Besprechung Übung 1 & Vorstellung Übung 2

Software Qualität, FS09

06.04.2009

Reinhard Stoiber

Übung 1

- Ergebnisse: grossteils gut
- Subversion (svn+ssh) auf Windows Server 2003
  - Bereitete Probleme mit Schreibrechten nach commits (Rechteproblem zwischen OpenSSH, Cygwin und Windows)
  - Besser wäre: apache server; svn ohne ssh (mit VPN Kanal); oder Verwendung eines unixoiden Servers

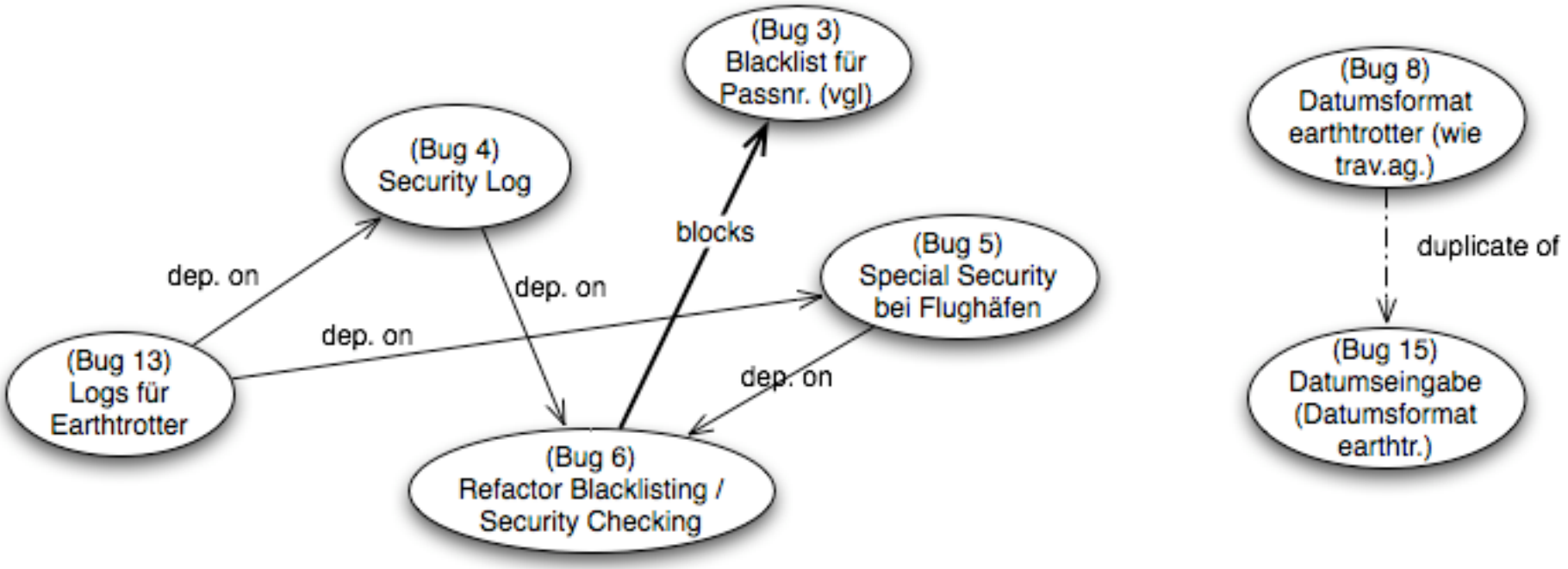
Teil A: Beziehungen zwischen den Problemmeldungen (1)

- depends on
  - dieses Problem kann/sollte nicht behoben werden ohne dass ein anderes Problem *zuvor* behoben wird
- blocks
  - die Behebung dieses Problems verhindert dass ein anderes behobenes Problem weiterhin gelöst sein kann (conflict)
- duplicate
  - es handelt sich um identische Probleme

Empfehlenswert: immer Kommentare schreiben wenn an Problembehebungen gearbeitet wurde ...

Teil A: Beziehungen zwischen den Problemmeldungen (2)

- Auszug der Beziehungen (demonstrativ):



Anm.: die Bug-Numerierung war zwischen den Gruppen verschieden

Teil B: Testen (1)

- War meist ausreichend bzw. gut gelöst
  - Ev. Probleme auf Rückgabe vermerkt

- Empfehlenswert: mehr als eine assert-Bedingung pro Test; gibt mehr Sicherheit

Teil B: Testen (2)

- Wichtige Tests:

1. korrekte Buchung im Format Earhttrotter → muss in Transaction Log;
   - auch mit Blacklist Pass und mit Security Flughafen testen
2. korrekte Buchung im Format Travelagent → muss in Transaction Log;
   - auch mit Blacklist Pass und mit Security Flughafen testen
3. einzelne Methoden der Klasse ValidityChecker
   (AirportCode, AirlineCode, Name)
4. inkorrekte Buchungseingaben (validityChecker)
   → keine Buchung, keine Logs
5. mehrmaliges Starten des Servers darf keine Exception werfen
6. Clients dürfen nur starten wenn Verbindung mit Server besteht
7. etc ...

Teil C: Verteilte Entwicklung mit SVN

- Bugfixes bzw. Implementierung der Problemmeldungen
- Unittests gewährleisten dass neuer Code die Funktionalität des bestehenden Codes nicht negativ beeinflusst (Regressionstests)
- Für neuen Code wenn möglich parallel bereits die Tests schreiben
- Neue Features: ändern oft bestehende Anforderungen; ändern in der Folge oft auch bestehende Unittests (vgl. Beziehungen zw. Problemmeldungen aus Bugzilla)

Teil D: Versionskontrolle

- Warum besser öftere und kleinere Commits?
  - Damit alle am gleichen Code arbeiten; Konflikte werden weniger und kleiner; besser synchronisiert; laufende Arbeit besser gesichert;
- Vor-/Nachteile des file lockings?
  - Nur eine Person kann gleichzeitig an einem File arbeiten. VT: keine Konflikte, NT: wenn jemand einen file lock erzeugt aber nicht am file arbeitet ...
- Erfahrungen mit Versionskontrolle?
  - viele haben bereits Erfahrungen ...

Übung 2

- Vorstellung der Übung 2
- Thema: Debugging

Übung 2 - Allgemeines

- Daten
  - Ausgabe: 06.04.2009
  - Abgabe: 20.04.2009, 24h
  - Besprechung: 04.05.2009
- Abgabe, Formales
  - Abgabe als .ZIP Datei (Dokumentation als .PDF und Source Code in separatem Verzeichnis) an stoiber@ifi.uzh.ch
  - Es kann in den Computerräumen des IFI gearbeitet werden
- Gruppen
  - Die Übung wird in 2er Gruppen ausgearbeitet

Teil A: Statische Analyse

- Gegeben: ein einfaches Programmstück in Java das eine Zeichenkette nach Zahlen, Wörtern und Zeilen analysiert
- Gesucht:
  - Alle Set-Use, Use-Set und Set-Set Beziehungen
  - Der Program Dependency Graph (PDG)
  - Der statische Backward Slice für eine bestimmte Programmzeile
  - Welche Arten von Problemen können mit Hilfe des PDG im Code erkannt werden?

Teil B: Rekonstruktion von Fehlern

- Gegeben: Bug Story für ein Programm das nur an ‘Wednesday’s funktioniert.
  - Grund: Nur 8 Bytes für den Wochentag im Speicher festgelegt und Überlauf des ‘y’ am Mittwoch ... (Siehe Text)
- Kennen Sie weitere mögliche *Einflussfaktoren* bezüglich der Reproduzierbarkeit von Fehlern, ausser dem Datum?
- Nennen Sie Beispiele ...

Teil C: Debuggen

- Gegeben:
  - eine Implementierung des Quicksort Algorithmus für lexikographische Sortierung + ein Text zum Sortieren
- Problem:
  - der gegebene Text löst im Programm eine Exception aus
- Gesucht:
  a. Finden Sie den einfachsten Input bei dem das Programm fehlerhaft ist; definieren Sie ‘breakpoints’
  b. Überlegen Sie sich die korrekten Variablenbelegungen an den breakpoints; testen Sie die gegebenen Hypothesen
  c. Benützen Sie den Eclipse Debug-Modus
  d. Verfeinern Sie zutreffende Hypothesen aus b) und finden Sie die Lösung für den Defekt
  e. Analysieren Sie den Programmierstil des Codes

Abgaben

- Ihre Lösungen zu den Aufgaben A, B und C im .PDF Format

- Danke für die Aufmerksamkeit.