Nutzungsorientiertes Statistisches Testen

Albrecht Beck
Alcatel SEL AG
Lorenzstr. 10, D-70435 Stuttgart
A.Beck@alcatel.de
http://www.alcatel.de/projekte/usst

Einleitung

Das Testen eines jeden realen Softwaresystems leidet unter einem grundsätzlichen Problem: Das „Kosten-Abdeckungs-Problem“: Erschöpfendes Testen ist aufgrund der enormen Komplexität moderner Systeme unmöglich. Ein vielversprechender Ansatz ist statistisches Testen der Nutzung des Systems, eine Methode die ursprünglich aus der „Cleanroom“-Methode stammt.

Das wesentliche Ziel unseres Experimentes war, die Anwendbarkeit im Telekommunikationsbereich zu testen. Die Zuverlässigkeit sollte im Vergleich zu bisherigen Software Releases verbessert werden. Durch Konzentration der (immer zu knappen) Testzeit auf die erwartete Nutzung hilft USST das Kosten-Abdeckungs-Problem zu überwinden. Eine Vorhersage der Zuverlässigkeit des Systems sollte möglich sein.

Grundidee

Entwickler und Anwender nehmen die Zuverlässigkeit des Systems unterschiedlich wahr: Entwickler neigen dazu, Fehler zu zählen und daraus Fehlerdichten zu berechnen, während Kunden im wesentlichen an einer niedrigen Ausfallrate interessiert sind. Diese Kundensicht muß beim Entwurf der Testfälle berücksichtigt werden: Es genügt im Integrations- und Systemtest nicht, nicht - und ist wahrscheinlich auch gar nicht nötig - volle Abdeckung des Programmcodes durch Testfälle zu erreichen. Vielmehr ist es nötig, die Nutzung des Systems abzudecken. In allen realen Systemen ist es jedoch nur möglich, eine Stichprobe zu testen. Es gibt also zwei Typen von Testfällen: Die einen führen wir selbst aus, die anderen nicht (bzw. überlassen sie unseren Kunden ...). Leider wird die Zuverlässigkeit des ausgelieferten Systems durch die letzteren bestimmt.

Die Zuverlässigkeit des Systems wird durch die Wahrscheinlichkeit bestimmt, daß einer der nicht ausgeführten Testfälle fehlschlägt. Um diese Wahrscheinlichkeit schätzen zu können, ist eine statistische Stichprobe erforderlich. Es ist wenig sinnvoll, alle möglichen Testfällen zu erzeugen und daraus eine Stichprobe auszuwählen, die tatsächlich ausgeführt wird. Ebenso unsinnig ist es, vollkommen zufällig Testfälle aus allen möglichen Eingaben zu erzeugen. Wir verfolgen vielmehr den Ansatz, ein Modell der Nutzung des Systems zu erstellen und daraus Testfälle zu generieren, die sowohl realistisch als auch zufällig sind.

Die Modellierung erfolgt mit Markov-Ketten. Eine Markov-Kette kann als Zustandsdiagramm angesehen werde, in dem jeder Zustandsübergang eine bestimmte Wahrscheinlichkeit hat. Diese Notation ist den Software-Ingenieuren vertraut und intuitiv verständlich. Dabei ist zu beachten, daß die *Nutzung* des Systems und nicht das interne Verhalten der Software modelliert wird. Aus diesem Modell können unmittelbar Testfälle erzeugt werden.

Testplanung

Im allgemeinen ist es nicht möglich, die Nutzung eines Systems mit einer einzigen Markov-Kette zu modellieren - das entstehende Modell wäre nicht mehr handhabbar. Daher steht am Anfang etwas Planung:

- *was* soll getestet werden
- *wann* kann es getestet werden (d.h. alleine oder zusammen mit anderen Tests)
- *wie* wird das Testbudget auf die Tests verteilt
- *wo* lohnt sich Modellierung, wo nicht.

Zunächst werden alle Quellen von Eingaben identifiziert und klassifiziert. Anschließend werden die Systemfunktionen bestimmt und klassifiziert. Schließlich wird entschieden, welche Testmodelle tatsächlich konstruiert werden und wie das vorhandene Testbudget verteilt wird.

Modellentwicklung

Nutzungsmodelle werden durch Markov-Ketten repräsentiert. Markov-Ketten können als gerichteter Graph dargestellt werden. Die Zustände müssen der Markovschen Bedingung genügen: alle Information zur Bestimmung der Übergangswahrscheinlichkeiten zu einem zukünftigen Zustand muß vollständig in dem Zustand enthalten sein.

Wir können verschiedene Typen von Nutzungsmodellen unterscheiden:

- Im *Stimulus-Modell* entsprechen die Zustände Systemeingaben. Dieses Modell ist nützlich, um z.B. Initialisierungsdateien zu erzeugen.
- Im *Nutzungsvariablen-Modell* sind die Zustände Kombinationen von Nutzungsvariablen. Die Übergänge beschreiben Systemeingaben. Dies ist das passende Modell für interaktive Anwendungen.
- Im *Antwort-Modell* korrespondieren die Zustände mit Systemreaktionen, die Übergänge beschreiben Eingaben. Dieser Modelltyp gibt eine erste Annäherung für Systeme mit sehr vielen Zuständen als erster Schritt auf dem Weg zu einem Nutzungsvariablen Modell.

Wir fanden es zweckmäßig, zunächst die Struktur der Modelle zu erzeugen. Diese Modelle werden dann

einem Review unterzogen. Die Übergangswahrscheinlichkeiten werden erst zugewiesen, wenn die Struktur des Modells fertig ist. Die Wahrscheinlichkeiten werden so realistisch wie möglich gewählt, idealerweise sollten sie auf Messungen und Daten vom Kunden basieren. Bei neuen Leistungsmerkmalen ist man allerdings auf Schätzung angewiesen.

Mehrere Leistungsmerkmale einer ISDN-Vermittlungsstelle wurden modelliert. Für ISDN Leistungsmerkmale ist das Nutzungsvariablen-Modell passend..

Eine statische Analyse der Modelle ist möglich: Erwartungswerte für die Länge der Testfälle und die Anzahl der für eine vollständige Abdeckung aller Zustände bzw. Übergänge benötigten Testfälle.

Teile und herrsche - der Umgang mit der Komplexität der Modelle

Leider neigen Nutzungsmodelle dazu, rasch sehr groß zu werden. Mehrere Ansätze sind möglich, um mit der Komplexität umzugehen.

Abstraktion

Abstraktion bedeutet Konzentration aufs Wesentliche. Details können fortgelassen und während der Ausführung der Tests ergänzt werden. Wenn Testautomatisierung beabsichtigt ist, bedeutet das eine Menge Nacharbeit. Abstraktion funktioniert am besten bei manueller Testausführung.

Aufteilen in mehrere Modelle

Für unterschiedliche Aspekte des Systems werden unterschiedliche Modelle erzeugt.

Hierarchische Modelle

Der hierarchische Ansatz ist sehr mächtig: Zunächst wird die Nutzung des Systems auf einer hohen Abstraktionsebene beschrieben. In weiteren, rekursiven Schritten, werden die Zustandsübergänge zu Untermodellen expandiert.

Im Moment fehlt allerdings ein Werkzeug, das Testfälle aus hierarchische Modellen generieren kann. Es ist also nötig, entweder die Modelle oder die Testfälle zu expandieren.

Ein Sonderfall der Expansion von Testfällen ist die Kombination von Markov-Ketten mit Grammatiken. Dabei werden die Zustandsübergänge durch Grammatiken beschrieben. Dieser Ansatz erwies sich bei der Modellierung der Bedienerschnittstelle als besonders nützlich.

Funktionale Tests

Das Ziel von funktionalen Tests ist es die Korrektheit der implementierten Funktionen zu überprüfen. Eine Menge Fehler wird noch gefunden, Fehlerkorrekturen sind häufig. Dies macht die Anwendung der Markov-Theorie zur Berechnung der Zuverlässigkeit nutzlos. Dennoch sind nutzungsorientierte Tests hier nützlich: Testabdeckung ist ein wichtiges Problem bei funktionalen Tests. Abdeckung des Programmcodes (C0, C1) ist in dieser Testphase bedeutungslos. Wir definieren Testabdeckung als Abdeckung des Nutzungsmodells (Zustände und Übergänge). Dies ergibt eine objektive Metrik für den Testfortschritt, die aussagekräftiger als ist als z.B. „95% der Testfälle erfolgreich“. Um eine vollständige Abdeckung des Modells zu erreichen, sind i.A. mehr statistisch generierte Testfälle erforderlich als ausgeführt werden können. Es zeigt sich jedoch, daß unter den statistisch generierten Testfällen sehr viele Duplikate sind, die nicht mehrfach ausgeführt werden müssen. Dies reduziert die Anzahl der nötigen Testfälle auf ein erträgliches Maß.

Zertifizierungstest

Das Ziel des Zertifizierungstests ist die Zuverlässigkeit des Systems zu messen, nicht Fehler zu beheben. Er sollte erst begonnen werden, wenn eine hinreichende Stabilität des Systems erreicht ist. Der Zertifizierungstest dient als Entscheidungsgrundlage für die Freigabe des Systems.

Die Markov-Theorie erlaubt eine präzise Berechnung der Zuverlässigkeit. Die berechnete Zuverlässigkeit wird gegen die Anzahl der ausgeführten Testfälle aufgetragen. Dieses Diagramm macht den Testfortschritt sichtbar. Solange keine (oder nur wenige) Fehler auftreten, nimmt die Zuverlässigkeit zu. treten jedoch Fehler auf, nimmt die Zuverlässigkeit ab. Wenn die Zuverlässigkeit konvergiert, kann das Testen beendet werden. Ist die errechnete Zuverlässigkeit gut genug, kann die Software ausgeliefert werden. Andernfalls ist Nacharbeit erforderlich.

Ergebnisse

Das Experiment hat gezeigt, daß typische Leistungsmerkmale einer ISDN-Vermittlungsstelle mit „flachen“ Markov-ketten modelliert werden können. Die Modellierung führte zu einem wesentlich besseren Verständnis der Leistungsmerkmale durch die Tester. Die aus den Modellen generierten Testfälle erwiesen sich als etwa 30% effektiver als „traditionell“ erzeugte Tests. Die Fehler werden darüber hinaus früher entdeckt: der Anteil der beim Kunden gefundenen Fehler konnte um 40% reduziert werden. Die wenigen Probleme auf waren auf Wechselwirkung mit anderen, traditionell getesteten Funktionen zurückzuführen.

=> USST lohnt sich!

Der gewählte Ansatz zeigte aber auch praktische Grenzen: Mehrbenutzer-Modelle, Wechselwirkung von Leistungsmerkmalen und komplexe Eingaben führen rasch zu explodierenden Modellen. Wir fanden eine praktikable Lösung für komplexe Eingaben (Kombination mit Grammatiken), Mehrbenutzer-Modelle und Wechselwirkung von Leistungsmerkmalen erfordern fortgeschrittene Modellierungstechniken.

Literaturhinweis

Eine ausführliche, englische Fassung des Beitrags findet sich unter http://www.alcatel.de/projekte/usst