Sieben Anti-Thesen erfolgreicher Open-Source Projekte

Grazer Linuxtage 2008 - Keynote
19. April 2008 9:30, FH Johanneum

Rene Mayrhofer
Universität Wien
Gibraltar / eSYS Informationssysteme GmbH

Was ist Open Source?

Open Source / Free Software bedeutet

- Quelltext ist verfügbar (wenn man die Software hat)
- darf weitergegeben werden
- Änderungen können (in bestimmter Form) weitergegeben werden
- Copyleft: Änderungen davon müssen unter den selben Bedingungen weitergegeben werden

Strenge Definition in Praxis: Debian Free Software Guidelines

Open Source Projekte

- produzieren Open Source ...
- sind aktiv!
  - schon nach Projektbeginn
  - noch vor Projektende
- werden öffentlich verteilt
- werden verwendet

Anti-These 1:
„Ihr habt ein Problem, ich die Lösung“

„Ihr habt ein Problem, ich die Lösung“

⇒ Super!

- „Und noch dazu ist meine Lösung allgemein.“

⇒ Was will man mehr?

- „Sie löst die ganze Problemklasse, ist also auch für die ganz großen Einsatzgebiete gedacht.“

⇒ Geht auch X, Y, Z?

- „Die Architektur kann das alles, und jeder kann einfach die trivialen Implementierungen und Detailprobleme lösen.“

⇒ Wann können wir damit rechnen?

„Ihr habt ein Problem, ich die Lösung“

- Es gibt eine tolle Architekturidee für ein Problem Anderer

⇒ Sie haben ein schönes Dissertationsthema gefunden.

Aber keines für ein neues Open Source Projekt

Eat your own dog food!

- Löse zuerst die eigenen Probleme, dann erst die Anderer!
- Projektleiter und Entwickler müssen selbst auch Benutzer sein (am besten die anspruchsvollsten mit den umfangreichsten Installationen)
- Reality-Check ist wichtig
- Eigenes Leiden an mangelnder Qualität, Dokumentation (was habe ich mir da vor 6 Monaten dabei gedacht?), Installationsroutine, etc. verbessert Projektergebnis ungemein
- Hilft, auf das Wesentliche zu konzentrieren
  ⇒ Funktionsfähigkeit vor Eleganz (aber nicht ohne)
- Wenn die Entwickler es nicht mehr einsetzen, besser die Leitung abgeben

Anti-These 2:
Die Community wird es schon richten

Die Community wird es schon richten

- „Hier ist unser Pre-Alpha Code, er kompiliert, läuft aber nicht wirklich. Testet das mal, sendet detaillierte Fehlerberichte und Patches um das zu beheben. Achja, und hier wäre auch noch die TODO-Liste...“

Die Community wird es schon richten

- Wer ist „die Community“
- Was ist der Anreiz, sich ins Projekt einzubringen?
- Warum die Zeit und Ressourcen in dieses Projekt investieren?

Entwickler müssen die Hauptarbeit machen

- Nur Code der läuft (und installierbar und dokumentiert ist) wird verwendet
- Sorge für minimale Einstiegshürden!
- Mache Mitarbeit leicht
  - Sub-Anti-These 2.1: hier ist unser cooler/s Bug-Tracker / DVCS / Buildsystem mit den meisten Features, darüber läuft Alles
  - Sub-Anti-These 2.2: im Forum hilft sich die Community selbst, ist nicht nötig unsere wertvolle Elite-Zeit damit zu vergeuden
  - Sub-Anti-These 2.3: fragen an die Mailing-Liste die schon im Archiv beantwortet wurden werden ignoriert (oder RTFM/L/F)
  - Oft ist **Dokumentation** wichtiger als neue Features!
- Entwickler / Mitglieder tragen Hauptlast, wahre Selbstläufer sind selten

Anti-These 3:
Die Entwickler haben Recht

Die Entwickler haben Recht

- „Wir wissen es besser“
- „Das ist technisch nicht anders möglich“
- „Das ist unsauber und/oder unelegant“
- „Das passt nicht in das große Design / das elegante Architekturmodell“
- „Nein“

Die Benutzer haben Recht!

- Höre auf die Benutzer!
- Auch wenn sie wenig von den internen Details verstehen, so sind sie doch oft Experten auf ihrem Fachgebiet und wissen manchmal mehr über das Anwendungsszenario als alle versammelten Entwickler
- Aber: vgl. Anti-These 2

Anti-These 4:
„Meine Freizeit geht nur mich etwas an“

„Meine Freizeit geht nur mich etwas an“

- ... solange man alleine wohnt und kein Geld braucht
- „Wenn ich gerade weniger Zeit habe, dann passiert halt einmal weniger.“

⇒ Sollte das Projekt nicht erfolgreich werden?

- Jedes erfolgreiche Projekt braucht ab einer gewissen Phase mehr Zeit als die Lebensumstände an Freizeit erlauben
  - Projekt ändert sich
  - Freizeit ändert sich

Das Projekt als Teil des Berufs

- In der Freizeit kann es starten ⇒ „Garagenprojekte“
- Um erfolgreich zu werden, muss es (Teil des) Beruf(s) werden
  - Langfristige Koordination zwischen Beruf, Privatleben und Open Source Projekt (schwierig)
  - Business Plan (riskant): es ist nicht verboten, Geld zu verdienen – dem Projekt hilft es meistens drastisch
  - Abgeben an Andere, Erreichung des ewigen Status als Projektinitiator und geliebter Emeritus (unwahrscheinlich)

Anti-These 5:
World Domination

World Domination

- Projektziel: 95% XYZ in 5 Jahren
- „Wir erschaffen die beste Lösung."
- „Die ersten Jahre wird es etwas anstrengend am Wochenende, aber dann verdienen wir viel Geld damit."

⇒ Unwahrscheinlich

Open Source heißt Vielfalt und Auswahl

- Viele Möglichkeiten zu haben ist ein Kerngedanke
  - Benutzer wählen die für sie am besten passende Variante
  - Kombination und Integration mehrerer Lösungen
- Konkurrierende Open Source Projekte befruchten sich gegenseitig
  - Forks um Neues zu probieren
  - Andere Architektur, ähnliches Ziel
  - „Das können wir aber besser“
  - vgl. genetische Optimierungsverfahren: Evolution vs. Mutation

Anti-These 6:
Sicherheit bauen wir noch vor 1.0 ein

Zuerst muss es einmal laufen, dann erst...

- „Auf der Uni/FH haben wir das auch so gemacht."

⇒ Wie viel Code aus der Forschung bzw. dem Prototyping wird in der Praxis verwendet?

- „Aber wir wissen ja noch gar nicht was für Bedrohungen auftreten könnten."

⇒ Dann wird es Zeit, **jetzt** darüber nachzudenken!

Sicherheit von Anfang an!

- Internet: kein System ist eine Insel
  - nicht alle Benutzer sind freundlich
  - nicht alle Eingaben sind korrekt
  - erwarte das Unerwartete
- Sicherheit und Schutz der Privatsphäre nachträglich einzubauen funktioniert nicht!
  - **beginnen beim Design** und der Architektur
  - erfordern **Prüfungen auf allen Ebenen** – Syntax, Semantik, Kontext
  - Code setzt dies nur um

Anti-These 7:
Wir machen Software, keine Politik

Ist ja nur Code...

- „Die sollen machen was sie wollen, wir programmieren ja nur“

⇒ Kann man das Projekt dazu verwenden:

  - Daten zu verschlüsseln / signieren?
  - proprietäre Formate / Hardware zu verwenden?
  - unter bestimmten Umständen Angriffe durchzuführen (aktiv oder passiv)?
  - Daten zu verteilen / kopieren?

Nur wie dann?

- „Wie macht man es Allen recht?“
- Gesucht:
  - Hausverstand
  - Interesse für andere Belange – „über den Tellerrand blicken“
  - Verständnis für andere Sichtweisen
  - Zivilcourage

Beispiele

	Debian: *swan	Gibraltar Firewall	Ubuntu	OpenUAT	Gnome vs. KDE	qmail	postfix
Eat your own dog food	~ (o/s)	X	?	~	X	?	?
Hauptarbeit bei den Entwicklern		X	X	z.T.	X	X	**X**
Höre auf die Benutzer	~	X (UW)	**X++**	~	z.T.	~	**X**
Freizeit wird zum Beruf	X	X	**X**	X	z.T.	?	?
Alternativen schätzen	X				**X**	~	X
Sicherheit per Design	X	X	z.T.	X	~	X	**X**
Bezug zur Politik	X	z.T.		X			

Thank you for your attention!

Slides: http://www.mayrhofer.eu.org/presentations
Later questions: rene@mayrhofer.eu.org

OpenPGP key: 0xC3C24BDE
7FE4 0DB5 61EC C645 B2F1 C847 ABB4 8F0D C3C2 4BDE