Empirische Untersuchungen des Side-by-Side Programming

Ulrich Stärk

Institut für Informatik

FU Berlin

10.01.2008

Gliederung

- Side-By-Side Programming
- Ziele der Arbeit
- Durchführung
- Probleme
- Blickwinkel
- Weiteres Vorgehen

Side-by-Side Programming

Freie Universität Berlin

- Eine von Alistair Cockburn in Crystal Clear vorgeschlagene Technik
  - Alistair Cockburn. Crystal Clear. Addison-Wesley, 2004.
  - Beschreibt Sicherheitsmaßnahmen um einen erfolgreichen Ausgang eines Projekts zu unterstützen
    - Im Gegensatz zu detaillierten Methoden z.B. in XP
    - Häufige Releases, laufende Verbesserung, Osmotische Kommunikation, persönliche Sicherheit, Fokus auf das Projekt, Zugang zu erfahrenen Benutzern, Automatische Tests, Konfigurationsmanagement und häufige Integration

Side-by-Side Programming (2)

Freie Universität Berlin

- 2 Programmierer arbeiten unmittelbar nebeneinander
  - Jeder arbeitet an seiner Aufgabe
  - Bildschirm des jeweils anderen leicht einsehbar
  - Zusammenkünfte möglich um Fehler zu suchen, gemeinsam ein Stück Code zu schreiben, etc.

- Weniger invasive Alternative zu Pair Programming
  - „traditionelle“ Arbeitsweise
    - Jeder Entwickler Arbeitet für sich am eigenen Computer
  - Trotzdem „Osmotische Kommunikation“
    - Wissenstransfer durch z.B. zufälliges Mithören oder Mitbekommen von Tätigkeiten des Anderen

Stand der Forschung

- Bis auf ein Paper keinerlei wissenschaftliche Arbeiten darüber
- Jerzy R. Nawrocki, Michal Jasinski, Lukasz Olek, and Barbara Lange. Pair Programming vs. Side-by-Side Programming. Lecture Notes in Computer Science, 3792:28-38, 2005.
- Quantitativer Vergleich von Soloprogrammierung, Pair Programming und Side-By-Side Programming
  - Side-By-Side Programming schneller als Solo- und Pair Programming
    - Gleiche Aufgabe in kürzerer Zeit erledigt (Solo 100%, PP 74%, SBS 61%)
    - Somit nur 22% Overhead beim SBSP hingegen 48% beim PP im Vergleich zu Solo Programmierung
  - Allerdings vermutlich weniger Kenntnis vom gesamten Quellcode als beim Pair Programming
    - Umsetzung eines Change Requests dauerte länger als beim Pair und Solo Programming
    - Schwache Hypothese

Ziele

- Beitrag zum Wissen über Side-by-Side Programming
  - Bisher nur ein Artikel, der das Thema behandelt

- Entwicklung einer Theorie
  - Grounded Theory
    - Völlig unbekanntes Terrain, daher explorative Methode erforderlich
  - Entwicklung eines Kodierschemas
    - Benennungen für beobachtete Phänomene
  - Kodierung von Videodaten
    - Durch Beobachtung von Programmierenden gewonnen
  - Blickwinkel auf die Umstände, die zu Trennung / Zusammenkunft führen

Grounded Theory

Freie Universität Berlin

- Entwicklung von in Daten begründeten Theorien
- Erdacht von Strauss und Glaser Mitte 1960
- Ausprägung nach Strauss/Corbin
  - Kodieren der Daten
    - Vergabe von Codes für die beobachteten Konzepte
    - Dabei ständige Überprüfung mit bereits kodierten Phänomenen
  - Identifizierung von Beziehungen zwischen den Konzepten
  - Formulierung einer Theorie aus den Konzepten und ihren Beziehungen zueinander
- Systematische und nachvollziehbare Forschungsmethode
  - Für qualitative Daten
  - Explorativ
    - Keine vorherige Annahmen über das Ergebnis

Durchführung der Arbeit

Freie Universität Berlin

- Workshop
- Experiment
- Beginn der Auswertung/Kodieren
- 2. Experiment
- Kodieren
- Theorienbildung

Workshop

Freie Universität Berlin

- Workshop zum Thema Webentwicklung mit Java
  - 4 Tage
  - 10 Teilnehmer aus höheren Semestern
    - Komplexer Stoff
  - 5 Paare
  - Motivation für die Teilnahme am Experiment
    - „Ich gebe Euch Wissen, Ihr nehmt an meinem Experiment teil“

- Einführung in die Frameworks Hibernate, Spring und Tapestry
  - Praktische Anwendung des Erlernten in Übungen
  - 1. Tag Hibernate und Spring + Übungen
  - 2. Tag Tapestry + Übungen
  - 3. Tag Übungen
  - 4. Tag Experiment

Workshop (2)

Freie Universität Berlin

- Entwicklung einer dreischichtigen webbasierten Bibliotheksanwendung
  - Datenhaltung, Programmlogik, Präsentation
  - Medienverwaltung
  - Ausleihe
  - Rückgabe

- Probanden praktizierten schon während des Workshops Side-By-Side Programming ohne es explizit gesagt zu bekommen
  - Enge Zeitvorgaben so dass sich Trennung und parallele Bearbeitung anboten
  - Jedes Paar hatte 2 Rechner nebeneinander zur Verfügung

Experiment

- Am letzten Tag des Workshops
- Erweiterung der Bibliotheksanwendung
  - Aufgabe war umfangreich und mit guten Möglichkeiten zur parallelen Bearbeitung
- Arbeit wieder im Paar
  - 2 Rechner direkt nebeneinander
  - Blick auf den Bildschirm des anderen leicht möglich
- Aufzeichnung von Audio und Video der Probanden sowie des Bildschirminhalts

Probleme

- Völliges Neuland
  - Bisher keinerlei Wissen über Side-by-Side Programming
  - Völlig andere Abschlussarbeit als gewöhnlich
    - Normalerweise Programmierung und dazu ein bisschen Empirie
    - Hier: Fast nur Empirie
- Probleme beim Workshop
  - Unzuverlässigkeit der Probanden
    - Einer erschien nicht zum Experiment
    - Ein Paar praktizierte absichtlich nur Pair Programming
      - Hätte man auf Side-by-Side Programming bestehen sollen?
  - Problem bei der Aufzeichnung
    - Ein Video ging verloren, für ein Paar liegt also nur ein Video vor
  - Somit nur drei verwertbare Videos
    - 2. Experiment erforderlich

Probleme (2)

- Technische Probleme
  - Videokonvertierung
  - Sehr große Dateien

- Datenflut
  - Jedes Phänomen im Video zu kodieren viel zu umfangreich
  - Entwicklung eines Blickwinkels um die Kodierarbeit zu leiten

Blickwinkel

- Blickwinkel auf Trennung bzw. Zusammenkunft der Probanden

- Aktivitäten, die zu Trennung / Zusammenkunft führen
  - Aktivität ist dabei jegliche Tätigkeit, die zur Erstellung eines Softwareartefakts nötig ist
    - z.B. Programmierung, Dokumentation lesen, Aufgabenabstimmung, Fehlersuche und –behebung

- Aufgabe lässt sich in zu erledigende Schritte einteilen
  - Versuch diese Schritte in der Kodierung wieder zu finden
  - Evtl. hängt das Trennungsverhalten mit diesen Schritten zusammen

- Eigenschaften der Personen könnten Einfluss auf Trennungsverhalten haben
  - Verhältnis zueinander
  - Kenntnisse

Blickwinkel (2)

- Kodierung anhand des Blickwinkels
  - Kodiere nur, was im Blickwinkel liegt
    - Also Trennung / Zusammenkunft und was davor und danach geschieht
  - Aufbauend auf das Kodierschema von Stephan
  - Z.B.
    - P1.report_problem
    - P1.rally
      - P2 und P1 Treffen sich vor dem Rechner von P1
    - P.think aloud_problem
    - P1.thin aloud_finding
    - P2.agree_finding
    - P2.parting
    - P2.search_sth
    - P2.write_sth
    - Usw.

Weiteres Vorgehen

- 2. Experiment
  - Geeignete Probanden und Aufgabe finden
    - Aufgabe muss den Probanden die Möglichkeit geben, zu zweit parallel an ihr zu Arbeiten
      - Client/Server Anwendung, kleines Spiel
- Weiteres Kodieren der Videodaten
- Herstellung von Beziehungen zwischen den kodierten Konzepten
- Entwicklung einer Theorie über die Trennung / Zusammenkunft beim Side-by-Side Programming

Fragen / Anmerkungen

- Fragen / Anmerkungen

Vielen Dank!