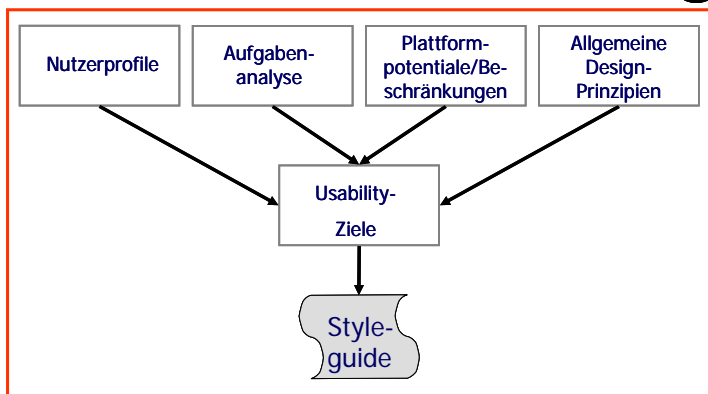
3.1 Anforderungsanalyse

- Die Ergebnisse aus den vier Aufgaben werden im Produkt-Styleguide dokumentiert.
- Hier werden auch relevante, allgemeine Design-Prinzipien für die Gestaltung der Nutzerschnittstellen diskutiert.
- Sie finden Anwendung im späteren Designprozess zusammen mit allen produkt-spezifischen Informationen, die in diesen ersten vier Aufgaben ermittelt wurden.



3.1 Anforderungsanalyse

Umsetzungsmodul der Anforderungsanalyse



- **Solange die spezifischen Eigenschaften unserer Nutzer nicht bekannt sind** (Nutzungshäufigkeit der Software, Tip-Geschwindigkeit etc.), **können keine optimalen Entscheidungen über die Interface-Design getroffen werden.**
- ein Nutzerprofil spiegelt die allgemeinen Eigenschaften einer Kategorie von Nutzern wieder in Bezug auf den allgemeinen Stil eines Interfaces
- die einzelnen Nutzerprofile werden zusammengefasst zum Gesamtprofil der Nutzergruppe, das Auskunft über folgende Eigenschaften gibt:
 - **psychologische Eigenschaften** (Einstellung, Motivation, ...)
 - **Wissen und Erfahrung** (Arbeitserfahrung, Geschicklichkeit, ...)
 - **Eigenschaften der Arbeit** (Nutzungshäufigkeit, Struktur der Arbeit, ...)
 - **physische Eigenschaften** (Kurzsichtigkeit, Farbenblindheit, ...)

- Wie werden die Profile erhoben?
 - durch Fragebögen
 - von der Marketing-Abteilung
 - von der Personalabteilung
- Die Erhebung der Nutzerprofile sollte alle paar Jahre wiederholt werden, da sich die Eigenschaften der individuellen Nutzer und die Zusammensetzung der Nutzergruppen im Laufe der Zeit ändern.
- Input für Aufgabenanalyse:
 - Identifizierung der Nutzerkategorien
- Input für Definition der Usability-Ziele:
 - Ziele werden von Nutzeigenschaften mitbestimmt: unterschiedliche Usability-Ziele für unterschiedliche Nutzer-Kategorien

Schrittweises Vorgehen:

- 1) Nutzerkategorien festlegen
- 2) Relevante Nutzereigenschaften bestimmen
- 3) Erster Entwurf eines Fragebogens
- 4) Diskussion des Fragebogen-Entwurfs mit der Abteilungs-/Geschäfts-/Projektleitung
- 5) Revision des Fragebogens
- 6) Durchführung erster Interviews mit Pilotfragebogen
- 7) Revision des Fragebogens

Schrittweises Vorgehen:

- 8) Auswahl der Nutzergruppe
- 9) Versenden der Fragebögen
- 10) Erfassen des Rücklaufs und Festlegung der Auswertungsmethoden
- 11) Daten-Summary
- 12) Interpretation der Daten
- 13) Präsentation der Ergebnisse

- Aufgabenanalyse bezieht sich auf Projekte, die auf die IT-Unterstützung bestimmter Arbeitsschritte zielen, d.h. wenn bereits ein bestimmtes Set an Funktionen und Features skizziert wurde.
- Sie dient in erster Linie der Entwicklung eines nutzerzentrierten Modells der Aufgaben und Arbeitsschritte, wie sie derzeit realisiert werden.
- **drei Ziele:**
 - 1) Gewinnung von mehr Effizienz, die Automatisierung ermöglicht.
 - 2) Re-engineering von Arbeitsprozessen zur besseren Unterstützung der Geschäftsziele.
 - 3) Minimierung von Trainingsaufwand, indem die angestrebte Anwendung so nahe wie möglich an existierende Fähigkeiten, Kenntnisse, Fertigkeiten, Gewohnheiten und Entwicklungspotentiale(!) der Nutzer heranreicht.

Grundlegende Schritte:

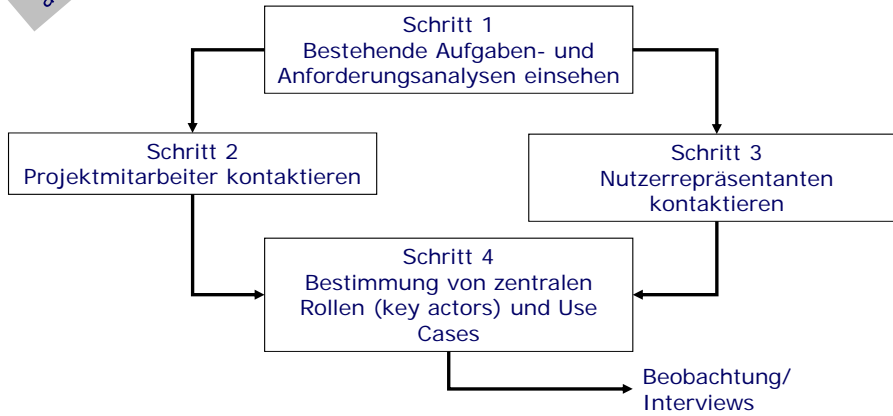
- 1) Sammeln von Information über die Arbeit die automatisiert werden soll bzw. die durch IT unterstützt werden soll.
- 2) Sammlung von Daten durch Beobachtung und Interviews mit realen Nutzern in realen Arbeitsumgebungen
- 3) Entwicklung eines Modells der gegenwärtigen Arbeitsorganisation

Nebeneffekte der Aufgabenanalyse:

- 1) Inspiration für neue Ideen
- 2) wichtige Features, die in die Produkte integriert werden sollten
- 3) Verbesserung von Interfaces für bereits eingesetzte Produkte

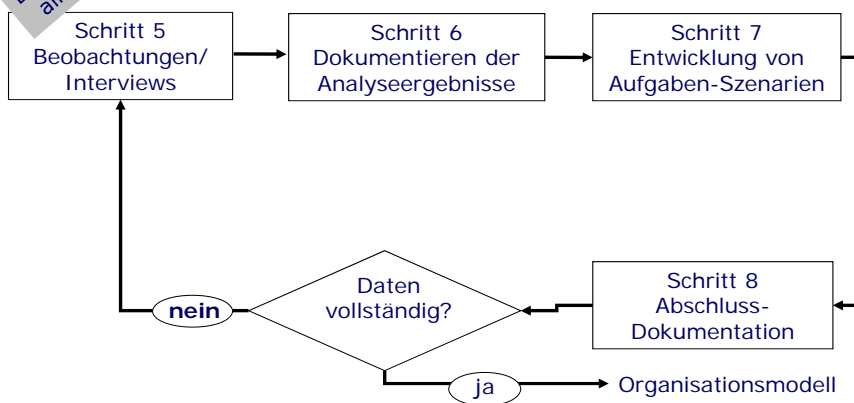
Dokumentation
aller Schritte

Sammeln von Hintergrundinformationen



Dokumentation
aller Schritte

Beobachtung/Interviews



Entwicklung eines Organisationsmodells der aktuellen Nutzeraufgaben (OMAN)

Dokumentation
aller Schritte



- Hard- und Software-Plattformen eröffnen **Potentiale**, führen aber auch zu **Beschränkungen** für das Design der Interfaces.
- Der Aufwandsabschätzung für bestimmte Interface-Entwürfe unter verschiedenen Plattformen.
- Wenn eine bestimmte Plattform vorherrschend ist (z.B. Windows), dann ist Bestimmung der Potentiale und Beschränkungen einfach. Dieser Vorgang muss aber für verschiedene Releases der Plattform wiederholt werden.
- Benutzung unterschiedlicher Entwicklungsplattformen kann zu unterschiedlichen Potentialen und Beschränkungen für die gleiche Plattform-Umgebung führen. Das gleiche gilt für die Integration von Datenbankpaketen und ähnlichem.
- Die Weiterentwicklung der Hardware hat Einfluss auf das Verhalten der Interface-Elemente und der zugrunde liegenden Prozesse.

Schrittweises Vorgehen:

Schritt 1:
Bestimmung aller Interface-relevanten Aspekte der Hard- und Software-Umgebung.

Schritt 2:
Studium der plattformspezifischen Dokumentation.

Schritt 3:
Beratung mit technischem Personal

Schritt 4:
Dokumentation Plattform-Potentiale und -Beschränkungen.

- Bestimmung allgemeiner **Design-Prinzipien**, die bei der aktuellen Entwicklung **sinnvoll Anwendung finden**.
- Allgemeine Prinzipien ersetzen nicht die Analyse produkt-spezifischer Anforderungen und die zyklische (iterative) Usability-Evaluierung.
- Wenn allerdings die Analyse sehr schwierig ist, kann beim ersten Design-Zyklus auf allgemeine Prinzipien verstärkt zurückgegriffen werden.

Vier Prinzipien für ein gutes Interface-Design:

- 1) Gründliche Anforderungsanalyse
- 2) Anwendung allgemeiner Design-Prinzipien (inkl. Guidelines/ Styleguides)
- 3) Annäherung an das Interface-Design durch einen strukturierten Prozess
- 4) Anwendung wiederholter Usability-Testzyklen

Schrittweises Vorgehen

Schritt 1: Studium unternehmensspezifischer Styleguides

Produkt-Styleguide, Plattform-, Produktfamilien-, Corporate-Styleguide

Schritt 2: Studium allgemeiner Styleguides

Bücher, Zeitschriften, Web-Sites und spezielle Studien (Konferenzbände) zu allgemeinen Design-Prinzipien.

Warum Usability-Ziele?

- 1) Sie bilden eine gute Leitlinie für das Design der Interfaces.
 - Ein gemeinsames und zutreffendes Bild der Nutzergruppen (abgeleitet aus den Nutzerprofilen) und ein gemeinsames und zutreffendes Modell der Arbeit und der Arbeitsumgebung (aus der Aufgabenanalyse) helfen, den Design-Prozess besser zu fokussieren.

- 2) Sie dienen als Akzeptanz-Kriterien für die Evaluierung.
 - Die Entscheidung, einen weiteren Design-Zyklus zu durchlaufen oder auf die Interface-Entwicklung überzugehen, ist fundierter.

Usability Ziele festlegen

Qualitative Usability-Ziele

- Qualitative Ziele sind hilfreich, das Interface-Design vor allem in der Anfangsphase zu leiten.
- Beispiele:
 - Das System soll keine Kenntnis der ihr zugrunde-liegenden Technologie erfordern.
 - Beim Übergang zu neuen Releases sollten Änderungen, die für die Aufgaben der Nutzer irrelevant sind, nicht sichtbar sein.
 - Das System soll Gruppenarbeit unterstützen.

Quantitative Usability-Ziele

- Das Erreichen qualitativer Ziele ist oftmals schwer zu präzisieren. Im Gegensatz dazu sind quantitative Ziele objektiver und genauer messbar.

Beispiele:

- Festlegen einer bestimmten oder höchst zulässigen Ausführungszeit.
- Die Ausführungszeiten werden für ein bestimmtes Niveau an Nutzererfahrung festgelegt: Experte: ease-of-use, neuer Nutzer: ease-of-learning.

Quantitative Usability-Ziele

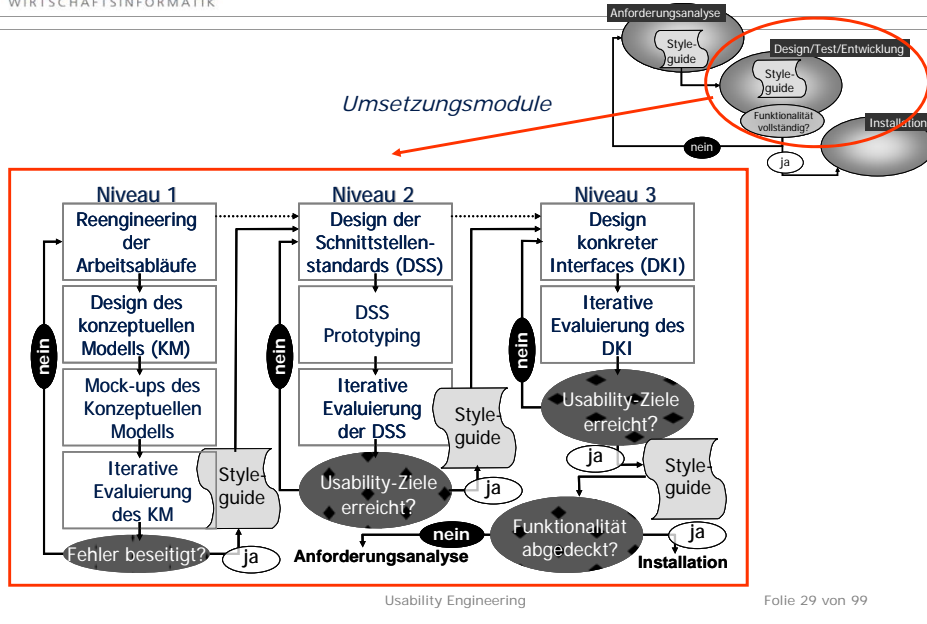
- Absolute Ziele benutzen dabei absolute, quantitative Größen wie Bearbeitungszeit (in Min., Sek.), Anzahl der Fehler etc.
- Relative Ziele beziehen sich auf die Erfahrung der Nutzer mit einem bestimmten Produkt/Interface relativ zu den Erfahrungen mit einem anderen Produkt/ Interface.
- Klare Präferenz zwischen Alternativen.
- Niveau der Zufriedenheit mit einem bestimmten Interface. (5-stufige Skala: unzufrieden ... vollauf zufrieden)
- Performanz-Ziele quantifizieren die aktuelle Performanz eines Nutzers in der Ausführung einer bestimmten Aufgabe. Üblich: Zeit, um die Aufgabe auszuführen bzw. um die Ausführung zu erlernen, Anzahl und Art der Fehler.

- Ziele können sich auf eine einfache Transaktion beziehen oder auf eine komplexere Aufgabe, die mehrere Arbeitsschritte umfasst.
- Ziele können sich auch auf ein gesamtes Arbeitsgebiet beziehen.
- In die Definition der Ziele sollten je nach Anwendungsumgebung die entsprechenden Schlüsselgruppen einbezogen werden: Nutzer, Marketing, Qualitätsmanagement, Entwickler, Support-Abteilung, ...
- Ziele sollten nach Prioritäten geordnet werden.

Usability-Ziele festlegen, schrittweises Vorgehen

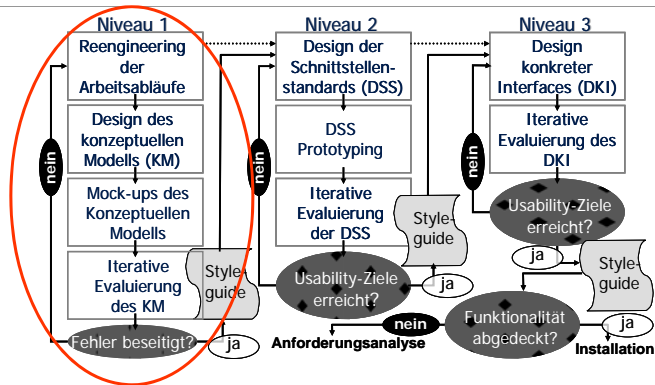
- Schritt 1: Nutzerprofile berücksichtigen.
- Schritt 2: Aufgabenanalyse berücksichtigen.
- Schritt 3: Geschäftsziele berücksichtigen.
- Schritt 4: Qualitative Ziele skizzieren.
- Schritt 5: Prioritäten festlegen.
- Schritt 6: Formulierung der quantitativen Ziele.
- Schritt 7: Ranking der Usability-Ziele
- Schritt 8: Review der Usability-Ziele
- Schritt 9: Benchmark-Daten für relative quantitative Ziele.

3.2 Design, Test und Entwicklung



3.2 Design, Test und Entwicklung

3.2.1 Niveau 1



Phase 2: Design/Test/Entwicklung
Diese Phase wird in drei Niveaus eingeteilt.

Niveau 1 werden Aufgaben zugeordnet, die sich auf übergeordnete Design-Aspekte (high-level design issues) beziehen.

Aufgabe 1: Re-engineering der Arbeitsabläufe (work re-engineering).

Aufgabe 2: Design eines konzeptuellen Modells (conceptual model design).

Aufgabe 3: Skizzen und Attrappen des konzeptuellen Modells (conceptual model mock-ups).

Aufgabe 4: Iterative Evaluierung des konzeptuellen Modells (iterative conceptual model evaluation).

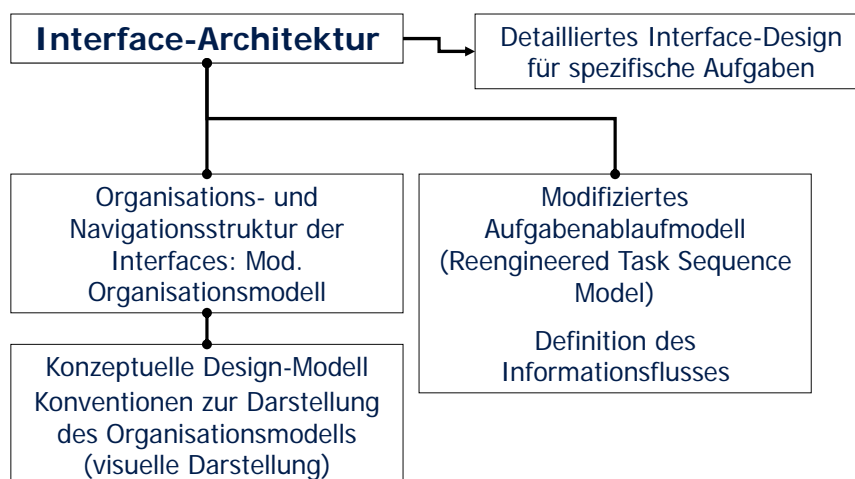
Aufgabe 2 bis 4 werden in iterativen Schritten wiederholt, bis die größten Usability-Bugs eliminiert sind.

Das Ergebnis ist ein relativ stabiles konzeptuelles Modell.

Anforderungsanalyse → Aufgabenanalyse:

- Organisationsmodelle der Arbeiten, für die IT-Unterstützung (Automatisierung) erarbeitet werden soll.
- Modelle spiegeln wider, wie die Nutzergruppe über diese Arbeiten denkt, redet bzw. sie durchführt.
- Use Cases (aus konsolidierten Aufgaben-Szenarios)
- **Wichtig ist das Verständnis der Arbeitspraktiken!**

- **Der erste Schritt im Design-Prozess ist das Re-engineering dieser Modelle, um**
 - 1) das Potential der IT-Unterstützung zu nutzen,
 - 1) die Geschäftsziele besser zu unterstützen,
 - 2) den zusätzlichen Trainingsaufwand zu minimieren, den die neue Anwendung erfordert und um
 - 3) eine bessere Arbeitseffizienz zu erreichen.
- Usability-Ziele leiten das Reengineering der Arbeitsmodelle.
- **Das modifizierte Organisationsmodell der Nutzeraufgaben (reengineered task organization model) bildet die Grundlage für die Interface-Architektur.**



Grundlegende Schritte:

- 1) Re-engineering des aktuellen Organisationsmodells, der Use Cases und Aufgaben-Szenarien
- 2) Validierung und Anpassung des modifizierten Organisationsmodells
- 3) Dokumentation des modifizierten Organisationsmodells und des modifizierten Aufgabenabfolgemodell (Teil des Produkt-Styleguides).

- Das modifizierte Organisationsmodell bildet die Grundlage (Struktur und Organisation) der Interface-Architektur.
- Die Interface-Architektur schließt auch Konventionen ein, die die Präsentation dieser Struktur umfassen (Design des Konzeptuellen Modells und Screen-Design-Standards).
- Das Design konkreter Interfaces wird von der Interface-Architektur geleitet.

Konzeptuelles Modell:

- Reihe von Präsentationsgrundsätzen für die konsistente Darstellung funktionaler Komponenten.
- Re-engineering ist nicht Design, sondern die Organisation von Funktionalität.
- Das erste Design des Konzeptuellen Modells gibt in wenigen und knappen, aber repräsentativen Teilen, die gesamte Produktfunktionalität wieder.
- Zunächst noch keine Verbindung zu Inhalten, sondern nur Screen-Layout und -Design.

Schrittweises Vorgehen

Schritt 1: Konzeptuelles Modell: produkt- oder prozessorientiert

Exkurs:

Ein Produkt-Interface repräsentiert:

- 1) **Produkte**, die von den Nutzern erzeugt werden
- 2) **Tools**, die die Nutzer bei der Erzeugung der Produkte unterstützen
- 3) **Aktionen**, die die Nutzer bei der Erzeugung von Produkten verwenden

Primärprodukte, Sekundärprodukte

- Primärprodukte sind der eigentliche Zweck der Anwendung (also bspw. Tabellen). Sekundärprodukte sind von den Nutzern erzeugte Tools zur Produkterzeugung.

Konvention:

- Primärprodukte können als eigenständige Icons wiedergegeben werden, Sekundärprodukte werden in Menüeinträgen oder Dialogboxen aufgelistet.
- Die Beachtung dieser Konvention stellt sicher, dass das angestrebte Konzeptuelle Modell mit denen verbreiteter GUI-Plattformen einhergeht (mentales Modell).
- Der Fokus soll auf das Primärprodukt gerichtet werden und die adäquaten Sekundärprodukte werden darum herum angeordnet.

Schritt 2: Bestimmung von Produkten oder Prozessen

Schritt 3: Darstellungsgrundsätze für Produkte und Prozesse

Schritt 4: Gestaltungsregeln für Fenster

Schritt 5: Festlegung der wichtigsten Fenster

Schritt 6: Definition und Design wichtiger Navigationspfade

Schritt 7: Dokumentation alternativer Konzeptueller Modelle in Skizzen und Erläuterungen

- **Mock-ups** (Attrappen) unterstützen die **formale Evaluierung** des Konzeptuellen Modells, die in der folgenden Aufgabe vorgesehen ist.
- Mock-ups geben nur kleine, aber repräsentative Teile der gesamten Produkt-Funktionalität wieder.
- Es werden Teile der Menüleiste einbezogen, um Navigationspfade zu verdeutlichen.
- Von den vorangegangenen Skizzen werden zwei oder drei ausgewählt und in Mock-ups umgesetzt.
- Die Mock-ups können als laufende Anwendungen implementiert werden („high-fidelity“ mock-ups) oder aber auch als Folge von Screen-Bildern („low-fidelity mock-ups“).

Schritt 1: Auswahl der Funktionalität

Festlegung einer bestimmten Menge an Funktionen und Features, die in Mock-ups dargestellt werden sollen.

Schritt 2: Entwurf der Mock-ups

Die Entwürfe enthalten nur so viele Details wie für das allgemeine Verständnis der Navigation durch das Interface notwendig ist. Erklärungstexte helfen, den Zweck der jeweiligen Fenster zu erläutern.

Schritt 3: Implementierung der Mock-ups

- **Schnelles und frühes Feedback** über den Grad an Usability des Konzeptuellen Modells, dargestellt anhand der Mock-ups.
- Hier werden formale Techniken angewendet, die effektiver und objektiver in der Auswertung sind als das subjektive Feedback der Nutzer während einer Demo (formal usability testing).

Testgruppen

- Vertreter der verschiedenen Nutzergruppen führen realistische Testaufgaben an ihrem Arbeitsplatz durch.
- Um auf **ease-of-learning** zu testen, erhalten die Nutzer nur ganz wenige Instruktionen (1-/2-seitiges „Handbuch“).
- Bei **ease-of-use** durchlaufen die Nutzer zunächst ein Training, damit sie mit dem erworbenen Wissen Expertenstatus reflektieren und dann die Testaufgaben durchführen.

Die **Evaluierung umfasst folgende Schritte:**

- 1) Test-Planung und –Vorbereitung
- 2) Testdurchführung und Sammlung der Analysedaten
- 3) Datenauswertung und Redesign-Empfehlungen

Am Ende jedes Test werden modifiziert:

- Design des Konzeptuellen Modells
- Mock-ups
- Testplanung

Testvorbereitung

- In die Vorbereitung sollten alle Personen einbezogen werden, die an der Produkt-/Anwendungsgestaltung beteiligt sind (Entwickler, Designer, Projektmanager, Marketing-Abteilung, ...).

Schritt 1: Fokus: ease-of-learning oder ease-of-use

- Ease-of-learning: vorrangig neue Nutzer, geringes Vorbereitungstraining.
- Ease-of-use: intensiveres Vorbereitungstraining, um Expertenstatus zu simulieren.

Schritt 2: Fokus: Nutzergruppen und Aufgaben

- Es wird der Umfang der Nutzer festgelegt, die einbezogen werden: eine oder mehrere Nutzerkategorien.
- Es werden Testaufgaben gewählt, die eine zentrale Rolle spielen werden bzw. oft durchgeführt werden.

Schritt 3: Festlegung der Testaufgaben

- Aus den Aufgaben-Szenarien (Aufgabenanalyse) werden geeignete Testaufgaben abgeleitet und beschrieben.

Ergebnis-basierte Aufgaben oder prozess-basierte Aufgaben?

Schritt 4: Gestaltung von Test und Begleitmaterial

Folgende **Phasen** sind dabei zu berücksichtigen:

- **Beobachter-Briefing.** Instruktionen für Designer, Entwickler usw., die die Tests beobachten.
 - **Begrüßung.**
 - **Einführung.** Erklärung, wie die Tests durchgeführt werden und was von den Testnutzern erwartet wird. Betonung, dass das Interface getestet werden soll und nicht die Nutzer.
 - **Pre-Test-Fragebogen.** Erfassung spezifischer Nutzereigenschaften: Aufgabengebiet, Erfahrung, etc.
 - **Training.** Trainingsaufwand abhängig von Test auf ease-of-learning oder ease-of-use

- **Erlaubnis der Videoaufzeichnung.**
- **Testaufgaben**, die an die Nutzer ausgehändigt werden.
- **Analyseblätter.** Fehler und andere problemrelevante Beobachtungen werden festgehalten.
- **Post-Test-Fragebogen.** Sammlung subjektiver Eindrücke der Nutzer.
- **Zusammenfassung der Datenanalyse.** Ziel der Evaluierung ist es, grundlegende und auffällige Design-Fehler zu entdecken.

Schritt 5: Testumgebung

- Für typische Büroanwendungen können die Tests in Usability-Labs durchgeführt werden.
- Wenn keine Labs verfügbar sind, kann auch ein „typischer Büroraum“ eingerichtet werden.
- Für andere Umgebungen sollte ein realistischer Arbeitsplatz gewählt werden.
- Anordnung der Videokameras, der Beobachter (im Hintergrund). Verfügbarkeit brauchbarer Daten.
- Zusätzliches Hilfspersonal für die Tests (Kamera, Post-Test-Fragebögen etc.)

Schritt 6: Pilot-Testnutzer

- Repräsentative Nutzer der aktuellen Nutzer.

Schritt 7: Pilot-Test

- Pilot-Testnutzer führen die Tests durch und „debuggen“ eventuelle Fehler im Test und den Begleitmaterialien.
- Problembereiche, die die Testnutzer testen:
 - Prototyp funktioniert und erforderliche Daten sind vorhanden
 - Support-Material ist verfügbar.
 - Beobachtung/Datensammlung durchführbar.
 - Alle Aufgaben können in der vorgesehenen Zeit erledigt werden.
 - Ausreichend Zeit für den Neustart der Mock-ups ist vorhanden.
 - Alle Beobachter verstehen die Grundregeln für die Beobachtung.

Schritt 8: Revision der Testprozeduren und Materialien

als Ergebnis aus Schritt 7.

Schritt 9: Testnutzer/Testagenda

- Entsprechend der Beschreibungen der Nutzer aus Schritt 2 werden repräsentative Testnutzer aus verschiedenen Nutzer-Kategorien ausgewählt.
- In den Zeitablauf sind nicht nur die Tests selbst mit allen ihren Vor- und Nachbereitungen einzuplanen, sondern auch Pausen, evtl. Verspätungen etc.

Durchführung der Tests

Schritt 1: Testdurchführung und Datensammlung

Die Testphasen werden wie geplant durchlaufen.

- Begrüßung
- Erhebung der Pre-Test-Informationen (Fragebogen). Diese Daten dienen dazu Nutzer-Eigenschaften zu dokumentieren, die helfen, das Nutzerverhalten während der Testphase zu erklären.
- Vorstellung der Tests
- Für ease-to-learning-Tests werden den Nutzern die Trainingsunterlagen ausgehändigt.
- Bei ease-to-use-Tests wird das Training für jeden Nutzer individuell durchgeführt mit unmittelbaren praktischen Beispielen.
- Es ist sicherzustellen, dass alle Nutzer alle Mock-ups durchlaufen; die Reihenfolge muss dabei von Nutzer zu Nutzer gewechselt werden.

- Vorstellung der ersten Testaufgabe
- Es ist wichtig, während der Tests die Nutzer nicht zu führen oder ihnen wichtige Informationen über die Handhabung der Interfaces zu geben.
- Die Nutzer werden angeregt, laut zu denken während der Durchführung ihrer Arbeiten. Die Entdeckung von Fehlern und Arbeitsschritten, die Verwirrung stiften, steht hier im Vordergrund.
- Festhalten der Daten auf den vorbereiteten Datenblättern.
- Festhalten der Tests auf Video ist nicht notwendig, kann aber hilfreich sein als „Testdaten-Backup“, oder für Präsentationszwecke.
- Alle Arbeiten an einem Mock-up sind abzuschließen, bevor mit dem nächsten fortgefahren werden kann.
- Nach den Tests werden die Nutzer über ihre subjektiven Eindrücke und Präferenzen befragt (Post-Test-Fragebogen).

Schritt 2: Zusammenfassung der Daten

Zusammenfassen der Daten, wie Anzahl und Art der Fehler pro Aufgabe und Nutzer-Kommentare, die sich auf Fehler beziehen.

- Wie oft trat ein bestimmtes Problem auf?
- Wie viel Zeit wurde auf Fehler/auf produktive Arbeit verwendet?
- Anzahl der Nutzer, bei denen ein bestimmtes Problem auftrat.
- Anzahl der Fehler aller Art für eine bestimmte Aufgabe

Schritt 3: Datenanalyse und -interpretation

Fokus auf die Daten, die ein Problem anzeigen.

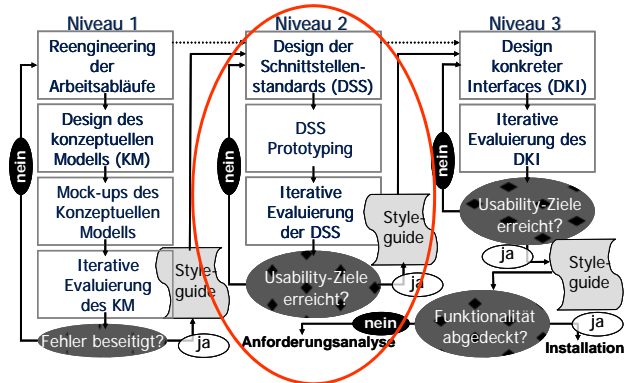
Schritt 4: Schlussfolgerungen und Änderungsempfehlungen

Es sind Schlussfolgerungen aus den festgestellten Problembereichen zu ziehen und Empfehlungen zu deren Beseitigung abzuleiten.

Schritt 5: Dokumentation der Ergebnisse

Reports und Videos, die die Ergebnisse dokumentieren.

- Reports: Zusammenfassung (welche Fehler traten auf) und detaillierter Teil (Daten zu Fehlern, Daten-Interpretationen und empfohlenen Problemlösungen).



Niveau 2 beschäftigt sich mit der Entwicklung produkt-bezogener Standards.

Aufgabe 1: Standards für das Design der Schnittstellen (screen design standards)

Aufgabe 2: Entwicklung von Prototypen für die Schnittstellenstandards (screen design standards prototyping)

Aufgabe 3: Iterative Evaluierung der Schnittstellenstandards (iterative screen design standards evaluation)

Aufgabe 4: Produkt-bezogener Styleguide (style guide development)

Wie schon für Niveau 1 werden die Design/ Evaluierungs-Zyklen aus Niveau 2 mit der Dokumentation der Schnittstellenstandards im produkt-bezogenen Styleguide beendet.

Er enthält ein stabiles Design des konzeptuellen Modells und ein stabile Standards und Konventionen für die Interfaces.

Fortschreibung des Styleguides aus der Anforderungsanalyse.

- **Schnittstellen-Standards** (screen design standards) stellen Konsistenz und Einfachheit bei der späteren Gestaltung der konkreten Benutzeroberfläche sicher.
- Konsistenz unterstützt dabei ease-of-learning und ease-of-use gleichermaßen.
- Standards stellen die Qualität sicher, wenn sie auf einer genauen Nutzer- und Anforderungsanalyse, auf Usability-Zielen und allgemeinen Design-Prinzipien basieren.
- Standards unterstützen die **Wiederverwendbarkeit** von Programmteilen und vermeiden die redundante Erstellung von Schnittstellen-Elementen.
- Schnittstellen-Standards sind bei der späteren Gestaltung der Interaktionselemente und Screens zu beachten und einzuhalten.
- Die Standards sind eine **Kombination aus Plattform-Standards und produkt-spezifischen Standards**.

Standards beziehen sich auf:

- Gestaltung von Steuerelementen (Check-Boxen, Buttons, Combo-Boxen etc.)
- Ort und Darstellung von Standard-Komponenten (Status-Zeile, Titelleiste, Navigationselementen etc.)
- Terminologie
- Einsatz von Farbe und Schriften
- Einsatz von Maus und Short-Cuts
- Gestaltung von Nachrichten (Message-Windows)

Schritt 1 - Entwurf von Standards für Steuerelemente.

- (Ja/nein-Auswahl nur mit einem Steuerelementtyp z.B.)
- Sobald die Nutzer mit den Standards der Steuerelemente vertraut sind, können sie schnell und effizient die Benutzerschnittstellen bedienen.

Schritt 2 - Entwurf von Standards für Fenster für Produkte/Prozesse.

- Standards für das Layout dieser Fenster (einschließlich der jeweiligen Steuerelemente, Titel): Fenster-Templates.

Schritt 3 - Entwurf von Standards für Dialogboxen

- Standards für das konsistente Design des Inhalts der Dialogboxen (Position der Buttons, Design editierbarer und nicht-editierbarer Felder, optionaler und zwingender Eingaben).

Schritt 4 - Entwurf von Standards für Nachrichtenfenster

- Unterscheidung von Fehlermeldungen, Warnungen und Statusmeldungen. Konsistentes Format und Vokabular, einheitliche Positionierung der Buttons.

Schritt 5 - Entwurf von Standards für Maus-und Keyboard-Interaktionen

- Standards für das konsistente Design von Short-Cuts, Anzahl der Mausclicks, Bedeutung der rechten Maustaste etc.

Schritt 6 - Entwurf von Standards für Systemmeldungen

- Standards für das Feedback, das die Anwendung liefert auf Ereignisse wie Abschluss einer Aufgabe, Auswahl, Zustand laufender Prozesse etc. Visuelle Hinweiselemente.

Schritt 7 - Dokumentation der entworfenen Standards

- Alle Standards sind mit Bild und Text zu beschreiben.
- Im frühen Design-Stadium kann eine handschriftliche Dokumentation hilfreich sein (zahlreiche Änderungen).
- Die endgültigen Standards sollten aber detaillierter (formaler) dokumentiert werden.

- Prototypen unterstützen die Evaluierung der Schnittstellen-Standards
- Die Evaluierung von Design-Ideen auf abstraktem Niveau durch die Nutzer unterstützt das Verstehen und die Berücksichtigung der Nutzerbedürfnisse.
- Evaluation-Feedback nicht nur für die Schnittstellen-Standards, sondern auch für die prototypische Darstellung der hier verwendeten Funktionalität.
- Erweitert die Evaluierung des Konzeptuellen Modells um neue Funktionalität.

- Zunächst wird ein **Subset der** gesamten Produkt-**Funktionalität** für das Prototyping ausgewählt:
 - Kleinster Umfang an Funktionalität, der für den Test (möglichst) aller Standards notwendig ist.
 - Im Gegensatz zu den Mock-ups werden hier die vollständigen funktionalen und Interface-Details für das Subset an Funktionalität definiert.
 - Low-fidelity oder high-fidelity Prototypen (vollständige Interaktion, aber nicht vollständige Funktionalität mit Datenbank-Anbindung etc.)

Prototyping - Schritt 1

Auswahl der Funktionalität (Untermenge), die im Prototypen einbezogen werden soll.

- Funktionen, die eine möglichst große Anzahl von Schnittstellenstandards testen.
- Funktionen, die als essentiell betrachtet werden.
- Funktionen, deren Interfaces problematisch sein werden.
- Funktionen, die als repräsentativ für die gesamte Funktionalität gelten.
- Funktionen, die in einer Funktionsfolge angehören.

Prototyping - Schritt 2

Skizzierung der Interfaces zu der ausgewählten Untermenge der Produktfunktionalität. Die Skizzen basieren auf dem Konzeptuellen Modell und den entwickelten Schnittstellen-Standards.

- Fenster und Dialogboxen
- Steuerelemente wie Buttons, Menüleisten usw.
- Nutzer-Interaktionen und Pfade durch die Interface-Struktur entsprechend der gewählten Funktionalität
- Messages

Prototyping - Schritt 3

Implementierung des Prototypen (low-fidelity vs. high-fidelity)

Der Zweck der Evaluierung ist auch hier wieder, ein schnelles und frühes Feedback über die Usability der entwickelten Prototypen zu bekommen (Investitionskosten, einfache Anpassungen).

- Test und Evaluierung von Schnittstellen-Standards und einer Untermenge der Produkt-Funktionalität
- Test mit lauffähigen, detaillierten und interaktiven Prototypen.
- Testaufgaben sind strukturierter, detaillierter und spezifischer als in vorangegangenen Tests.
- größerer Fokus auf das Zeitverhalten

Zunächst wieder Fokus auf ease-of-learning oder ease-of-use (abhängig von den Usability-Zielen).

Formales Testen: 3 bis 10 repräsentative Nutzer testen zentrale, häufig durchgeführte und realistische Aufgaben in ihrer Arbeitsumgebung (sofern möglich).

Ease-of-learning: Minimale Anweisungen, knappes Handbuch.

Ease-of-use: kurzes Einführungstraining

Videoaufnahmen von Vorteil sowie „laut Denken“

Die iterative Evaluierung durchläuft folgende Schritte:

- Testplanung und Entwicklung von Begleitmaterial
- Testdurchführung und Datenerfassung
- Analyse und Interpretation der Daten und Formulierung von Empfehlungen für das Redesign.

Am Ende jedes Tests werden modifiziert:

- Screen Design Standards,
- der Prototyp und
- Testplan und Begleitmaterial

Die iterative Evaluierung ist beendet, wenn größere Design-Probleme beseitigt und die quantitativen Usability-Ziele erreichbar erscheinen.

Testvorbereitung

Schritt 1

- Entscheidung für den Testfokus „ease-of-learning“ oder „ease-of-use“.

Schritt 2

- Aufgabenfokus, Nutzergruppen.
- Grundsätzlich sollte vermieden werden, auf Testnutzer zurückzugreifen, die bereits in vorangegangenen Tests dabei waren: Vorhandene Grundkenntnisse des Designs verfälschen das Ergebnis; breite Streuung der Testnutzer ist von Vorteil.

Schritt 3

- Entwurf der (realistischen!) Testaufgaben, abgeleitet aus Aufgabenszenarien, evtl. mit Hilfe eines erfahrenen Endnutzers.

Schritt 4

- Erstellung von Begleitmaterial (Briefing, Fragebögen, Training, Aufgabenbeschreibung)

Schritt 5

- Testumgebung: möglichst realitätsnahe Umgebung, am besten Originalumgebung

Schritt 6

- Bestimmung repräsentativer Pilot-Testnutzer (möglichst unterschiedlich zu vorangegangenen Testnutzern).

Schritt 7

- Pilottest: „Debuggen“ von Test und Begleitmaterial

Schritt 8

- Revision von Testplan und Begleitmaterial

Schritt 9

- Terminierung der Tests und Testnutzer (entsprechend Schritt 2).

Testdurchführung

Schritt 1

- Durchführung der Tests und Datenerfassung.
- Begrüßung, Pre-Test-Fragebogen, Vorstellung des Tests, Training und Präsentation der Testaufgaben.
- Keine Führung der Nutzer durch die Testaufgaben!
- Lautes Denken (evtl. Nachfragen) und Videoaufzeichnung von Vorteil, Erfassen von Daten über das Zeitverhalten.
- Festhalten der Daten in den Datenerfassungsbögen.
- Post-Test-Fragebogen.

Schritt 2

- Zusammenfassung der Daten (Anzahl der Fehler pro Aufgabe, Art der Fehler, Nutzerkommentare etc.)

Schritt 3

- Analyse und Interpretation der Daten. Fokus auf Datenlage, die bestimmte Probleme anzeigen (Häufung von Fehlern, auffällig lange Ausführungszeiten etc.).
- Probleme können sich auf die Schnittstellen-Standards oder aber auf das Konzeptuelle Modell beziehen.

Schritt 4

- Schlussfolgerungen und Redesign-Empfehlungen

Schritt 5

- Dokumentation der Testergebnisse (Executive summary, Darstellung der Probleme und Änderungsempfehlungen).

Styleguide

- Dokument, das die Ergebnisse der Anforderungsanalyse und des Nutzer-Interface-Designs zusammenführt.

Vorteile eines Styleguides:

- Projekte mit langen Entwicklungszyklen laufen Gefahr, einen strengen Fokus auf die Usability-Ziele zu verlieren. Informationen aus frühen Phasen kann verloren gehen.
- Bei komplexen Projekten verschafft der Styleguide den nötigen Überblick über die Informationen, die für das Interface-Design benötigt werden.
- Kommunikationsmedium bei großen Entwicklerteams
- Synergie-Effekt Ergebnisse eines Projekts können auf andere Projekte mit ähnlichen Aufgabenstellungen und Nutzergruppen übertragen werden.

Struktur eines Styleguides

- Einführung
- Überblick über die Produktfunktionalität
- Nutzerprofile
- Aufgabenanalyse
- Plattform-Potentiale und -Beschränkungen
- Usability-Ziele
- modifizierte Aufgabenmodelle
- Konzeptuelles Modell
- Schnittstellen-Standards
- Feedback

Erstellen eines Styleguides

Schritt 1

- Dokumentation der Ergebnisse aus der Anforderungsanalyse.

Schritt 2 (Validierung des Konzeptuellen Modells)

- Das Konzeptuelle Modell kann auf Plattform-, Unternehmens- und Produktfamilien-Styleguides basieren.
- Der Produkt-Styleguide nennt produkt-spezifische Besonderheiten und sollte sich auf die vorgenannten Styleguides beziehen.

Schritt 3 (Validierung der Schnittstellen-Standards)

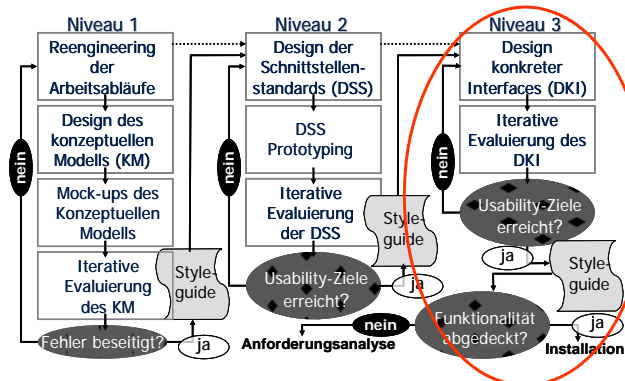
- Die Schnittstellen-Standards können ebenfalls auf den vorgenannten Styleguides basieren.

Schritt 4 (Verbreitung des Styleguides)

- Der Styleguide muss so gestaltet sein, dass er für alle Entwickler verständlich ist.
- Am besten ist es, wenn Designer und Entwickler eine bei der Erstellung der Standards zusammenarbeiten.

Schritt 5 (Nutzung des Styleguides)

- Selbst wenn der Styleguide als Kommunikationsmittel für Designer und Entwickler eingesetzt wird, ist noch nicht gesagt, dass sich alle am Design beteiligten auch danach richten.



Niveau 3: Entwicklung konkreter Interfaces (basierend auf den Ergebnissen von Niveau 1 und 2)

Aufgabe 1: Design konkreter Interfaces (detailed user interface design)

Aufgabe 2: Iterative Evaluierung des konkreten Interface-Design (iterative detailed user interface design evaluation)

- Die Entwicklung der konkreter Interfaces in all ihren Details ist das letztendliche Ziel des Usability-Lebenszyklus.
- Alle vorangegangenen Aufgaben dienen vor allem dazu, diese Entwicklung so effizient und effektiv wie möglich zu gestalten.
- Ziel ist ja eine **umfassende Entwicklung der Interfaces, die Performanz und Zufriedenheit der Nutzer optimiert, wobei ein kosteneffizienter Entwicklungsprozess** beschrrieben wird.
- Die konkreten Interfaces basieren auf dem Design des Konzeptuellen Modells und den Schnittstellen-Standards.
- Die Gestaltung aller Navigationswege, Fenster und Interaktionen folgt die Interface-Grundlagen und den Standards, die in den vorangegangenen Schritten entwickelt wurden.

Das Konzeptuelle Modell beinhaltet:

- Entscheidung über produkt- oder prozess-orientiertem Ansatz
- Bestimmung von Produkten oder Prozessen
- Darstellungsregeln für Produkte oder Prozesse
- Darstellungsregeln für Fenstertypen
- Bestimmung der wichtigsten Fenster und Navigationspfade zwischen diesen Fenstern.

Schnittstellen-Standards beziehen sich auf:

- Regeln für den Einsatz von Steuerelementen
- Regeln für die Positionierung und das Format von Standard-Elementen der Interfaces (Titelzeile, Statuszeile, Navigationssteuerung etc.)
- Terminologie
- Regeln für den Einsatz von Farben
- Regeln für den Einsatz von Schriften
- Regeln für Maus- und Tastaturbelegung
- Regeln für die Gestaltung der Nachrichtenfenster (inhaltlich und Layout)

Die genannten Standards werden nun bei folgenden Aufgaben eingesetzt:

- Vollständige Beschreibung aller Pfade zwischen Fenstern, Dialogboxen und Nachrichtenfenstern
- Vollständiges Design der Menüzeile und aller Steuerelemente
- Vollständiges Design des Inhalts aller Fenster, Dialogboxen und Nachrichtenfenster
- Vollständiges Design aller Interaktionen mit Maus und Tastatur.

Design

Schritt 1

- Vollständige Bestimmung aller Pfade zwischen Fenstern, Dialogboxen und Nachrichtenfenstern

Schritt 2

- Vollständiges Design der Menüzeile und aller anderen Steuerelemente
- Design der Steuerelemente, die es den Nutzern erlauben, sich entlang der Navigationspfade zu bewegen.

Schritt 3

- Vollständiges Design des Inhalts aller Fenster, Dialogboxen und Nachrichtenfenster

Schritt 4

- Vollständiges Design aller Interaktionen mit Maus und Tastatur

- Ziel ist die **Verfeinerung der konkreten Interfaces**.
- **Evaluierung der gesamten Oberfläche** anhand der Usability-Ziele (inkl. Zeitverhalten).
- Normalerweise sollte die Evaluierung in diesem Design-Stadium **nur kosmetische Probleme** aufdecken.
- Einbeziehung von **Funktionalität**, die **bisher noch nicht getestet** wurde.

Die Evaluierung konkreter Interfaces ist ähnlich den vorangegangenen Evaluierungen.

- Die iterative Evaluierung des Konzeptuellen Modells und Schnittstellen-Standards testet und bewertet allgemeine Screen-Designs und einige wenige detaillierte Screen-Designs.
- Test basiert auf der aktuell entwickelten Anwendung
- Testaufgaben für die iterative Evaluierung des Konzeptuellen Modells waren sehr allgemein gefasst und unstrukturiert. Testaufgaben hier sind detaillierter, spezifischer und strukturierter, da eine Vielzahl von Design-Details getestet werden sollen.
- Tests konkreter Interfaces sind eher formaler Natur.
- Evaluierung ist hier stärker auf quantitative Ziele gerichtet sowie auf das Zeitverhalten („Lautes Denken“ kommt hier nicht zur Anwendung).

Die iterative Evaluierung durchläuft folgende Schritte:

- Testplanung und Entwicklung von Begleitmaterial
- Testdurchführung und Datenerfassung
- Analyse und Interpretation der Daten und Formulierung von Empfehlungen für das Redesign.

Am Ende jedes Tests werden modifiziert:

- Interfaces,
- Anwendung und
- Testplan und Begleitmaterial

Testvorbereitung

Schritt 1

- Entscheidung für den Testfokus „ease-of-learning“ oder „ease-of-use“.

Schritt 2

- Aufgabenfokus, Nutzergruppen.
- Grundsätzlich sollte vermieden werden, auf Testnutzer zurückzugreifen, die bereits in vorangegangenen Tests dabei waren: Vorhandene Grundkenntnisse des Designs verfälschen das Ergebnis; Breite Streuung der Testnutzer ist von Vorteil.

Schritt 3

- Entwurf der (realistischen!) Testaufgaben, abgeleitet aus Aufgabenszenarien, evtl. mit Hilfe eines erfahrenen Endnutzers.
- Fokus dieser Tests ist eher auf das Verständnis der Nutzer bezogen auf Schnittstellen-Details und Interaktionen.

Schritt 4

- Erstellung von Begleitmaterial (Briefing, Fragebögen, Training, Aufgabenbeschreibung, Erlaubnis der Videoaufzeichnung)

Schritt 5

- Testumgebung: möglichst realitätsnahe Umgebung, am besten Originalumgebung

Schritt 6

- Bestimmung repräsentativer Pilot-Testnutzer (möglichst unterschiedlich zu vorangegangenen Testnutzern).

Schritt 7

- Pilottest: „Debuggen“ von Test und Begleitmaterial

Schritt 8

- Revision von Testplan und Begleitmaterial

Schritt 9

- Terminierung der Tests und Testnutzer (entsprechend Schritt 2).

Testdurchführung

Schritt 1

- Durchführung der Tests und Datenerfassung.
- Begrüßung, Pre-Test-Fragebogen, Vorstellung des Tests, Training und Präsentation der Testaufgaben.
- Keine Führung der Nutzer durch die Testaufgaben!
- Kein lautes Denken. Videoaufzeichnung von Vorteil, Erfassen von Daten über das Zeitverhalten.
- Festhalten der Daten in den Datenerfassungsbögen.
- Post-Test-Fragebogen.
- Bei vergleichenden Tests: gleiche Aufgaben!

Schritt 2

- Zusammenfassung der Daten (Anzahl der Fehler pro Aufgabe, Art der Fehler, Nutzerkommentare etc.)

Schritt 3

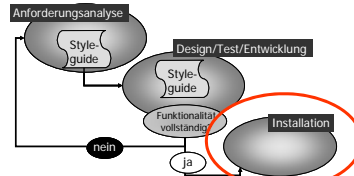
- Analyse und Interpretation der Daten. Fokus auf Datenlage, die bestimmte Probleme anzeigen, die das Erreichen der Mindest-Akzeptanzkriterien verhindern.
- Probleme können sich auf die Usability spezifischer Designs beziehen oder aber auch auf Schnittstellen-Standards oder das Konzeptuelle Modell.

Schritt 4

- Schlussfolgerungen und Redesign-Empfehlungen

Schritt 5

- Dokumentation der Testergebnisse (Executive summary, Darstellung der Probleme und Änderungsempfehlungen).

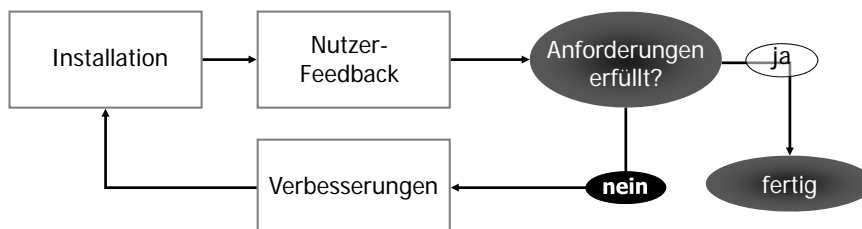


Phase 3: Installation

Aufgabe: Nutzer-Feedback (user feedback)

- Nach der Installation des Produktes werden Meinungen und Reaktionen der Nutzer gesammelt.
- Sie dienen der Verbesserung des Designs, des Designs neuer Releases und dem Design neuer, aber verwandter Produkte.

Usability Engineering - Installation



Nutzer-Feedback

Zweck des Nutzer-Feedbacks nach dem Release der Software, der Installation der Software:

- Daten für die Wartung und Verbesserung der Software
- Daten für zukünftige Releases des Produkts
- Daten für Design und Entwicklung ähnlicher Produkte, die von den gleichen oder ähnlichen Nutzern benutzt werden
- Allgemeine Usability-Erkenntnisse, die wertvoll sind für zukünftige Entwicklungen.

Nutzer-Feedback

- **Möglichkeit der Erweiterung der Evaluierung** auf neue Nutzergruppen oder auf eine Untermenge von Funktionen oder Features, die bisher noch nicht getestet wurden.
- Zunächst wird wieder entschieden, ob sich das Interesse auf ease-of-use oder ease-of-learning bezieht. Diese Entscheidung hat Einfluss auf das **Timing der Evaluation**:
 - Ease-of-learning: Zeit unmittelbar nach Einführung der Software.
 - Ease-of-use: drei bis vier Monate nach Einführung, bis die Nutzer einen gewissen Kenntnisstand erreicht haben.

Nutzer-Feedback

Mögliche Techniken:

- Usability Testing
- Interviews
- Fokus-Gruppen
- Fragebögen
- Nutzungsstudien

Nutzer-Feedback

Welche Techniken eingesetzt werden, hängt von folgenden Parametern ab:

- Verfügbare Ressourcen
- Messung der Performanz oder der Zufriedenheit
- Ease-of-use oder ease-of-learning
- Spezielle Usability-Aspekte oder generell Suche nach Problemen
- Komplexität des Produkts
- Vorgegangene Evaluierungen

Die Techniken können kombiniert werden.

Nutzer-Feedback

Schritt 1: Fragebogen-Entwurf

Schritt 2: Verbesserung des Fragebogenentwurfs

Schritt 3: Verteilung der Fragebögen

Schritt 4: Datenanalyse

Schritt 5: Schlussfolgerungen