Open Source in der Medizintechnik

Konzepte und Anwendungsmöglichkeiten

Dr. Matthias Hopf
Professor für Angewandte Informatik

Georg-Simon-Ohm Hochschule Nürnberg
Fakultät Elektrotechnik Feinwerktechnik
Informationstechnik

Ohm / EFI

- Georg-Simon-Ohm-Hochschule Nürnberg
  - Fakultät Elektrotechnik Feinwerktechnik Informationstechnik
  - 50 Professoren
  - Ca. 1300 Studierende
  - Studiengänge E-Technik, Mechatronik/Feinwerktechnik, Media Engineering, Medizintechnik

Meine Wenigkeit

- Professor für Angewandte Informatik
  - Programmiersprachen
  - Computergraphik
  - Multimediaapplikationen (Web 2.0, Android, etc.)
  - Datenbanken
  - Digitaltechnik
- Starker Fokus auf Open Source
  - SuSE Linux GmbH 2004-2011
  - Board of Directors, X.org Foundation

Open Source: Was Ist Das?

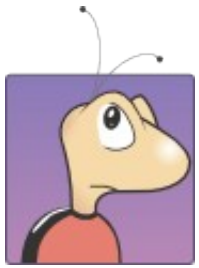
- Erfolgreiches (junges) Entwicklungsmodell
- Bekannteste (extrem erfolgreiche) Beispiele:
  Linux, Android, Firefox, Apache, LibreOffice, gcc
- Ursprünglich nur auf Software bezogen,
  inzwischen auch Medien, Hardware, Geschäftsmodelle,
  etc...



Open Source: Was Ist Das Genau?

- Der *Quelltext* ist frei verfügbar, darf weitergegeben, verändert, und benutzt werden
  - Scheinbar Widerspruch zur Marktwirtschaft
  - Tatsächlich: Erfolgreiche Geschäftsmodelle
- *Free Software*, nicht zu verwechseln mit *Freeware*!
  - Freeware ist kostenlos in der Benutzung
  - Der Quelltext von Freeware ist i.d.R. *nicht* offen
  - Open Source Software *kann* u.U. kostenpflichtig sein
  - Open Source: *„Free as in freedom, not as in beer.“*
- Entscheidend für exakte Eigenschaften: Lizenz

Open Source: Warum Überhaupt



- Sehr starke Entwicklungswerkzeuge
  z.B. GNU Compiler, Bugzilla, svn, git, ...
- Oft guter Support durch Community + Firmen,
  teilw. besser als bei kommerzieller Software
- Langfristig(!) weniger Bugs und höhere Sicherheit
  durch Peer Review der Community
- Massive Kostenersparnisse
  Betriebssystem, Compiler, IDEs, Dokumentverarbeitung,
  Collaborationssoftware, B2B, Shopsysteme, ...
- Richtig genutzt:
  Viel kürzerer Time-To-Market
- **Nicht alles was mit Open Source Software erstellt wird muss auch freigegeben werden!**

Lizenzen

- 4 grundsätzliche Kategorien
  - Keine Codefreigabe (kein Open Source)
    Klassische kommerzielle Softwarelizenz
  - Codefreigabe mit *viralem* Charakter (klassisches Open Source)
    GPL (GNU), AGPL, CC-share-alike
  - Codefreigabe mit *partiell viralem* Charakter
    LGPL, MPL (Mozilla), EPL (Eclipse)
  - Codefreigabe ohne großen Auflagen
    MIT, BSD, X11, Apache, Zlib, Public Domain
- Open Source:
  Viele (>50) nur teilweise kompatible Lizenzen


Viraler Charakter?

- GPL: *wichtigste* Open Source Lizenz
  - Linux, GNU Tools, weite Teile der OS Softwarelandschaft
  - GPL ist *viral*, d.h. Veränderungen *müssen* publiziert werden!
- **Aber**: Nur Veränderungen an der *Software* selber müssen veröffentlicht werden, nicht also:
  - Mit vi/CodeBlocks/OpenOffice geschriebene Programme/Texte
  - Mit gcc übersetzte Programme
  - Mit inkscape gemalte Icons
  - etc...
- **Achtung**: *Linken* mit GPL-Bibliotheken erstellt ein *abgeleitetes Werk* (derivative work) → Veröffentlichung!
  - Bei *LGPL*-Bibliotheken ist diese Einschränkung aufgehoben → Keine Veröffentlichung nötig

Ist Codeveröffentlichung problematisch?

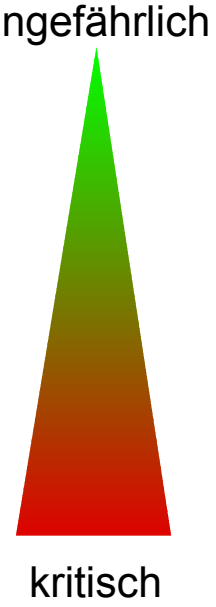
- **Hallo ?!?**

  Da drinnen steckt die gesamte IP der Firma...

- Kommt darauf an...
  - Hardware/Software Kombinationslösungen:
    Verkauft sich über die Hardware
  - Services als Geschäftsmodell
    IP und kommerzieller Erfolg steckt in Web-Services,
    Open Source Clients erzeugen Marketing-Effekt
    Time-To-Market entscheidend
  - In manchen Bereichen: Voraussetzung für Behörden

Anwendungsfälle

- Einfluss der Lizenz:
  abhängig vom Anwendungsfall
  - Betriebssystem auf Entwickler-/Firmenrechner
  - Betriebssystem auf Produkten
  - Entwicklungsumgebungen, Tools, Compiler
  - Server-Infrastruktur
  - Teilkomponenten auf Produkten
  - In der Produktentwicklung benutzte Bibliotheken
  - Codeschnipsel, -beispiele
  - Komplette Produkte



- Je stärker auf Open Source aufgebaut wird desto eher muss der eigene Code veröffentlicht werden

  → Im Zweifel Experten konsultieren

Anwendungsfall: Betriebssystem

- Erstes Szenario:
  OS Betriebssystem auf Firmenrechner,
  OS Betriebssystem auf Entwicklungsrechner

  Harmlos

  - Keinerlei Einfluss der Betriebssystemslizenz
    auf darauf bearbeiteten Daten oder erstellte Programme
  - Guter Support durch Open Source Firmen,
    z.B. SuSE, Canonical, Red Hat, Oracle, ...

- Zweites Szenario:
  OS Betriebssystem auf Produktrechner
  - Keinerlei Einfluss der Betriebssystemslizenz
    auf darauf ausgeführten Programmen
  - Linken gegen Betriebssystem-Bibliotheken in Lizenzen
    explizit erlaubt
  - Sehr vorteilhaft v.a. für Kleincomputer (*Embedded Devices*)

Anwendungsfall: Entwicklungsumgebung

- Szenario:
  Entwicklung mit CodeBlocks + gcc
  - Keine Lizenzgebühren gegenüber
    Microsoft Visual C++ + Microsoft oder Intel Compiler
  - Bekannt *extrem* hohe Codequalität der GNU Compiler Suite
  - Keinerlei Einfluss der Lizenz des Compilers oder der Entwicklungsumgebung auf darauf bearbeitete Daten oder erstellte Programme
  - Compiler in OS Betriebssystemen bereits enthalten, für Windows als MinGW kostenlos und einfach installierbar
  - IDEs: Eclipse, CodeBlocks, Netbeans, kdevelop, qtdeveloper ...
- Auch Tools alleine (z.B. svn, git) können bereits sehr hilfreich sein

Harmlos

Anwendungsfall: Server-Infrastruktur

- Szenario:
  Webservice für Massenmarktprodukte
  - I.d.R. auf Linux + Apache

I.d.R. harmlos

- Eigene Services als externe Programme (cgi):
  - Hier keinerlei Einschränkungen
- Eigene Services gebunden an Server:
  - Server mit nicht viraler Lizenz (z.B. Apache):
    Keinerlei Einschränkungen
  - Server mit viraler Lizenz:
    Keine Einschränkung, solange Server *nicht als Produkt vertrieben wird* – Änderung für den eigenen Betrieb müssen nicht veröffentlicht werden
  - Server mit AGPL Lizenz:
    Vorsicht: *Alle* Änderungen müssen publiziert werden

Anwendungsfall: Teilkomponenten

- Szenario:
  Komplexe Anwendung
  auf eigener Hardware
  - System besteht aus *mehreren* Programmen, manche davon Open Source (z.B. GPL)
- Keine Einschränkungen, solange Programme getrennt übersetzt wurden
  - Kommunikation über Files, Shared Memory, etc. gelten *nicht* als Erstellung eines abgeleiteten Werks
  - Ausnahme wieder: AGPL

Vorsicht

Anwendungsfall: Bibliotheken

- Szenario:
  Produkte mit GUI
  - GUI-Elemente werden über Bibliotheken bereitgestellt (z.B. Qt, GTK)

Vorsicht!!

- Nur Problematisch bei viralen Lizenzen (GPL)
  - Nur partiell virale Lizenzen (LGPL) oder nichtvirale (MIT): Keinerlei Einschränkungen
  - GPL: Eigener Code steht automatisch(!) unter der GPL
  - Durch Gerichte bestätigt
  - GPL 3.0: Keine Einschränkung durch Patente o.ä. erlaubt!

- Vorsicht bei exotischen Bibliotheken
  - GUI-Bibliotheken quasi alle unproblematisch
  - Betriebssystem-Bibliotheken immer unproblematisch

Anwendungsfall: Codeschnipsel

- Szenario:
  Produkt aufbauend auf Codebeispiel
  - Beispiel einer Bibliothek wurde als Basis für die Applikation benutzt

Gefahr!!

- Extrem problematisch solange nicht *explizit* unter nichtviraler Lizenz
  - GPL oder LGPL: Eigener Code steht automatisch(!) unter der gleichen Lizenz
  - Keine Lizenz angegeben:
    Keinerlei Erlaubnis, Code zu verwenden!

- Vorsicht bei Codebeispielen aus Foren
  - Unterhalb einer gewissen Schaffenshöhe unproblematisch
  - Nur: Wie hoch genau ist diese Schaffenshöhe?

Anwendungsfall: Eigene Produkte

- Szenario:
  Android-Client
  für eigenen Webserivce

  Wissentliche Entscheidung

  - Benötigte Funktionalität zu großen Teilen bereits in GPL-Software gefunden
  - Time-To-Market entscheidend

  → Entscheidung: Client als Open Source
- Achtung: Marketing-Effekt nur wenn Reaktionen der Firma positiv eingeschätzt werden
  - Reaktion auf Fehler-Reports (Bugzilla o.ä.)
  - Dokumentation
  - Codequalität
  - Sicherheitslücken auf der Serverseite schneller sichtbar

Fazit

- Open Source Software kann gewinnbringend in Unternehmen eingebracht werden
- Auch wenn eigene IP geschützt werden muss!
- Open Source Betriebssysteme: Unkritisch
- Open Source Entwicklungswerkzeuge: Unkritisch
- Open Source Serversoftware: I.d.R. Unkritisch
- Einsatz von OS Bibliotheken: Lizenz entscheidend
  (Kurzform: Alles außer GPL, AGPL, CC)
- Eigene Projekte als Open Source:
  Manchmal interessant (Marketing, Fehlerbereinigung)
  Wahl der eigenen Lizenz entscheidend
- Standard-Disclaimer: IANAL („*I am not a lawyer*“)
- Folien auf www.hopf.in