

Proprietary vs. Open Source Software Markets in IT: Apple, Microsoft, Google

von Gregor Koppensteiner

Agenda

- Einleitung / Vorwort
- Proprietäre Software – Definition
 - Historischer Hintergrund
 - Philosophie und Abgrenzung
- Open-Source Software – Definition
 - Historischer Hintergrund
 - Philosophie und Abgrenzung
- Vergleich der beiden Innovationsmodelle
 - Aus produktbezogenen Perspektive
 - Aus betriebswirtschaftlichen
- Verwendung in führenden IT-Unternehmen
- Verbreitung von proprietärer und Open-Source Software
- Fazit und Diskussion

„Was verbinden Sie mit dem Begriff
proprietäre Software / Open Source?“



Unterteilung der Seminararbeit in 2 Ansätze:

- aus betriebswirtschaftlichen Betrachtungsweise
- aus einer softwaretechnischen/ produktbezogenen Perspektive

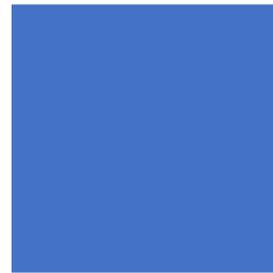
Einleitung

- Interessantes breites Thema, dass aus mehreren Ansichten betrachtet werden kann
- Theoretische und praktische Perspektive mit zahllosen Anwendungsbeispielen
- Zeitliche Relevanz
- Forschungsfrage der Seminararbeit:
 - Was ist die spezielle Natur von proprietärer und Open Source Software und wie koexistieren beide Innovationsmodelle im wirtschaftlichen Kontext?

Quellen



Vorhandene Literatur bezieht sich nur auf jeweils einen der zwei Perspektiven



Ziel: beide vereinen und Basis für nachfolgende Forschungsfragen liefern



72 Quellen 151 Zitationen

Wichtigsten Quellen



Journalartikel: Adoption of open-source software versus proprietary software: An exploratory study von Swati Dhir (2017)



Journalartikel: Competition Between Open Source and Proprietary Software: Strategies for Survival von Michael Sacks (2015)



Essay: Why Software Should Not Have Owners. von Richard M. Stallman (1994)



Journalartikel: Working for Free? Motivations for Participating in Open-Source Projects von Hars & Ou (2002)

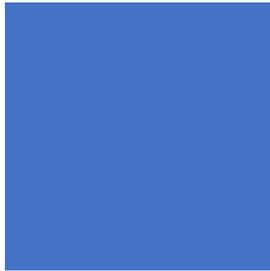
Theoretische Grundlagen

- Letzten 40 Jahre Softwareentwicklung blicken auf immer schnelle Entwicklungen zurück
- Zwei Konzepte etablieren sich

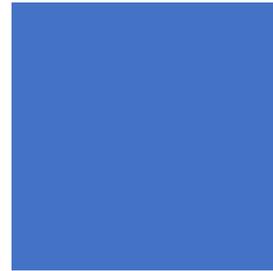
Proprietäre Software

Definition

Proprietäre Software – Definition



Generelles Verständnis eines Produkts: "Herstellungsweise" eines Produktes liegt beim Hersteller - auch bei Software?



„Herstellung“ in diesem Kontext: Quellcode, Zugänglichkeit und Modifizierung



Klares Geschäftsmodell mit eindeutigem Produkt

Proprietäre Software - Definition

Aus Sicht des Entwicklers:

- Angebotene Software unterliegt Urheberrecht, und nur der Autor oder Eigentümer hat die Kontrolle über die (Weiter-) Entwicklung
- Gängige Praxis: Entwickler treten Rechte an Softwareunternehmen ab

Aus Sicht des Konsumenten:

- Quellcode gilt als Wettbewerbsvorteil
 - Schützenswertes Gut aus Sicht des Entwicklers
- Geschützt durch EULA, End User License Agreement

Proprietäre Software - Definition

Welche Freiheiten werden dem User entzogen:

Weitergabe, Einsicht und Modifikation

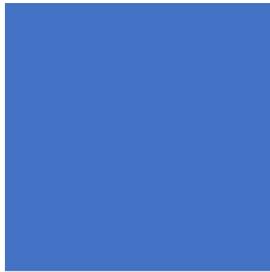
Proprietäre Software

Historischer Hintergrund, Philosophie und Abgrenzung

Proprietäre Software – Historischer Hintergrund



Proprietäre Software – Philosophie



Verknüpfung von Einnahmen
mit Fortschritt



Innovation wird an
Wirtschaftlichkeit gebunden



Unternehmen etablieren
proprietäres Geschäftsmodell

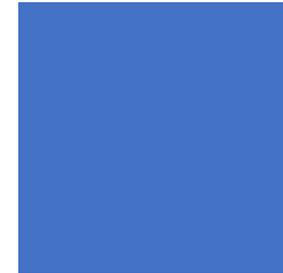
Proprietäre Software – Abgrenzung



Proprietär \neq kommerziell



Freeware kann auch proprietär
sein



Unterscheidung nach
proprietär (Closed-Source)
oder Open-Source

Open-Source Software

Definition

Open-Source Software - Definition

führte zu einem
Überdenken der
etablierten Software-
Strategie

Gegenstück zum
proprietären
Softwareentwicklungs-
Ansatz

in 2000er Jahren als
die „Zukunft der
Softwareentwicklung“
vorhergesagt

Open- Source Software - Definition

Der User kann den Quellcode kann ohne
zusätzliche Kosten:



verbreiten,



einsehen und



beliebig
modifizieren

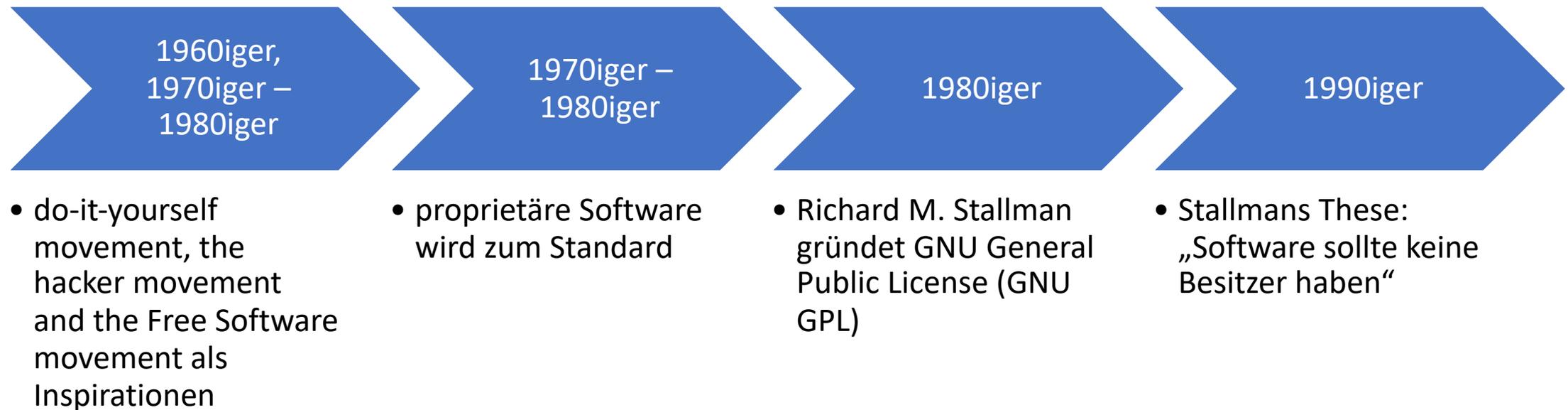
Open-Source Software - Definition

- Populärsten Vertreter von Open-Source Innovation:
 - Linux (Computerbetriebssystem mit ~ 4% Marktanteil)
 - Mozilla Firefox (Web-Browser mit ~ 850 Mio. Downloads)
 - Android (von Open Handset Alliance und Marktführer im Bereich mobileOS)
- im „Mainstream“ angekommen

Open-Source Software

Historischer Hintergrund, Philosophie und Abgrenzung

Open-Source Software – Historischer Hintergrund



Open-Source Software – Philosophie

- Warum sollte jemand seine Arbeit als Open-Source-Projekt zur Verfügung stellen?
- Zwei Arten an Motiven
 - intrinsische Motivation:
 - (1) Bedürfnis nach spezieller Software
 - (2) „fun of the play“ und Lernmethode
 - (3) „gifting“: vom User zum Entwickler werden
 - extrinsische Motivation
 - (1) Anerkennung innerhalb der Gruppe
 - (2) Karrieresprungbrett

Open-Source Software – Abgrenzung

Open-Source ≠ Freeware

Open-Source ≠ Shareware

Open-Source ≠ Trail Software

Bereitstellung des Quellcodes ausschlaggebend:

- Lizenzrechte
- Weiterverbreitung
- Zugänglichkeit und
- individuelle Modifikationen

„Verwenden Sie eher Open Source oder
Proprietäre Software?“

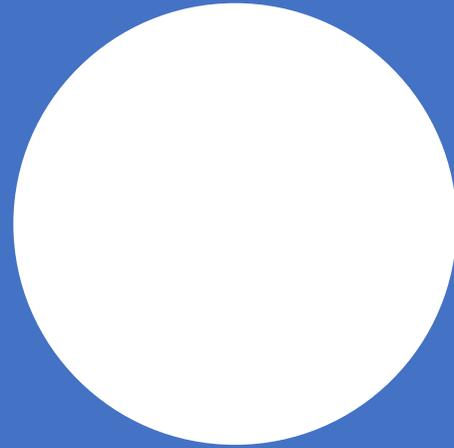


Vergleich der beiden Innovationsmodelle



Vergleich der beiden Innovationsmodelle

- Spaltung unter Softwareentwicklern und Betriebswirten
- philosophische Debatte
- beide Seiten vertreten nachvollziehbare Ansichten



Vergleich der beiden Innovationsmodelle

Aus produktbezogenen Perspektive

Produktbezogener Vergleich: Usability

Proprietäre Software

- harter Wettbewerb führt zu höherem Standard an Benutzerfreundlichkeit
- Usability und Funktionen gelten als Wettbewerbsvorteil und unterliegen stetiger Verbesserung

Open-Source Software

- schlechte Reputation
- Fokus mehr auf Server und Backend-Seite
- lang- bis mittelfristig kein Marktvorteil

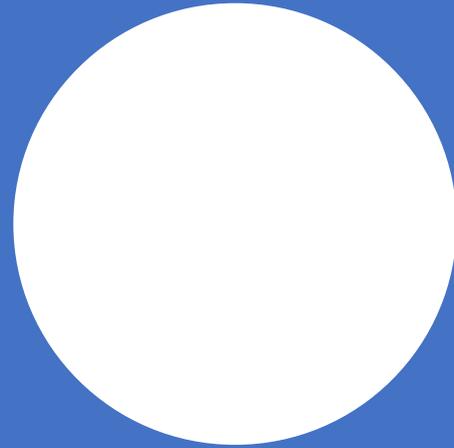
Produktbezogener Vergleich: Benutzerabhängigkeit

Proprietäre Software

- starke Abhängigkeit der User gegenüber der Hersteller
- „vendor lock-in“-Strategy
- erhebliche Wechselkosten

Open-Source Software

- wird durch langlebigen Support und Flexibilität bei der Modifizierung verhindert
- einer der Hauptgründe bei Auswahl von Software in Unternehmen



Vergleich der beiden Innovationsmodelle

Aus wirtschaftlicher Perspektive

Vergleich der beiden Innovationsmodelle



Ökonomen sehen proprietäre Software und Open-Source Software nicht auf den gleichen Märkten konkurrieren



Proprietäre Software-Anbieter stehen in stärkerer Konkurrenz zueinander als Open-Source-Anbieter



Langfristig: Koexistenz, solange die jeweiligen Märkte weiterbestehen



Software kein „Produkt“ im herkömmlichen Sinne

Betriebswirtschaftlicher Vergleich: Total Cost of Production

Proprietäre Software

- Investments in
 - R&D
 - Akquisitionen
- Ziel: externes Know-How wird erkaufte, „Entwicklungs-Outsourcing“
- Gegenfinanzierung hauptsächlich aus Lizenzierungserträgen

Open-Source Software

- ähnliche Investments wie in PS-Unternehmen
- mehr Fokus auf Integration von kleineren Unternehmen ins in-house Team
- Gegenfinanzierung hauptsächlich aus Training, kommerzieller Support und kundenspezifischer Beratung

Betriebswirtschaftlicher Vergleich: Total Cost of Ownership

Proprietäre Software

- hoher Grad an Vertrauen der Kunden
- (Grenz-)Kosten steigen stetig, hauptsächlich wegen erhöhtem Bedarf an Lizenzen

Open-Source Software

- TCO hängt stark von der Qualität und aktiven Bereitschaft der IT-Abteilung ab
- (Grenz-)Kosten abnehmend, aufgrund verwiegender Implementierungskosten

„Welche Aspekte beachten Sie beim Auswahl von Software?“ (100 Punkte zu vergeben)



Verwendung in führenden IT- Unternehmen

Apple, Microsoft und
Google

„Welcher Ansatz wird bei Erstellung Softwareportfolio verfolgt?“



Apple & Open-Source

- Programmiersprache Swift frei zugänglich
- Apple bezeichnete sich 2015 als „das erste große Unternehmen, das die Open-Source-Entwicklung zu einem wichtigen Teil seiner Software-Strategie gemacht hat, diese nutzt und weiterhin erhebliche Mengen an seiner Open-Source-Software veröffentlicht“. → falsche Marketing-Aussage wird revidiert
- Apple erstes Unternehmen, das Profit aus Open-Source schlägt
- langsame Verfügbarkeit von Quellcode
- Ausnutzung monetäre Vorteile

Apple & proprietäre Software

- kein klassisches Softwareunternehmen – Hauptprofit mit Hardware
- kontroverses proprietäres Produktportfolio
 - Diagnose-Tools Software zur Verhinderung von Reparaturen durch Drittanbieter
 - Betriebssystem iOS klassisch für Vendor-Lock-In?

Microsoft & proprietäre Software

- war Jahrzehnte lang der Archetyp für Einsatz proprietärer Software
- hauptsächlich proprietäres Produktportfolio mit kostspieliger Entwicklung
- Einsatz von Vendor-Lock-in-Effekts im großen Stil

Microsoft & Open-Source

- kontroverse Vorgeschichte mit Open-Source-Bewegung
- Turn-around / 180-Grad-Wende nach Einsicht
- inzwischen einer der größten Beitragszahler und Unterstützer von Open-Source Projekten
- schrittweise Annäherung und Anerkennung:
 - Visual Studio (Entwicklungsumgebung)
 - Powershell (plattformübergreifende Eingabefunktion für Betriebssysteme)
 - Teile von Microsoft Edge (Web Browser)
 - Übernahme von GitHub (Code-Repository)

Google & Open-Source

- Google nutzt Innovationsfreude von Open-Source-Entwicklern
- Mitbegründer von Open Handset Alliance
 - Organisation hinter Android
- Android stärkt mit hohem Marktanteil Googles Hauptgeschäft

Google & proprietäre Software

- Chrome Browser: Beispiel für clevere Kombination von OS & PS
 - OS-Elemente dienen als „Unterbau“ (Chromium)
 - PS-Elemente heben sich von der (ebenfalls Chromium-basierten) Konkurrenz ab
- weitere solcher [Projekte](#) in naher Zukunft
- Google schafft es am besten die Vorteile beider Innovationskonzepte zu vereinen

Verbreitung von proprietärer und Open-Source Software

Betrachtung der Märkte für Computer- und
Smartphone-Betriebssysteme, sowie Web Browser

Markt für Computer Operating Systems

- seit jeher von Microsoft Windows dominiert (läuft auf ~ 3/4 aller PCs)
- prominenter OSS-Vertreter: Linux
 - ~ 2% Marktanteil

→ eindeutig proprietärer Gewinner

Markt für Mobile Operating Systems

- seit 2013 dominiert von Android (OSS)
 - erster Verfolger iOS (PS) als Gegenpart
 - Android zeichnet sich durch hohe Modifizierung aus
→ beliebt bei Unternehmen
- eindeutig OS als Gewinner

Markt für Web Browser

- hoher Grad an Wettbewerb (6 Anbieter)
- seit 2012 Google Chrome als Marktführer (setzt große Teile von OSS ein)
- komplette OSS-orientiert: Mozilla Firefox
- Microsoft Edge und Apple Safari nur Nebenakteure

Verbreitung von proprietärer und Open-Source Software



vorgestellte Märkte eher durch hohen Privatkunden-Anteil charakterisiert



hoher Einsatz von OSS in Unternehmen, begründet durch:

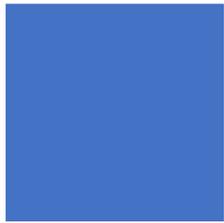
- beliebiger Modifizierung
- Langlebigkeit
- bessere Sicherheit
- teilweise geringere TCO

Fazit und Diskussion

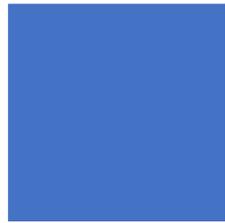
- Anfängliche Trennung der Entwicklungsstile gilt heute schon lange nicht mehr
- Open Source hat sich neben proprietärem Innovationskonzept etabliert
 - Koexistenz in teilweise gleichen Märkten
- Führende IT-Unternehmen implementieren Teile von OSS in bestehende PS-Projekte
 - Kombination betriebswirtschaftlichen mit produktbezogenen Vorteile

Fazit und Diskussion

Für Konsumenten



breitere Auswahl von
verschiedenster Qualität

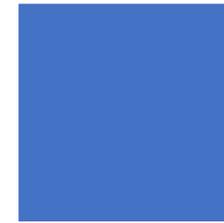


angemessener Freiheitsgrad
zu welchem Preis → trade-off

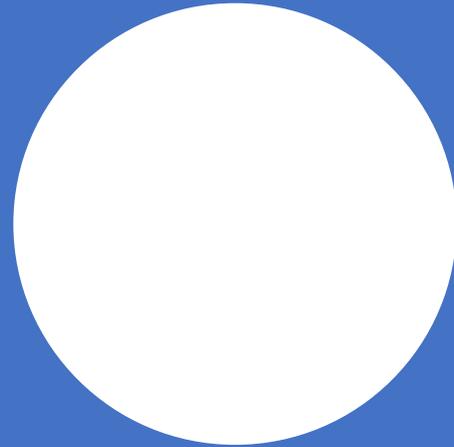
Für Entwickler



Von welcher Philosophie lässt
man sich hauptsächlich
beeinflussen?



Welche Motive sind einem
wichtiger: Das Produkt oder
monetärer Vorteil?



Transparente Quellenangabe

[Link](#) zu offenen Dokument



Diskussion und offene Fragen

Vielen Dank fürs Zuhören