

Understanding the Requirements for Developing Open Source Software Systems

Christian Spinner

Integrations Engineering HFU-Furtwangen

17. Juni 2009

- 1 Autor
- 2 Paper
 - Thema des Papers
 - Vorgehen des Autors
- 3 Inhalt und Ergebnis des Papers
- 4 Résumé

Walt Scacchi

Work

Senior Research Scientist am Institut für Software Research, Information and Computer Science (University of California, Irvine)



Akademischer Hintergrund

- B.S. Computer Science, California State University Fullerton (1973)
- B.A. Mathematics, California State University Fullerton (1973)
- Ph.D. Information and Computer Science, University of California Irvine (1981)

Walt Scacchi

Interessen

- Open Source Software Development Communities, Processes and Practices
- Software Process (Re)Engineering
- Computer Game Culture and Technology

Bisherige Arbeiten

- beteiligt an ca. über 30 Papers
- Software Engineering und Prozesse, Open Source

“Open source development ist besser, schneller, und billiger”

Understanding the Requirements for Developing Open Source Software Systems

- veröffentlicht Februar 2002
- IEEE - Paper
- Revision 2008
- 26 Seiten

Wie oft wurde dieses Paper zitiert?

- citeseer : 66. zitiert
- davon 33. selbst zitiert

Thema und Ziel

Thema

- Empirische Studie über die Anforderungen und Prozesse für die Entwicklung von Open Source Projekten
- weniger auf der technischen Ebene

Ziel

- Vergleich mit traditionellem Weg der Softwareindustrie
- Modelle und Artefakte der Open Source Welt
- Verstehen

Thema und Ziel

Thema

- Empirische Studie über die Anforderungen und Prozesse für die Entwicklung von Open Source Projekten
- weniger auf der technischen Ebene

Ziel

- Vergleich mit traditionellem Weg der Softwareindustrie
- Modelle und Artefakte der Open Source Welt
- Verstehen

Research-Prinzipien

hermeneutic circle

- Analyse und Interpretation des gesamten Open Source Requirements Prozesses
- Unter Berücksichtigung aller Teile

contextualization

- Identifizieren des Umfeldes
- Soziale und historische Backgrounds

Research-Prinzipien(2)

revealing the interaction of the researcher and subjects/artifacts

- Offenlegung der Interaktionen des Forschers
- Hier: Forscher ist Teilnehmer(Leser)-Zeitraum: 10 Monate
- Hier: Soziale und technische Interaktionen der Open Source Welt

abstraction and generalization

- Vergleich der Methoden und Artefakte über 4 Verschiedene Open Source Community Arten
- Zusammenfassung
- Generierung eines allgemeineren Modells

Research-Prinzipien(2)

revealing the interaction of the researcher and subjects/artifacts

- Offenlegung der Interaktionen des Forschers
- Hier: Forscher ist Teilnehmer(Leser)-Zeitraum: 10 Monate
- Hier: Soziale und technische Interaktionen der Open Source Welt

abstraction and generalization

- Vergleich der Methoden und Artefakte über 4 Verschiedene Open Source Community Arten
- Zusammenfassung
- Generierung eines allgemeineren Modells

Research-Prinzipien(3)

dialogical reasoning

- Vergleich mit vorhandenem Wissen
- traditionellem Weg

multiple interpretations

- Berücksichtigen von mehreren Teilnehmern
- Ansichtsweisen und Erfahrungen

Untersuchte Open Source Community Arten

- Networked Computer Game Worlds
- Internet/ Web Infrastructure
- X-ray astronomy and deep space imaging
- Academic software systems design



Eigenschaften der Communities

- Kein Face to Face
- Vernetztes Arbeiten
- Unterschiedliche Technologien
- Unterschiedliche Community Gedanken und Mitgliederanforderungen
- Technologien: Websites, Foren, Email, Internet Messaging, Bug-Reporting, Dokumentationen für den Einstieg, Repositories, Hyperlinks

Der Klassische Requirements Engineering Prozess

- Interviews und Dokumentationen
- Modellieren oder Spezifizieren
- Analyse
- Validierung
- SRS

Vergleich der beiden Prozesse

In der Open Source Welt:

- Keine Dokumente, die man als Spezifikation bezeichnen kann
- Behauptung der Requirements
- Spezifizierung implizit und verteilt
- Self-managed (Meritokratie)- Virtual-Owners
- Soziale Mechanismen sehr wichtig
- Alles ist global erreichbar

Verstehen der Anforderungen für Open Source Entwicklungen

- Auswahl von Software Informalisms
 - Website Posts
 - Email-List and Discussions
 - Code Dokumentation
 - Howtos, FAQs
- Formlose Beschreibungen
- Features oft schon implementiert und werden als Anforderung behauptet
- Mittelpunkt ist die Website

Verstehen der Anforderungen für Open Source Entwicklungen(2)

- Requirements = Beiprodukt
- Teilnehmer muss verstehen wo diese Requirements stehen
- Entwickler = Endbenutzer
- Community Building
- Komplexes Netzwerk aus sozialtechnischen Prozessen

Résumé

- Repräsentativ (Fast alle Sparten des Open Source Bereichs abgedeckt)
- Methoden der Studien werden erklärt
- Das Vorgehen ist sehr wissenschaftlich, und gut nachvollziehbar
- Der Autor ist bekannt in diesem Bereich
- Keine negativen Zitierungen