DFN-Nutzergruppe Hochschulverwaltung
"Auflösung der Verwaltungsgrenzen"

Idee und Organisation von Open Source

9. Mai 2005

Prof. Dr. Nicola Marsden
Studiengang Software Engineering
Hochschule Heilbronn

Idee und Organisation von Open Source

- Open Source Software Development – Ein Wunder?
- …und falls nicht:
  - was ist die ökonomische Logik der Leute, die da mitmachen?
  - was motiviert die Beteiligten?
  - wie wird das ganze koordiniert?
- Open Source Software Development an der Hochschule Heilbronn – Eine Fallstudie

Ein Wunder?

Umfangreiche und komplexe Softwaresysteme können

- erstellt,
- aufrecht erhalten und
- weiterentwickelt

werden in einem nonproprietären Setting, in dem

- viele Entwicklerinnen und Entwickler
- parallel
- in einem relativ unstrukturierten Verfahren

arbeiten.

Wie kann das sein?

- Warum geben Personen freiwillig einen Teil ihrer
  - Zeit
  - Kompetenz
  - kognitiven Ressourcen

  für Open-Source-Projekte?

- Wie funktioniert die Zusammenarbeit dieser einzelnen Personen, so dass ihre individuellen Beiträge ein gemeinsames Produkt ergeben?

Individuelle Perspektive

Soziale/ Organisationsperspektive

Mythos Nr. 1: "Ein gemeinsames Hobby"

- altruistische Motive:
  - das tun, was Freude macht, und es mit anderen teilen um seiner selbst willen
  - nachbarschaftliche Hilfe

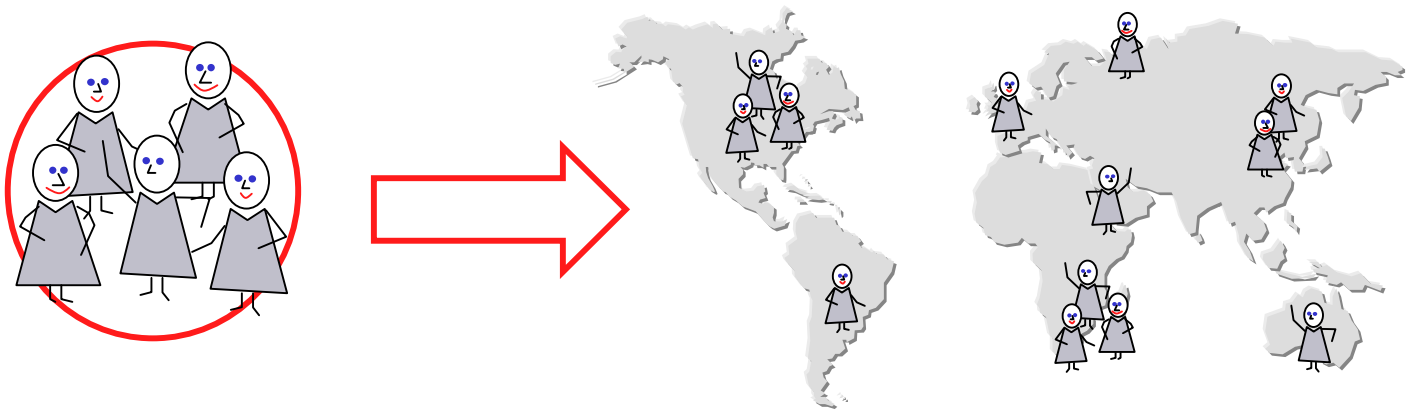
Individuelle Perspektive

- ähnlich denkende und arbeitende Individuen können mit wenig Missverständnissen und Konflikten gut zusammenarbeiten

Soziale/ Organisationsperspektive

Mythos Nr. 2: "Selbstorganisation"

- "Selbstorganisation" ist natürlich nicht falsch, aber um als Begründung einen Wert zu haben, muss das über lokale Gruppenbildung hinaus Besondere an Open Source auch konzeptionell erfasst werden



lokale Gruppenbildung

globale Communities

Erklärungsansätze jenseits der Mythen

- Ökonomische Logik:
  Angebot und Nachfrage? Konkurrenz? Öffentliches Gut?
- Individuelle Motivation:
  Warum stellen Personen freiwillig ihre Kompetenzen etc. zur Verfügung?
- Koordination:
  Individuelle Beträge reichen nicht – wie ergibt sich daraus ein gemeinsames Produkt?

Individuelle Perspektive

Soziale/ Organisationsperspektive

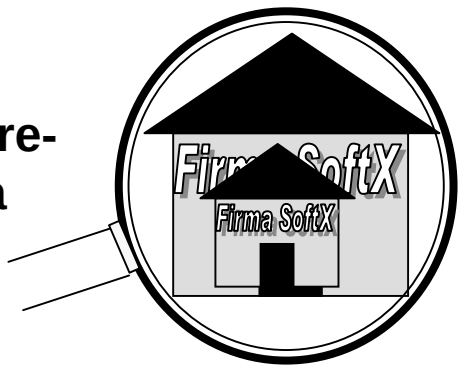
vgl. Weber, Steven (2004). The Success of Open Source. Cambridge: Harvard University Press.

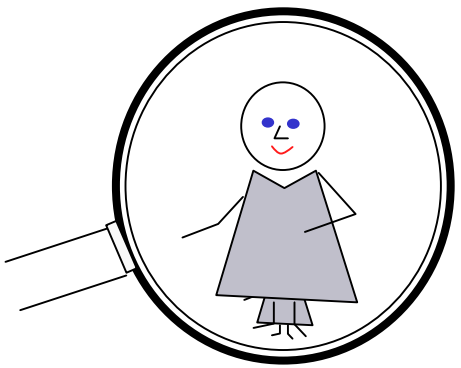
Ökonomische Logik

Warum ein wertvolles Gut verschenken?

- Auflösung des Paradoxes durch Perspektivenwechsel

Software-firma





Prosumer = Benutzer-Programmierer

- Rivalität hoch
- "high cost"-Situation

- Rivalität gering
- "low cost"-Situation

Vorteile, von denen nur Prosumer, nicht aber Trittbrettfahrende profitieren können, brauchen nur gering zu sein

vgl. Luthinger, Benno (2004): Alles aus Spaß? Zur Motivation von Open-Source-Entwicklern. In: Robert A. Gehring und Bernd Lutterbeck, Open-Source-Jahrbuch 2004. Berline: Lehmanns Media. www.thinkahead.org [2005-05-05]

Individuelle Motivation

Ergebnisse empirischer Studien

- **Gebrauch**
  Prosumer: Anpassen/Erstellen von Software für eigenes Problem.
- **Reputation**
  Klare Credits: Signalproduktion
- **Identifikation mit der Gruppe**
  Eingebunden in Gruppe und Subgruppe, gemeinsame Ziele
- **Lernen**
  Erfahrungsgewinn durch komplexe Problemlösung
- **Altruismus**
  a) ideologische Haltung b) Reziprozität
- **Spaß**
  Brooks: "Programming is fun because it gratifies creative longings built deep within us…"

Individuelle Motivation

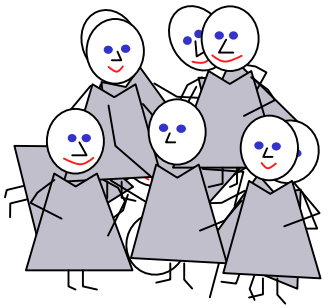
Spaß? Kommerziell vs. Open Source

- **Projektmanagement**
  Die Projektleitung in einem kommerziellen Umfeld ist üblicherweise nicht die Person, welche die Vision über die Applikation entwickelt hat und pflegt.
- **Formale Autorität**
  Die Projekteigentümerin in einem Open-Source-Projekt hat keine formale Autorität.
- **Anreizsystem**
  Programmierende in einem Open-Source-Projekt können üblicherweise nicht über direkte monetäre Anreize zum Engagement bewegt werden.
- **Zeitplan**
  Ein Open-Source-Projekt hat üblicherweise keine Abgabetermine.

Kooperation

Der Basar?

- Wird Open Source Software in einem chaotischen Prozess von Freizeit-Hackern erstellt, die nicht die gleichen Ansichten bezüglich Wartbarkeit haben wie ihre "professionellen" oder akademischen Entsprechungen???



- Praktische Konzept und Techniken, z.B.:
  - Konfigurationsmanagement
  - Refactoring
  - Analyse von Quell-Code
  - Code-Generatoren
  - Separation of Concerns

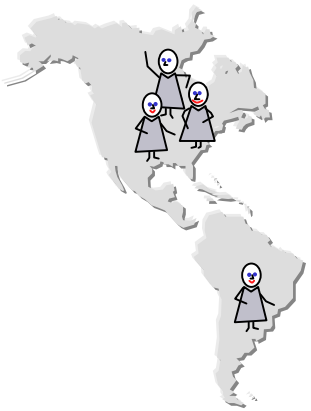
Kooperation

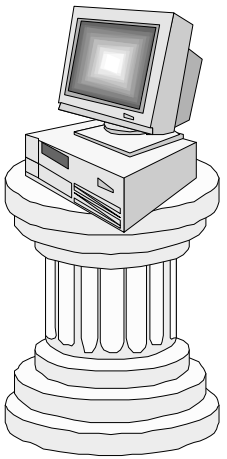
Evolution des Entwicklungsprozesses

- OS-Projekte beginnen meist ohne zusätzlichen Verwaltungsaufwand
- typische frühe Projektphasen zu Strukturierung und Koordination der noch folgenden Phasen spielen oftmals keine Rolle
- aber:
  administrativer Aufwand bleibt nicht konstant

➔ Entwicklungsprozess = dynamisch,
skaliert mit der darunterliegenden Architektur und mit der Anzahl und den Fähigkeiten der Projektbeteiligten

Wie sieht diese Kooperation in der Praxis aus?



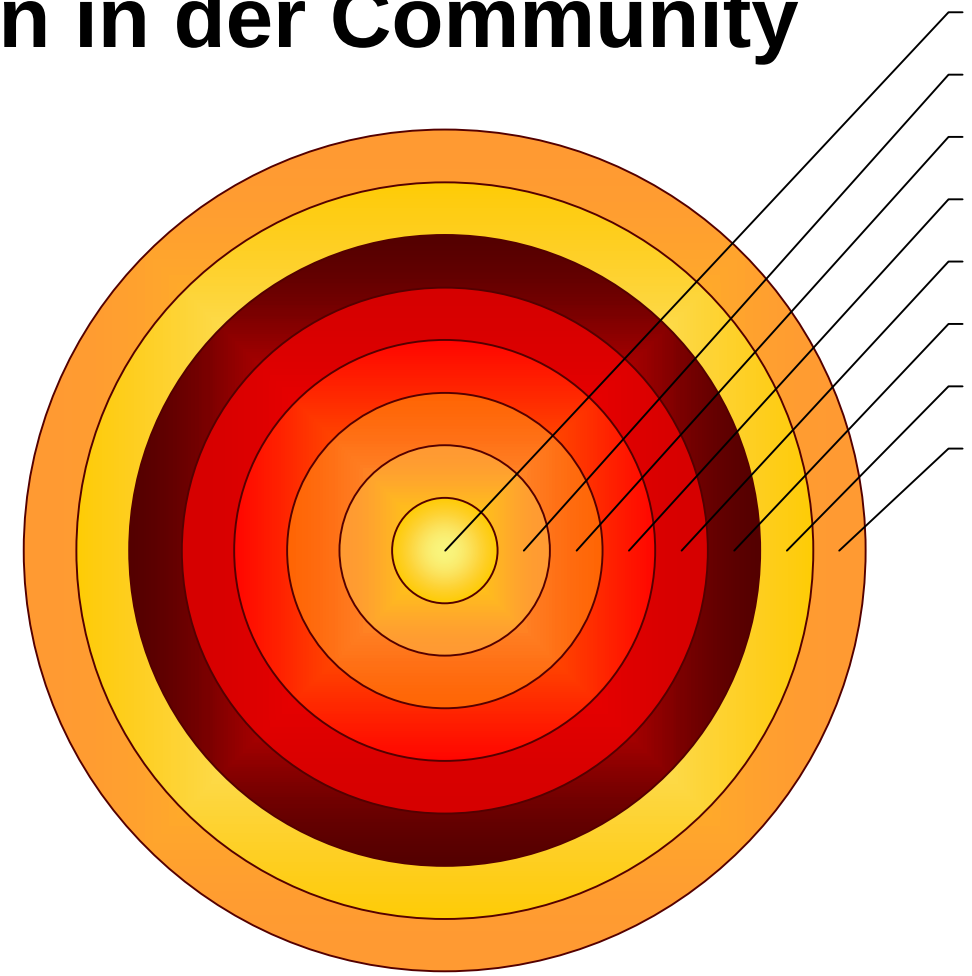


- Entwicklung der Community
- Entwicklung des Systems

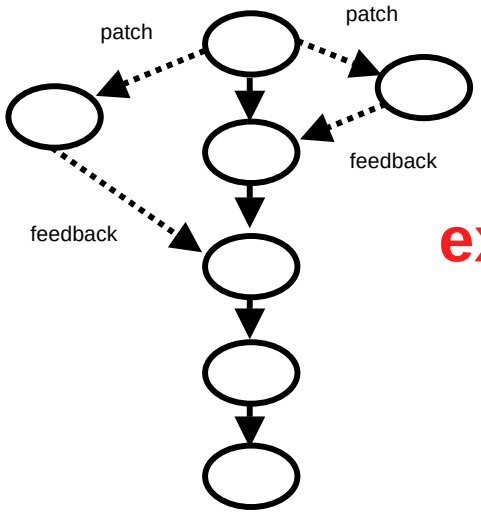
Wie funktioniert die gemeinsame Entwicklung von Kooperation und System?

Welche Evolutionsmuster gibt es?

Kooperation:
Die Rollen in der Community

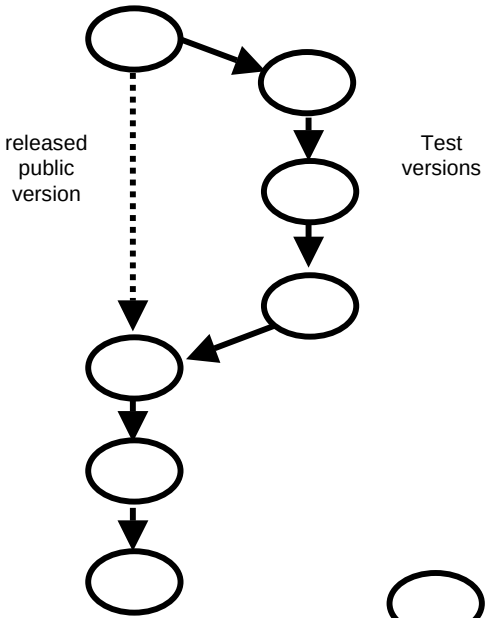


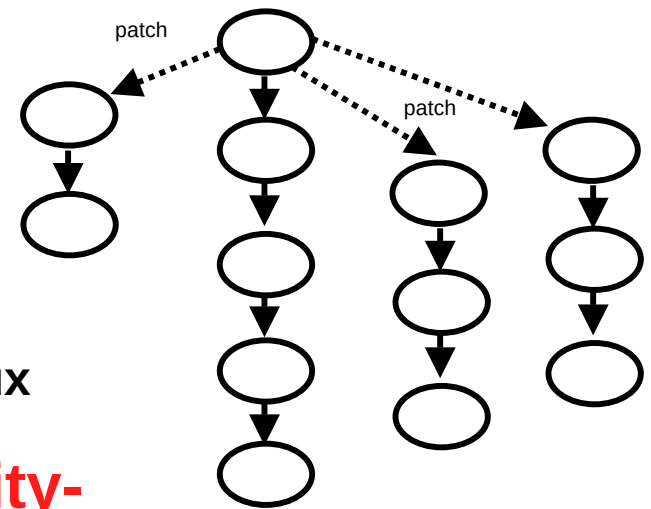
GNU Wingnut



exploration-oriented

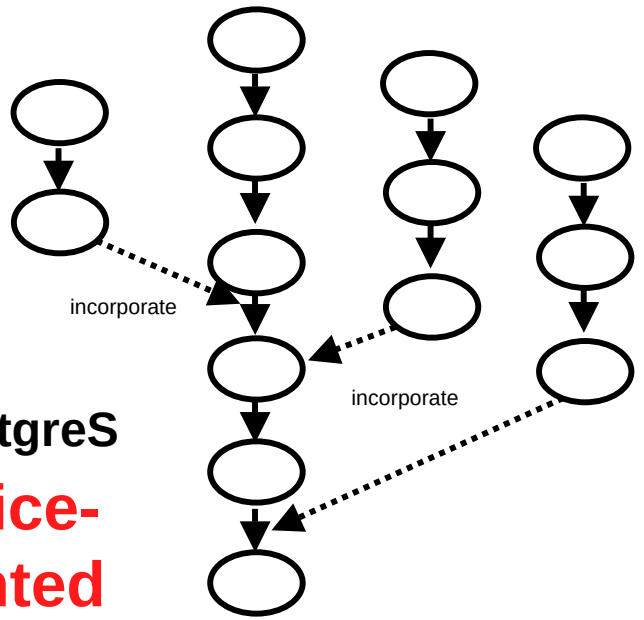
Jun





Linux

utility-oriented



PostgreSQL

service-oriented

Evolutionsmuster von OSS-Projekten

Typ	Ziel	Kontroll-stil	Evolution des Systems	Struktur der Community	Schwächen	Beispiel
Explora-tion-Oriented	Innovation und Wissen teilen	Kathedrale – zentrale Kontrolle	ein Strang, Feedback von Community	Projektleitung, viele LeserInnen	Gefahr der Aufspaltung	GNU-Systeme Jun Perl
Utility-Oriented	Individuelle Bedarfe befriedigen	Basar – dezentra-lisiert	mehrere Versionen parallel Wettbewerb	viele periphere Entwickelnde, Unterstützung für passiv Nutzende	schwierig, das richtige Programm zu wählen	Linux (außer Kernel)
Service-Oriented	Stabilen Service zur Verfügung stellen	zentrale Kontrolle durch Council / Kernteam	ein Strang, Patches werden integriert	Kernteam (core members) statt Projektleitung, viele passiv Nutzende entwickeln für EndnutzerInnen	wenig Innovation	PostgreSQL Apache

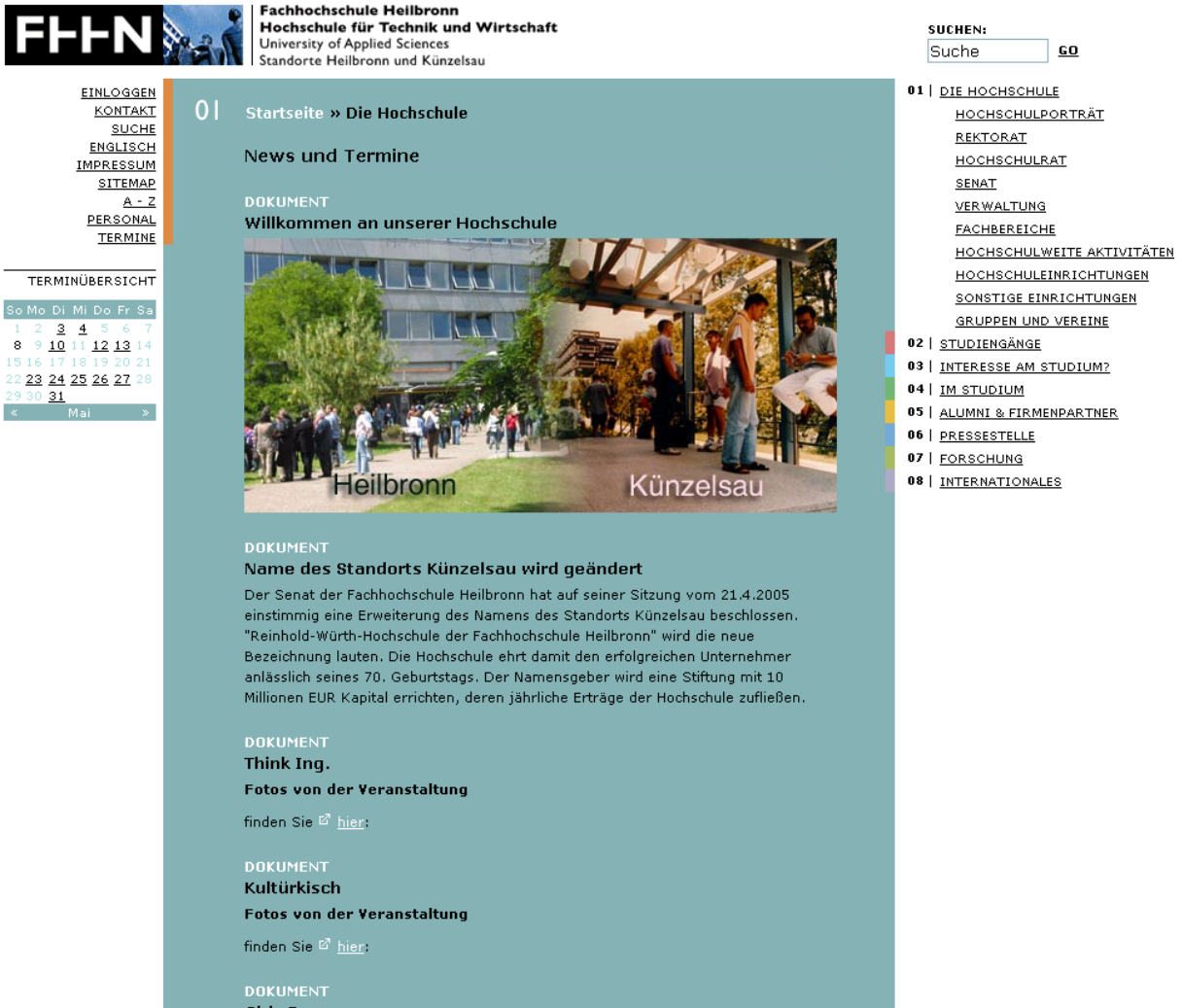
Fallbeispiel Community-Based Open Source Development

Reflexionen zur Lehrveranstaltung "Ausgewählte Kapitel des Software Engineering" (AKSE)

von Prof. Dr. Dominikus Herzberg

Fallbeispiel AKSE

WebCMS der Hochschule Heilbronn



Fallbeispiel AKSE

Rahmenbedingungen

- 35 Studierende der höheren Semester
- 5 Entwicklungsaufgaben zum WebCMS der FH Heilbronn
  - Ziel: eine erfolgreiche Umsetzung aller(!) Aufgaben
  - Zusatz: Dokumentation dazu muss existieren
  - Motivation: produktiver Einsatz im WebCMS der FH
- Infrastruktur
  - Das WebCMS selbst als Kommunikationsplattform
    [Chat, Diskussionsforen, Abstimmungen, Inhalte veröffentlichen etc.]
  - Subversion als Entwicklungs- und Versionierungssystem
  - Ein eigens abgestellter Server als Entwicklungsplattform
- Organisation
  - keine Vorgaben und vollkommen freigestellt
  - Wunsch: Community-Based Open Source Development

Fallbeispiel AKSE

Entwicklungspotential

- Kalkulation
  10 Wochen x 4 SWS x 2 (Heimarbeit) x 0.75 h/Semesterstunde
  = 60 Stunden pro Teilnehmer
  = 2100 Stunden bei 35 Teilnehmern
  ≈ 1½ Personenjahre
- Erwartung
  1½ Personenjahre sind mehr als genug für die Aufgaben
- Dilemma
  Das individuelle Zeitbudget erlaubt nur einen relativ geringen Beitrag beizusteuern
- Herausforderung
  Wie lassen sich die individuellen Beiträge nahezu additiv organisieren?
- Idee
  Kann hier ein Community-Based OSD-Prozess helfen?

Fallbeispiel AKSE

Gegebene Stimulation zu OSD-Prozess: Empfehlungen

- **Spezialisieren Sie sich**
  Suchen Sie sich ein bewältigbares Teilgebiet und werden Sie zum Experten darin
- **Nutzen Sie Stärken und vorhandene Kompetenzen**
  Bieten Sie vorhandenes Können als "Dienstleister" an
- **Suchen Sie sich Ihre "ökologische" Nische**
  Jede/r wird gebraucht; bringen Sie sich ein
- **Seien Sie XZ – extrem zielorientiert**
  Was zählt, ist das Resultat, nicht der Weg dahin. Nutzen Sie Ihre wenige Zeit stets dazu, dem Ziel ein Stückchen näher zu kommen
- **Nutzen Sie alle erdenklichen Informationsquellen**
  Warum Zeit und Energie verschwenden, wenn Hilfe so nahe ist

Fallbeispiel AKSE

Zusammengefasst: Traumhafte Bedingungen

- interessante Aufgabe mit realem Anwendungsbezug
- beachtliches Entwicklungspotential (~1½ Personenjahre)
- ausgezeichnete Infrastruktur
- nahezu ideale Supportbedingungen
- keine organisatorische Vorstrukturierung
  (es wurde lediglich zu Beginn ein Moderator benannt)

➔ Ideale Voraussetzungen für ein OSD-Projekt

Fallbeispiel AKSE

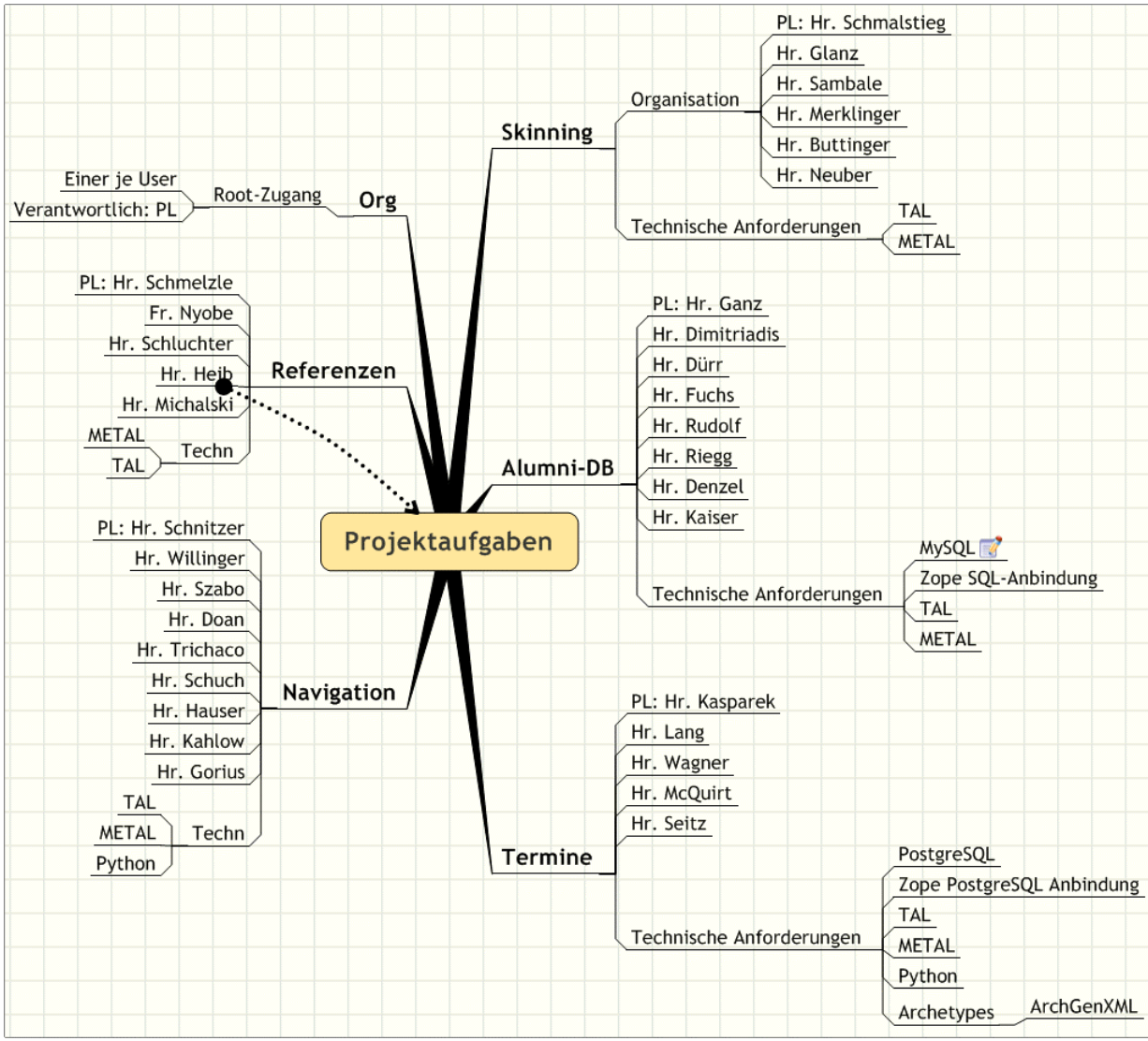
Projektstart: Was zunächst geschah ...

- Aggressive Formierungsphase
  führt zur groben Vorstrukturierung in aufgabenbezogene Gruppen
- Geringe Nutzung der vorhandenen Kommunikationsplattform
- Ausweichen auf selbst eingerichtete Mailing-Listen
  auch dort bleibt der Traffic gering
- Keine Inanspruchnahme des Supports durch Firma
- Fast keine Entwicklungsaktivitäten
- Man spricht von Einarbeitung und liest Dokumentation

Fallbeispiel AKSE

Selbst-organisation: Das Ergebnis

"Wir haben uns dazu entschlossen, in ganz normalen Teams zu arbeiten. Diese Arbeits-weise kennen wir und haben wir schon oft an der FH gemacht."

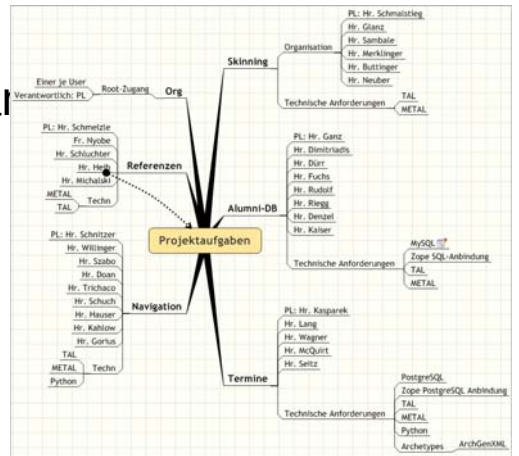


Fallbeispiel AKSE

Überraschung

Community-Based Open Source Development???

- Es gibt eine Projektleitung, die die Brücke zwischen den Teams schlägt
- Jedes Team hat einen Teamleitung
- Jedes Team hat eine Koordinationsstelle zu den anderen Tea
- Jedes Team hat EntwicklerInnen, Dokumentare etc.
- Es wird bislang nur wenig von der Entwicklungs- und Kommunikationsinfrastruktur genutzt



Kritik

- Die Teams sind zu groß ➔ unproduktiver Arbeitseinsatz
- Gruppen lokalisieren Wissen und Können; gruppenweiter Austausch fraglich
- Risikoschub: Problem additiver Einzelbeiträge wird missachtet
- Der Fokus verengt sich auf Teilvorhaben

Fallbeispiel AKSE

Vermutungen

- Community-based OSD benötigt ein Setting, das kaum provoziert werden kann bei
  - aufgenötigter Beteiligung (Pflichtveranstaltung)
  - unfreier Themenwahl (vorgegebene Aufgaben)
  - von außen bewertetem Verhalten (Notengebung)
  - vorgegebenem Zeitrahmen (einsemestrige Veranstaltung)

- Es nützt dann auch nichts
  - ein Projekt als OS-Vorhaben zu deklarieren und den Code freizugeben,
  - den produktiven Einsatz der Software in Aussicht zu stellen,
  - die Vision zum Nutzen der Allgemeinheit anzuführen,
  - eine nahezu ideale Infrastruktur anzubieten

Fallbeispiel AKSE

Warum ist das Ergebnis interessant?

- Kommerzialisierung des Open Source
  Wie kann in einem unternehmensgleichen Setting die Motivationsstruktur und Leistungsfähigkeit eines Open-Source-Projekts hergestellt werden? ("closed open source")

- Forschungsfragen
  Ist die Organisationsstruktur und Vorgehensweise im AKSE-Projekt Symptom der grundsätzlichen Problematik, fremdinitiativ einen OS-Spirit bzw. einen OSD-Prozess zu evozieren? Oder ist es schlicht das Resultat einer schlechten Vorbereitung und mangelhaften Begleitung zum Aufbau eines OSD-Prozesses?

"The best of both worlds" – geht das?

Teil 1

Open Source Software Development		Software Engineering in Organisationen
Konflikte: evolutionärer Ansatz: die bessere Lösung setzt sich durch	⟷	Konflikte: Einigung nötig
Prosumer – ursprüngliche AutorIn des Codes wird Maintainer	⟷	Management und Implementierung getrennt
jede/r entscheidet selbst, woran er oder sie arbeiten will	⟷	klare Rahmenbedingungen und Vorgaben
intrinsische Motivation vorhanden	⟷	Anreizsystem

"The best of both worlds" – geht das?

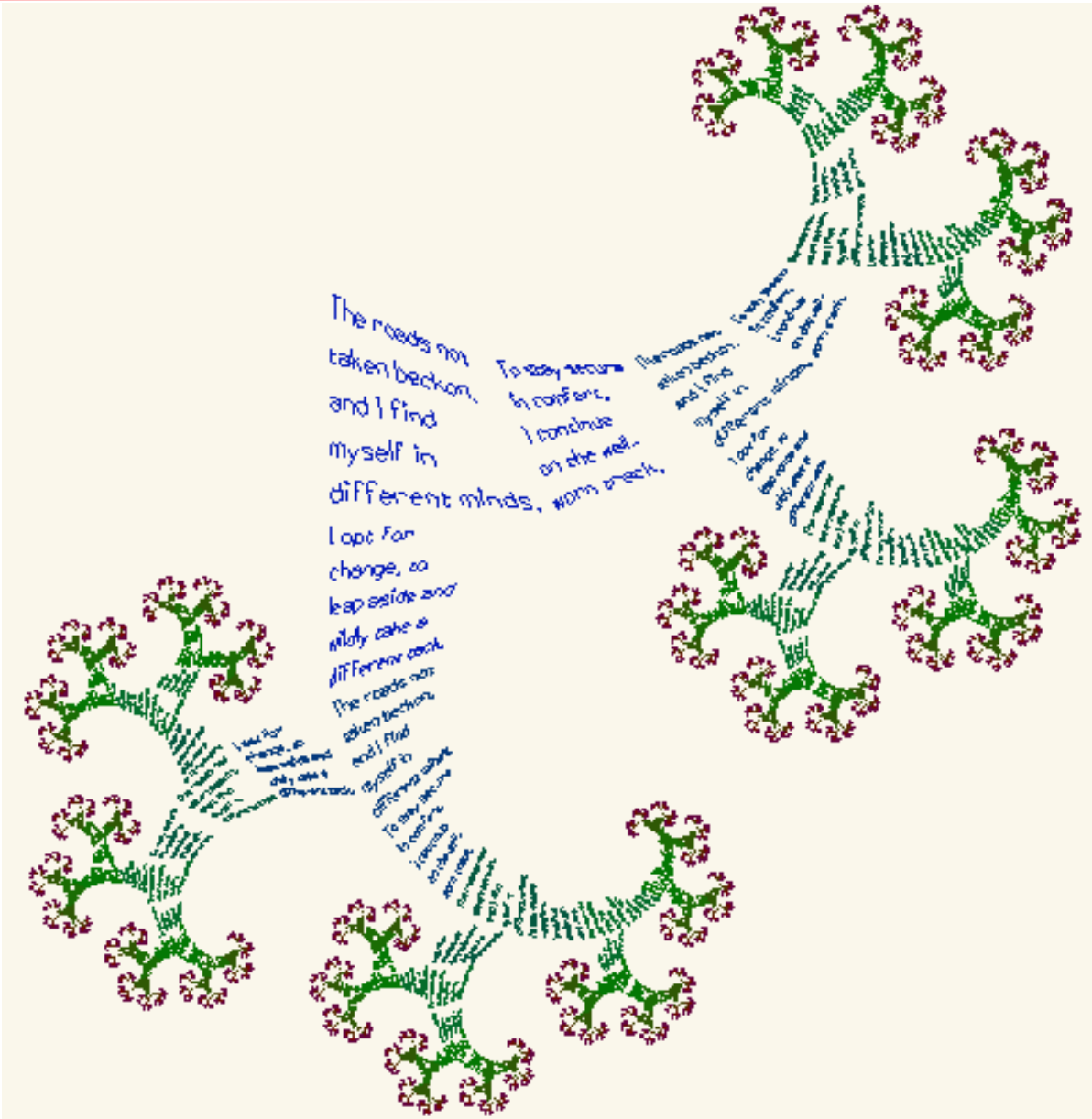
Teil 2

Open Source Software Development		Software Engineering in Organisationen
Koordination durch sofortiges Einspielen der Code-Änderungen (lauffähig!)	⟷	Koordination durch strategische Entwicklungsplanung à la "wer arbeitet wann woran?"
zeitliche Flexibilität	⟷	feste Zeitplanung
Autorität durch fachliche Kompetenz	⟷	Autorität durch Status, Persönlichkeitseigenschaften u.a.
konkrete eigene Problemstellung	⟷	Zielsetzung

Fazit:
Idee und Organisation von Open Source

- Open Source Software Development – Ein Wunder?
- …und falls nicht:
  - was ist die ökonomische Logik der Leute, die da mitmachen?
  - was motiviert die Beteiligten?
  - wie wird das ganze koordiniert?
- Open Source Software Development an der Hochschule Heilbronn – Eine Fallstudie

Danke für Ihre Aufmerksamkeit



http://www.moonstar.com/~nedmay/images/forks1.gif
[2005-01-03]