



Usability von Open Source Software

Annette Kaudel

Institut für Informatik

FU Berlin

23.07.2006

- **Begriffsdefinition Usability**

- Was macht gute Usability aus?

- **Ursachen**

- Welche Ursachen führen dazu, dass OSS mehr Probleme mit Usability hat als proprietäre Software?
- Inwiefern hängt dies mit dem speziellen Charakter und Entwicklungsprozess von OSS zusammen?

- **Lösungsansätze**

- Welche Lösungsansätze existieren um diese Probleme zu beheben?

- englische Ursprung
- leitet sich von **'user-friendly'** ab
- deutsch: **Benutzerfreundlichkeit**
- Synonyme:
 - Benutzungsfreundlichkeit
 - Gebrauchstauglichkeit
 - Gebrauchsqualität
 - Anwenderfreundlichkeit
 - Ergonomie
 - ...

- Teilaspekt aus dem Wissensgebiet der **Mensch-Computer-Interaktion**
(engl: Human-Computer Interaction, **HCI**)
- **Ziel:**
in den Software-Entwicklungsprozess Methoden zu integrieren,
die ein optimales Arbeiten des späteren Anwenders mit dem
Programm ermöglichen.
- Aus Nutzersicht ist Usability das wichtigste
Bewertungskriterium an eine Software.

- **Anforderungen** (nach Jacob Nielsen)
 - **Erlernbarkeit** (engl.: learnability)
 - gut verständlich
 - => Umgang leicht erlernbar
 - **Einprägsamkeit** (engl.: memorability)
 - erlernter Umgang muss sich gut einprägen
 - **Effizienz** (engl.: efficiency)
 - möglichst geringer Aufwand zum bewältigen einer Aufgabe
 - => produktives Arbeiten
 - **Fehlerrate** (engl.: error rate)
 - auch bei Fehlern stabil bleiben
 - gegenüber fehlerhafter Eingaben tolerant arbeiten
 - **Zufriedenstellung** (engl.: satisfaction)
 - Arbeit mit dem Programm darf keine Belastung sein

- **Programmierer sind selber die Nutzer**

- von Programmierern in erster Linie für sich selbst geschrieben („**Open Source developer-user**“)
- Hauptziel ist nicht möglichst großen Nutzerkreis anzusprechen => Software muss nicht an deren Bedürfnisse angepasst werden
 - „The relation to usability is that this implies that OSS is in certain ironic ways more egotistical than closed source software.“ (Nichols et al. 2003)
 - => breites Angebot an Entwicklersoftware, da für die gleiche Zielgruppe

- Programmierer haben oft Schwierigkeiten sich in die Bedürfnisse für 'Otto-Normal Nutzer' hineinzudenken
 - „That is, while hackers can be very good at designing interfaces for other hackers, they tend to be poor at modeling the thought processes of the other 95% of the population well enough to write interfaces that J. Random End-User and his Aunt Tillie will pay to buy.“ (Raymond 1999)
 - ebenfalls Problem bei proprietärer Software, aber mit Hilfe von z.B.: Usability Experten angegangen
- OSS Grundgedanke der Fehlerentdeckung durch die Community schlägt bei Usability fehl
 - **'wrong kind of eyeballs'**

- **Usability ist kein technisch anspruchsvolles Ziel**

- Motivationsgründe meist Lerneffekt oder Anerkennung innerhalb der Community
 - dies gilt mehr für den technischen Aspekt
- Lerneffekt = Erfahrung im Programmieren zu sammeln
- Ansehen in der Community = technische Leistungen als Programmierer

=> Interesse schwindet Zeit in Usability zu investieren

- **OSS Entwicklung noch relativ neu**

- Im Verhältnis zur Historie des gesamten Software Marktes ist OSS noch relativ jung.
- Usability auch dort zu Anfang kein großes Thema
- erst durch Erreichen der breiteren Masse als Computer-Nutzer Bedeutung gewonnen

=> OSS muss diesen Entwicklungsprozess noch nachholen

● **Fehlende Usability Experten**

- Usability Experten können die Probleme der Programmierer, sich in den Anwender hineinzusetzen, ausgleichen.
- Vielen OSS Projekten fehlen die finanziellen Mittel Usability-Experten zu engagieren.

=> Sie müssten sich aus eigenem Antrieb beteiligen.

- Vermutungen woran dies bisher scheitert:
 - Es gibt generell weniger Usability Experten als Programmierer.
 - Sie fühlen sich in OSS Communities nicht willkommen.
 - Sie interessieren sich nicht so stark für OSS.

- **Schwierige Spezifikation**

- Funktionale Anforderungen sind leichter spezifizierbar als nichtfunktionale wie Usability.

=> für Programmierer auch schwerer diese Anforderungen auch im Programmcode umzusetzen

- kein spezielles Problem von OSS, allerdings verstärkt es sich hier durch das hinzukommen der weiteren Ursachen

- **Verteilte Community**

- Beteiligung an OSS aus zahlreichen verschiedenen Ländern und Kulturen

=> positiver Effekt:

- kulturübergreifenden Austausch fördert Kreativität

=> negativer Effekt:

- wirkt Konsistenz entgegen
 - bedarf gut abgesprochener Zusammenarbeit
 - besonders problematisch beim Aufbau von Benutzeroberflächen und Menüstrukturen

- **Features versus Einfachheit**

- typische Entwicklungsprozess von OSS begünstigt Hinzufügen vieler Features
- Dies wirkt der Einfachheit des Programms entgegen.

=> kann negative Folge für die Usability der Software darstellen

- **indirekt Usability betreffende Ursachen**

für das geringe Interesse der Normal-Computer-Nutzer an OSS

- **proprietäre Software hat sich bereits etabliert**

=> Nutzer bevorzugen diese aufgrund des Gewöhnungseffektes

- Umstieg auf OSS würde neues Erlernen benötigen

- **Nutzung vorhandener Daten**

- Daten in programmspezifischen Format gespeichert
- Umstieg auf OSS ist mit erhöhtem Aufwand verbunden.
- Lösungsansatz: Konvertierungs- oder Import-Funktion gängiger Dateiformate
- trotzdem Effizienz-Verlust

- **fehlende Ressourcen**

- finanziellen Mittel meist nicht vorhanden nötige Ressourcen anzuschaffen
 - Usability Experten
 - spezielle Labore zur Durchführung von Usability Tests
- Problem auch von kleineren und finanzschwächeren Herstellern proprietärer Software

- **offener Quelltext kein Anreiz für Nichtprogrammierer**

- nur ein Anreiz für Programmierer, um die Option zu haben, das Programm ihren Bedürfnissen anzupassen
- für Nicht-Programmierer kein Zusatznutzen gegenüber gleichwertiger proprietärer Software
 - Andere Faktoren wie Preis, Qualität und Support sind eher entscheidend.

- **Unterstützung durch Firmen und Universitäten**

- Ressourcen und Erfahrung bereitstellen
- Beispiele: GNOME, Mozilla und OpenOffice

- **Usability Experten einbeziehen**

- Bugzilla u.ä. geeignet für funktionale Probleme
=> eine auf diese Probleme abgestimmte Plattform errichten
Beispiel: www.OpenUsability.org
 - Hier können Entwickler ihr OSS Projekt anmelden und gezielte Anmerkungen von Usability Experten erhalten.

• Vereinfachte Nutzer-Rückmeldungen

- Problem: Rückmeldungen an die Entwickler bei OSS überwiegend von erfahrenen Anwendern
 - benötigen gewissen Aufwand und Know-How
 - Melden eines 'Bugs' Registrierung bei Bugzilla nötig
- => automatisierte Tools
 - z.B. die Fehlerberichterstattung bei Microsoft Windows XP.
 - „expectation agents“
 - werden ausgelöst wenn ein Anwender eine nicht erwartete Anweisung an das Programm durchführt.
 - (Hinweis an Anwender + Rückmeldung an Programmierer)
- für OSS besonders besonders wichtig, da viele Beobachter hier entscheidend sind

● **Automatisierte Usability Tests**

- ersetzt Ressourcen für umfangreiche User-Befragungen und Usability-Tests ...
- zumindest einige Usability Aspekte können automatisiert nachkontrolliert werden
 - z.B. Performance
- hat seine Grenzen, da Usability schwer spezifizierbar

● **Nutzen von Usability bekannt machen**

- Interesse in der OSS Community für Usability wecken
- viele Anwender der 'eigenen' Software wünschen sich auch bei OSS viele Programmierer

- **Nielsen 1993:** "Usability Engineering". Academic Press
- **Nichols et al. 2003:** "The Usability of Open Source Software," First Monday, vol. 8, no. 1, Jan. 2003. http://firstmonday.org/issues/issue8_1/nichols/
- **Raymond 1998:** "The cathedral and the bazaar," First Monday, http://www.firstmonday.org/issues/issue3_3/raymond/
- **Raymond 1999:** "The Revenge of the Hackers," in Open Sources: Voices from the Open Source Revolution, <http://www.oreilly.com/catalog/opensources/book/raymond2.html>
- **Feller et al. 2000:** "A Framework Analysis of the Open Source Software Development Paradigm," in The 21st International Conference in Information Systems (ICIS 2000), <http://www.josephfeller.com/publications/ICIS2000FellerFitzgerald.pdf>

Vielen Dank!