



Statische Rekonstruktion von Ökosystemen

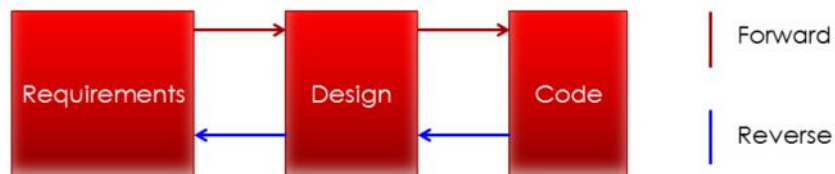
Master Seminar SS 2013
Andreas Knecht, email: 1214065@stud.hs-mannheim.de

Inhalt

- Einleitung
 - Reverse Engineering
 - Bereiche des Reverse Engineerings
 - Reverse Engineering in Ökosystemen
- Small Repository Observatory
 - Hauptansicht
 - Projekte und Entwickler
 - Kontext und Fokus
 - Bewertung

Reverse Engineering

- Unterdisziplin des Software-Engineerings, die sich mit der Rekonstruktion von Software-Systemen beschäftigt
- Identifikation der Systemkomponenten und deren Beziehung[6]
- Ziel: Beschreibung des Systems auf einer höheren Abstraktionsebene



Ursprüngliche Entstehung aus dem Maschinenbau.

Tools: Hexeditoren, Debugger, Disassembler, Decompiler

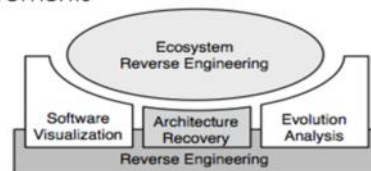
Kaum formale Methoden

Oftmals sind die Wartungsprogrammierer nicht die ursprünglichen und diese müssen das System verstehen

Wartungszeit umfasst 50-75% der Gesamtentwicklungskosten, dabei wird die meiste Zeit für das Verstehen verbraucht.

Bereiche des Reverse-Engineering

- SV: UML Darstellung, aber auch andere Formen, die im Wesentlichen aus (gerichteten) Graphen abgeleitet werden
- AR: Darstellung einiger Architektur Viewpoints, jedoch keine vollständig automatisierte Architekturrekonstruktion möglich
 - Es fehlen Begründungen einzelner Konzepte
 - Es fehlen die Qualitätsziele der Architektur
 - Es fehlen damalige Requirements
- EV: Codemetriken



SV: Reduktion von Komplexität

Ursprüngliche Entstehung aus dem Maschinenbau.

Tools: Hexeditoren, Debugger, Disassembler, Decompiler

Kaum formale Methoden

Reverse Engineering bei Ökosystemen

- "A software ecosystem is a collection of software projects which are developed and evolve together in the same environment." [1, 2, 5]
- Menschen, d.h. Entwickler sind Teil des Ökosystems
- Es gibt Studien die sich nur damit beschäftigen, wie Entwicklungsabläufe im Ökosystem funktionieren [3]
- Abhängig von der jeweiligen Perspektive
- Hauptinformationsquelle: Repository der jeweiligen Organisation, aber auch Bugtracker, Emails und MailingLists

Lungu definiert Ökosysteme auf rein technischer Ebene. Andere Autoren haben eine stärker akzeptierte Definition, die in mehr Papern referenziert wurde [7]

Repository-Informationen: Personen, Datum, Change-Log, Revision, Sourcecode und Files

Mailing-Lists: Thema, Personennamen, Zugehörigkeit zur Organisation (organisationstypische Email)

Emails: Kontakt zwischen Personen, Thema, Keywords im Text

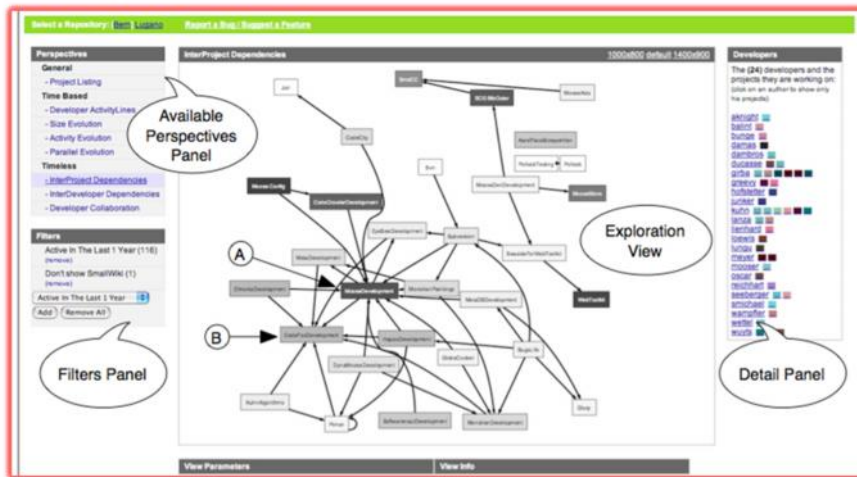
Bugtracker: Issues, wie lange dauerte im Schnitt das Lösen, Wie oft wurden Tickets wiedereröffnet

Small Repository Observatory

- Initiiert von Mircea Lungu et al., einem der führenden Software-Engineer auf dem Gebiet der Ökosystem Rekonstruktion
- Webapplikation, mit der Ökosysteme statisch analysiert werden können
- Nach Ansicht der Autoren haben sich frühere Rekonstruktionen zu sehr auf den Code konzentriert und nur Projekt für Projekt analysiert
- Bietet auch Organisationsspezifische Rekonstruktionsmöglichkeiten
- z.Z nicht öffentlich verwendbar

Lungu hat sogar eine Doktorarbeit darüber geschrieben und wird in nahezu jedem Rekonstruktionspaper referenziert oder erwähnt
Außerdem stammen 2 Systeme von ihm

Hauptansicht



Projects@Focus

- Welche Abhängigkeiten gibt es ? (Project Dependency Map)



Je größer die Anzahl der Klassen des Projektes, desto größer das Rechteck
Je mehr Commits in das Projekte getätigt wurden, desto dunkler die Färbung
Die Kanten sind Abhängigkeiten der Form Methodenaufwurf oder Subclassing

Projects@Context

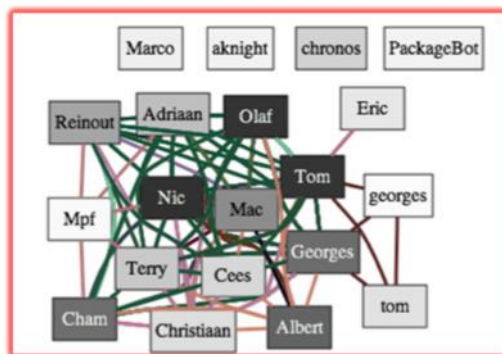
- Warum gibt es Abhängigkeiten?



Rot gefärbt = Verwendung von Methoden einer Klassen
Lila gefärbt = Subclassing

Developer@Focus

- Welche Entwickler gibt es und welche arbeiten vermutlich zusammen ? (Developer Collaboration Map)



Zwei Entwickler arbeiten zusammen, wenn sie im gleichen Projekt in einem gewissen Zeitraum mehr commits getätigt haben als ein gegebener Schwellwert.

Bewertung von statischen Ökosystemrekonstruktionen

- Rein projektspezifische Informationen reichen zur Ökosystemanalyse nicht aus
- Statische Rekonstruktion bietet eine Reihe von Vorteilen:
 - Analyse ohne Programmausführung
 - Keine Konfiguration oder Build nötig
 - Keine Ausführung von gefährlichem Code
- Metriken: Es ist nur das interpretierbar, was anhand der Informationsquellen gemessen werden kann und die Resultate sind oft relativ

Metriken: Das macht die Analyse schwierig und ist auch beschränkt auf die Informationsquellen des Repositories

Vielen Dank für eure
Aufmerksamkeit

- Gibt es Fragen ?

Quellen

- [1] Mircea Lungu, Michele Lanza. The Small Project Observatory - A Tool für Reserver Engineering Software Ecosystems. University of Lugano, Switzerland, ISSN: 0270-5257, ISBN: 978-1-60558-719-6, 2010
- [2] Kartiga Mohanmani, Chamundeswari Arumugam. Software Visualization of Text Context in Ecosystems. India ISSN: 2250-3005
- [3] Audris Mockus, Roy T Fielding, James Herbsleb. Two Case Studies of Open Source Software Development: Apache and Mozilla. Journal, ACM Transactions on Software Engineering and Methodology (TOSEM)
- [4] Mathieu Goeminne, Tom Mens. A Framework for Analysing and Visualising Open Source Software Ecosystems. ISBN: 978-1-4503-0128-2

Quellen II

- [5] Mircea F. Lungu. Reverse Engineering Software Ecosystems. Doctoral Dissertation, 2009
- [6] Reverse Engineering and Design Recovery: A Taxonomy, Elliot J.Chikosfky, James H.Cross II, IEEE Software 1990
- [7] Software Ecosystems - A Systematic Literature Review, Kontantinos Manikas und Klaus Marius Hansen, 02/2012, ISSN: 0107-8283