

Real-Time Collaboration Eine Kostprobe Workshop

Helge Nowak

hnowak@cincom.com

Twitter: @nowagil

Softwareentwicklung heute

- Softwareentwicklung ist Teamarbeit
- Die Kerntätigkeiten sind asynchron
- Jeder arbeitet für sich
- Die Ergebnisse werden nachträglich integriert
- Das gilt auch bei Pair-Programming und Continuous Integration

Softwareentwicklung morgen

- Softwareentwicklung ist Teamarbeit
- Die Kerntätigkeiten sind synchron
- Alle arbeiten gleichzeitig und feingranular an den Artefakten (Tests, Code usw.)
- Intensive teamweite Kommunikation und Interaktion: „Pair-Programming on Steroids“
- Keine separate Integration: „Constant Sharing“

Real-Time Collaboration (RTC)

- feingranular
 - synchron
 - schnelles Wechseln zwischen Kontexten
 - „immersive“
-
- So, wie heutzutage in der Softwareentwicklung NICHT gearbeitet wird !

Der Workshop

- Simuliert in spielerischer Weise die Zusammenarbeit unter RTC-Bedingungen
- Es geht um die Zusammenarbeit im Team
- Es geht um die Zusammenarbeit zwischen Teams
- Es werden Herausforderungen gestellt, wie sie in der täglichen Arbeit vorkommen

Aufgabe und Umgebung

- Die Aufgabe simuliert die Entwicklung eines Systems, das aus relativ eigenständigen Subsystemen besteht
- Jedes Team ist für ein Subsystem zuständig
- Die Systemabhängigkeiten zeigen sich im Laufe der Arbeit
- Um keine speziellen Programmierkenntnisse voraussetzen zu müssen – und um den Spaß zu erhöhen – wird ein Multidokumentroman in Google Docs geschrieben

Aufgabe

Alle erschaffen gemeinsam

- ein literarisches Werk
 - aus mehreren Dokumenten in Google Docs
- als Autorentams
- denselben Text
 - jedes Team verantwortlich für „sein“ Dokument
 - jedes Team editiert jedes Dokument
- zur gleichen Zeit
 - Umsetzen von Anforderungen gegen die Uhr

Drei Phasen

- Jede Phase bringt neue Anforderungen an die Zusammenarbeit
- Nach jeder Phase 10 Minuten Review
- Am Ende des Workshops generelles Review und Feedback

Ergebnisse Phase I

- Transparente Kommunikation
- Leichte und schnelle Koordination
- Paralleles Arbeiten an verschiedenen Aspekten
- Es entsteht eine Kultur freiwilliger dynamischer Arbeitsteilung
 - Aufgabenteilung ohne feste Zuordnung der Rollen, Selbstorganisation

Ergebnisse Phase I

- RTC ermöglicht höhere Produktivität
- RTC resultiert in besserer Qualität
- Wesentliche Vorteile:
 - völliges Fehlen von separaten Integrations-schritten
 - vollständige Transparenz

Ergebnisse Phase II

- Fehlende Kommunikation zwischen den Teams erschwert gute Implementierung von Abhängigkeiten
 - Lokale Zuständigkeit erzeugt Inkompatibilitäten zwischen den Teilsystemen, die erst spät oder gar nicht erkannt werden
- !! Obwohl das jeweilige Fremdsystem vollständig offen vorliegt

Ergebnisse Phase II

- System- und Prozeßbrüche gehen zu Lasten der Produktivität
- Inkrementelle Entwicklung unter RTC-Bedingungen minimiert Aufwand für Konsistenz- und Qualitätssicherung

Ergebnisse Phase III

- Limitierte Kommunikation zwischen den Teams erschwert Implementierung auf allen Ebenen
- Einengende Vorgaben werden vermieden, dadurch entstehende mangelnde Qualität wird in Kauf genommen
- Zusätzlicher Abstimmungsbedarf und -aufwand geht zu Lasten der Produktivität und Qualität

Ergebnisse Workshop

- Fehlende Kommunikation und Kooperation kann nicht durch stärkere Abgrenzung zwischen den Komponenten und Teams überwunden werden
- Lösung: vollständige RTC-Bedingungen für das Gesamtsystem, Collective-Ownership über die Teamgrenzen hinaus
- Alle Teilnehmer würden gerne Real-Time Collaboration in einem echten Softwareentwicklungsszenario ausprobieren

Workshop-Feedback

„Gut durchdacht“

„Spannend und sehr lehrreich“

„Es hat Spaß gemacht!“

Das Werk

Volland im Zwielight

- **Weyden:** <http://tinyurl.com/xdde12Weyden>
- **Grooten:** <http://tinyurl.com/xdde12Grooten>

© 2012

- **Autoren:** Konstanze Steinhausen, Patrick Hund, Lars Hüper, Andreas Schildorfer
- **Idee und Vorlage:** Jason Ayers, Helge Nowak

Helge Nowak

hnowak@cincom.com

Twitter: @nowagil