GI-2.1.7. TAV AK-TOOP

Protokoll des Treffs am 18. Februar 2001 in Frankfurt

http://www.informatik.fernuni-hagen.de/import/pi3/GI/akOOT.html

Ort

Dresdner Bank in Frankfurt/Main

Anwesende

Michael Brunner, Dresdner Bank,
Falk Fraikin, TU Darmstadt,
Bernhard Gehrke, Software Quality Engineering,
Stefan Jungmayr, FernUniversität Hagen,
Moritz Schnitzler, RWTH Aachen,
Ralf Wendtland,
Mario Winter, FernUniversität Hagen.

Ablauf

9^{30} - 10^{00} Uhr Kurzvorstellung der Teilnehmer, Aktualisierung der Agenda,
10^{00} - 10^{20} Uhr Vortrag M. Brunner: Das Projekt OSKAR der Dresdner Bank
10^{20} - 11^{30} Uhr Vortrag M. Schnitzler: Testen Rahmenwerksbasierter Programme
11^{30} - 12^{30} Uhr Diskussion
12^{30} - 15^{00} Uhr Spezifikationsorientierte Testendekriterien für Komponenten.

Vortrag M. Brunner: Das Projekt OSKAR der Dresdner Bank

oskar (online strategic position keeping and reporting) ist ein weltweit führendes IT-Projekt. Das Ziel des Projekts ist es, das bestehende System für die Umsatzverarbeitung, Buchung und Bestandsführung der Dresdner Bank von Grund auf zu erneuern. Dies ist die Voraussetzung für Beratungs- und Geschäftsprozesse in Echtzeit und damit für die Wettbewerbsfähigkeit von morgen. Das neue Kernsystem wird in Analogie zur modernen Industrieproduktion als Realtime Information Factory gebaut.

Rohstoff:

- Geldumsatzinformation

Produktionsaufträge:

- Disponieren und Buchen von Umsätzen
- Bereitstellen von angereicherten Umsatzinformationen

Produktion:

- hochautomatisiert und standardisiert
- modular aufgebaute Fertigungsstraßen

Produktionsergebnis:

- kosteneffiziente Realtime-Buchungen
- Neartime-Umsatzinformationen

Literatur: Gero Scholz, Markus Warg: Großprojekte erfolgreich managen. Mit der Informationsfabrik OSKAR in die IT-Echtzeit. Gebundene Ausgabe, 325 Seiten, FAZ Verlag, 2001, ISBN: 3898430332

Vortrag M. Schnitzler: Testen Rahmenwerksbasierter Programme

Die Konzepte der objektorientierten Programmierung, wie beispielsweise die Vererbung, fördern die Wiederverwendung existierenden Codes. Ein Programm wird typischerweise auf Basis eines Rahmenwerks implementiert, welches bereits seine grundlegende Struktur und Funktionalität vorgibt. Auf diese Weise entsteht eine Familie unterschiedlicher Programme für verschiedene Anwendungsfelder, die auf dem gemeinsamen Rahmenwerk aufbau-

en. Müssen die Programmvarianten jedoch getestet werden, bieten die existierenden Techniken kaum Möglichkeiten, dieselben Testfälle für verschiedene Varianten zu verwenden. Andererseits legt das gemeinsame Rahmenwerk das grundlegende Verhalten aller Programmvarianten in Form charakteristischer Objektkollaborationen fest. Werden Testfälle auf diesen Kollaborationen aufgebaut, können diese als Grundlage für einen Test aller Programmvarianten dienen. Die Modellierung mit Rollen hat sich als leistungsfähiges Mittel erwiesen, um solche Kollaborationen zu identifizieren und ihr erwartetes Verhalten zu spezifizieren. Unser Ziel ist es, aus den so modellierten, charakteristischen Kollaborationen des Rahmenwerks Testfälle abzuleiten, und mit diesen einen Prüfstand zu realisieren, der einen automatischen Test der zentralen Elemente jeder einzelnen Programmvariante erlaubt.

Spezifikationsorientierte Testendekriterien für Komponenten

Folgende Gruppen und deren Eigenschaften existieren

Funktional:

Tabelle 1: Funktionale Kriterien

Spezifikation/ Diagrammtyp	Charakter	Ausprägung	Überdeckungskriterien
Verträge (Contracts)	Formale Sprache	OCL	Bedingungsüberdeckung
	Formale Theorie	Prädikatenlogik	-“-
	Freitext	API	“Alle Funktionen“?
Zustands-diagramm			Zustandüberdeckung, Übergangs-überdeckung, n-Path Überdeckung
Interaktions-diagramm		SDL MSC, UML-Sequenz-diagramm	Knoten/Zweigüberdeckung, Pfad-überdeckung
		Kollaborations-diagramm	
Anwendungs-fälle	Freitext	Jacobson	Hauptablauf, alle Ausnahmeabläufe
		Aktivitätsdia-gramm	Knoten/Zweigüberdeckung, Pfad-überdeckung
		Sequenzdia-gramme	Knoten/Zweigüberdeckung, Pfad-überdeckung

Nicht Funktional (Refactoring, sog nicht messbare Kriterien):

- Performance
- Zuverlässigkeit
- Sicherheit
- Wartbarkeit

Weiterhin diskutierten wir folgende Punkte.

Wer spezifiziert Testfälle?

- Anbieter
- Kunde

Built-In Tests in Entwicklungsumgebungen?

- Contract Checking
- Coverage

Testverfahren. In den meisten Fällen wird man sich auf Black-Box Tests beschränken. Mit State-Charts sind, aufgrund kombinatorischer Explosion, nur kleine Komponenten testbar. Da Spezifikationen heute meist mit USE-

Cases und Contracts ausgeliefert werden, bietet es sich an, auf dieser Grundlage mit der Spezifikation der Testfälle aufzusetzen. State-Charts gehören in der Regel nicht zum Lieferumfang.

Exeption - Contracts. Wie können erworbene Komponenten gegen Spezifikation und im Zusammenspiel mit anderen Komponenten getestet werden?

- Enterprise Java Beans
- Corba-Componenten

Wichtige Fragen sind: Erfüllen die gelieferten Komponenten die geforderten Qualitätsansprüche z.B. in Bezug auf Persistenz? Werden State-Charts mitgeliefert, die das Verhakten der jeweiligen Komponente nach außen hin hinreichend beschreiben?

Wie lassen sich unterschiedliche Spezifikationen abgleichen? (Kunde versus Anbieter) Antwort: Komponenten testen! Die erstelle Testspezifikation ist dann eine (die) gemeinsame Spezifikation.

Testumgebung. Für Komponenten sind automatische (built in) Vor- und Nachbedingungstests durchzuführen. Diese können während des Testbetriebs eingeschaltet werden, sind dann aber für den Wirkbetrieb auszuschalten.

Werden Fehler der Komponenten erkannt, die Aufgrund von Lücken in der Spezifikation entstanden sind, so könne diese im Betrieb durch Wrapper gefiltert werden, falls es nicht möglich sein sollte, die Fehler in den betroffenen Komponenten rechtzeitig zu beheben.

Generische Testumgebung: Exeptions in Komponenten lokal eingrenzen, Systemverhalten durch Tracing mitschreiben.

Bei Active X: Wenn Komponenten an einen Kunden ausgeliefert werden, ist es sinnvoll Kotrollflussgraphen mit auszuliefern, welche die Sicht von außen auf die Funktionalität des Produkt hinreichend beschreiben. Hinreichend beschrieben werden muss die jeweilige Komponente auch in Bezug auf durchzuführende Test; nicht nur in Bezug auf die Integration in vorhandenes System.

Diskutiert wird in QS-Kreisen derzeit die Einführung eines Certification Labels für Komponenten, unklar ist bisher, wer dieses Erteilt und welche Kriterien für dessen Erteilung erfüllt werden müssen.

Nächstes Treffen des Arbeitskreises

Das nächste Treffen des AK-TOOP findet anlässlich des 17. Treffens der Fachgruppe (voraussichtlich vom 18.-19. Oktober 2001) in Nürnberg oder Umgebung statt. Wir sind Gast beim ASQF (Arbeitskreis Software-Qualität Franken) und bei der GI-Regionalgruppe Nürnberg, Fürth, Erlangen

Themenvorschläge:

- Testbarkeit und Test von COM+/CCM/ejb-Komponenten

Bitte per eMail bis zum 01. September 2001 bei Mario Winter anmelden (Mario.Winter@FernUni-Hagen.de, ggf. inkl. weiteren Themenvorschlägen).