Arbeitskreis „Testen objektorientierter Programme“
der GI-Fachgruppe TAV

Protokoll des Treffs am 18. Mai 2004 in Berlin (Fraunhofer Fokus)

Teilnehmerliste (unvollständig)

Michael Averstegge	
Jens Calamé	
Falk Fraikin	TU Darmstadt
Michael Gerke	HIS GmbH
Stefan Jungmayr	Teradyne DS (Protokollführer)
Roger Müller	Universität Münster
Dehla Sokenou	Universität Berlin
Andreas Schönknecht	TUI InfoTec
Mario Winter	FH Köln

Ablauf

1) Brainstorming zu typischen Fehlern in OO-Programmen
2) Arbeit in 2 Arbeitsgruppen:
   - Gruppe 1 kategorisiert die bisher zusammengetragenen OO-Fehler.
   - Gruppe 2 erarbeitet die Vorgangsweise für eine Umfrage zum Thema.
3) Gemeinsamer Abschluss mit Austausch der Ergebnisse

Kategorisierung von OO-Fehlern

Fehlerkategorien siehe Anhang in diesem Dokument.

Umfrage

Geplant sind zwei parallele Formen von Umfragen:

1. Eine gezielte Umfrage unter einschlägigen Experten und Universitätsinstituten.
2. Eine web-basierte Breitenumfrage.

Zu 1): Bereits bestehende Sammlungen bzw. Ergebnisse von empirischen Untersuchungen sollten genutzt werden.

Zu 2): Werbung ist geplant in Fachzeitschriften und Newsgroups. Der Fragebogen sollte nicht zu umfangreich sein (weniger als der Inhalt einer DIN-A4 Seite), die Häufigkeit/Kritikalität von ca. 10-15 Fehlerkategorien abfragen, und die Möglichkeit für zusätzliche Texteingaben bieten.

Ein Preisgeld soll die Teilnahme an der Umfrage attraktiver machen. Ein Motto für die Umfrage wird noch gesucht (wie z.B. “Bug-Parade” oder “Deutschland sucht den Super-Bug”.)

Aktionen bis zum nächsten Treffen

1) Alle: Literaturrecherche zu häufigen OO-Fehlern.
   Literatur im Wiki-Web (http://giserver.gi-ev.de/giwiki/) eintragen.

Nächstes Treffen des Arbeitskreises

Das nächste Treffen findet am 1.Oktober an der TU Darmstadt statt.

Anmeldung bitte bis zum 28. September bei Dr. Falk Fraikin (fraikin@ito.tu-darmstadt.de).

Weitere Infos

Weitere Informationen über den Arbeitskreis finden Sie auf folgender Webseite:
http://giserver.gi-ev.de/fachbereiche/softwaretechnik/tav/toop/

Anhang: Fehlerkategorien

1. Fehler nicht spezifisch für OO-Systeme

1.1 Konfigurationsfehler

1.2 Pointerarithmetik

1.3 Speicherverwaltung

1.4 Off-by-one

1.5 Multithreading

1.5.1 Race conditions

1.6 Nicht-nebenwirkungsfreie Operationen in Assertions

1.7 Semantik der Methode geändert, weitere Nutzer der Methode aber nicht in Kenntnis gesetzt

2. OO-spezifische Fehler

2.1 Design

2.1.1 Nicht-semantikerhaltende Vererbung

- Java "super not called" in der redefinierten Methode
- Semantikänderung von redefinierten Methoden
- Instanzvariablen überschrieben

2.1.2 Multiple inheritance

2.1.3 Kopplung

- Gegenseitige Attributreferenzen zwischen Klassen
- Zu tiefe Schachtelung von inneren Klassen

2.1.4 Kohäsion

2.1.5 Zyklische Abhängigkeiten

2.1.6 Irreführende Benamung

- Überladen von Operatoren
- Namen und Verantwortlichkeit von Methoden nicht konsistent
- "Rumprobieren" basierend auf Methodennamen (Nebenwirkung nicht berücksichtigt)
- Wildcard-Importe
- Klassen mit gleichem Namen in unterschiedlichen Paketen
- Methode fast gleichbenannt wie Klasse
- Überschreiben von nichtabstrakten Methoden durch abstrakte

2.1.7 Geheimnisprinzip

- Referenzen auf private Instanzvariablen nach außen gereicht
- Übertriebenes Geheimnisprinzip, später doch gebrochen (z.B. work-around durch Redundanz)
- Vaterklasse kennt erbende Klassen
- Schlechte Kapselung (nicht-private Attribute)

2.1.8 Zu komplexe Zustandsräume

- Dialoge

2.2 Implementierung

2.2.1 Unzulässige Annahmen über Aufrufreihenfolge

- Aufteilung in Konstruktur und Init

2.2.2 Objektinitialisierung

- Initialisierungsreihenfolge bei Unterklassen mit abstrakten Oberklassen
- Verhältnis von Konstruktoren und Destruktoren
- Referenzen auf nicht mehr gebrauchte Objekte (nicht freigegeben)
- Schlüsselwort virtual für Destruktor vergessen

2.2.3 Exception Handling

- Versteckter Kontrollfluss
- Finally vergessen
- Leeres Exception-Handling

2.2.4 Unbenutzer Code

2.2.5 Reflection

- Statische Informationen und Refactoring

2.2.6 Enumerations

- Abh. Klassen nicht neu kompiliert

2.2.7 Deep copy

2.2.8 Casting

- Überschreibende Methode nicht verwendet (C++)
- Methoden die Typ "Object" zurückliefern

2.2.9 Mangelnde Kenntnis der Sprachdefinition

- CompareTo (mit Strings) nicht auf +1 oder -1 vergleichen
- Klassenmethoden werden in der Generalisierungshierarchie anders behandelt als Instanzmethoden