Manuelles Testen großer industrieller Systeme

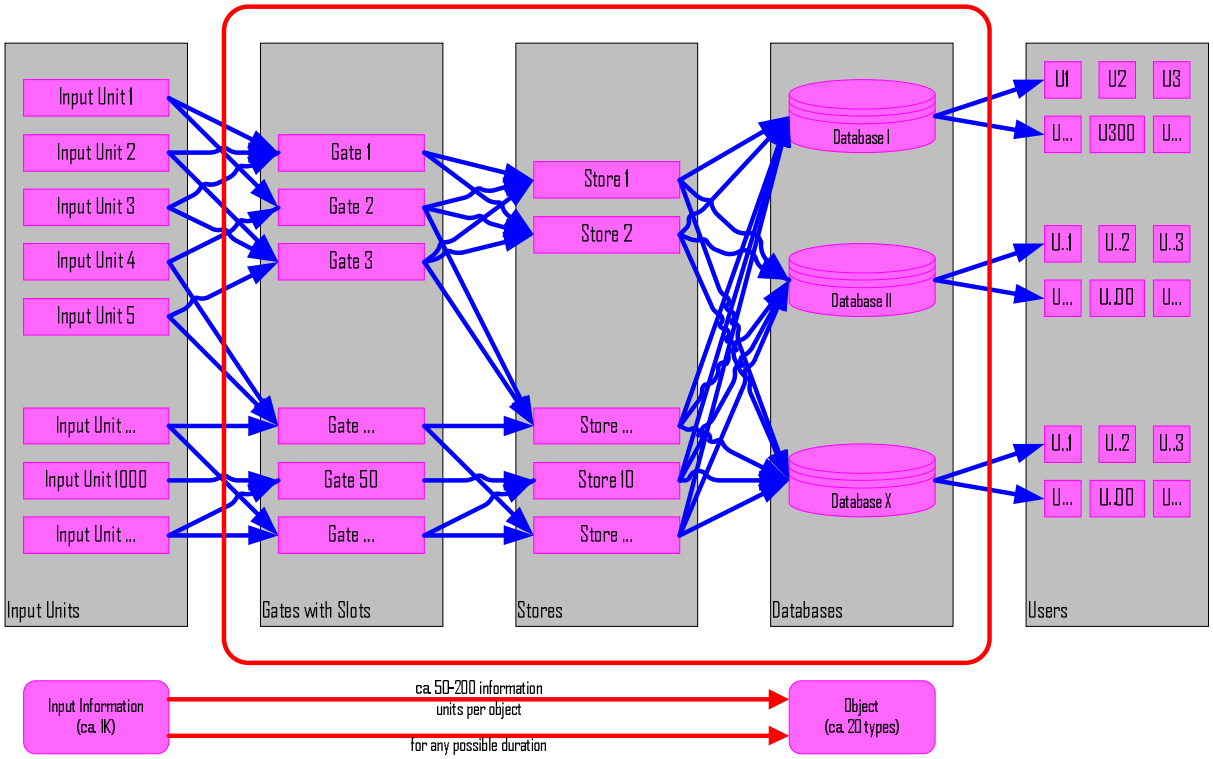
Dr. Uwe Doetzkies
Informatik für die Industrie
Berlin

In Kürze

Nichtfunktionale Anforderungen an große Softwaresysteme lassen sich in der Regel noch nicht automatisch testen. Der Autor präsentiert einen Erfahrungsbericht, wie manuelle Tests geplant, vorbereitet, durchgeführt und ausgewertet werden. Ziel ist es, durch die Verallgemeinerung dieser und anderer Erfahrungen zur Automatisierung solcher Tests beizutragen.

Anforderungen

Ein großes System ist ein System, das aus einer Vielzahl von Eingängen, internen Objekten und Verarbeitungsstationen besteht und das zu jeder Zeit eine Vielzahl von Informationen erhält:



An ein solches System werden Anforderungen der Art gestellt:

- das System muss stets arbeiten, auch wenn Teile davon ausfallen
- aus beliebigen Fehlersituationen muss das System selbständig wieder herauskommen
- auch in Überlast- und Fehlersituationen muss das System definiert arbeiten
- ausgefallene Systemkomponenten sollen möglichst automatisch wieder anlaufen

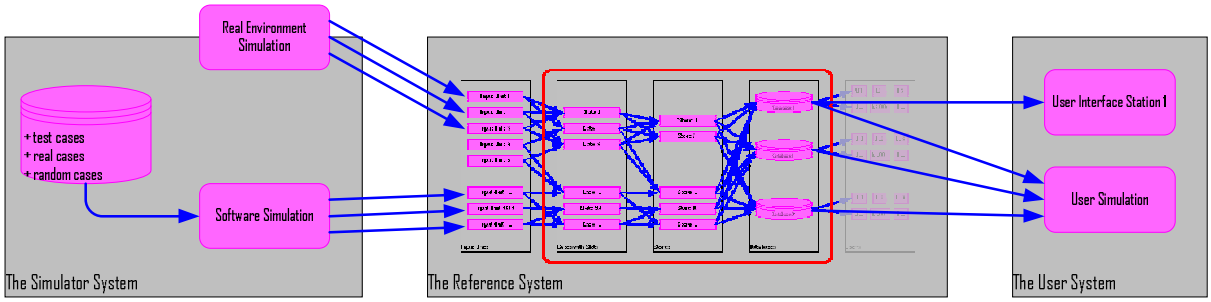
Üblicherweise werden solche Anforderungen mittels gezielter Redundanz gelöst, so wird z.B. dafür gesorgt, dass Informationen das System an mehr als einem Input-Knoten erreichen. Daraus ergaben sich weitere Anforderungen:

- jede Information soll mindestens einmal das System erreichen
- eine Information, die einmal das System erreicht hat, darf nicht mehr verloren gehen
- mehrfache Informationen werden nur einmal gespeichert

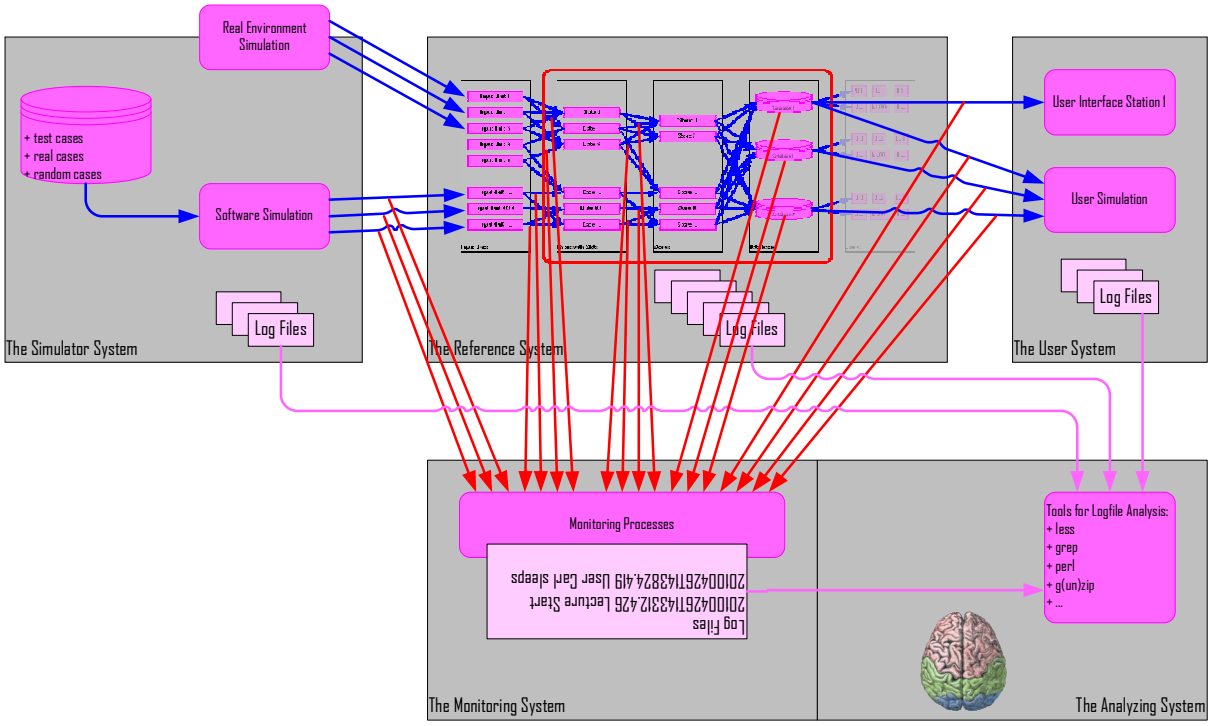
Durch die dem System innewohnende Redundanz können Störungen häufig kompensiert werden. Ungeachtet dessen sind sie natürlich unerwünscht und sollten durch Tests gefunden werden.

Nicht-automatisierbare Tests

Alle Tests, die die richtige Verarbeitung und Speicherung von Eingangsdaten nachweisen sollen, sind in der Regel automatisierbar.
Dagegen sind Tests, die die Funktion des Systems bei, während und nach Störungen unter verschiedenen Lastbedingungen nachweisen sollen, nur schwer oder gar nicht mit vertretbarem Aufwand automatisierbar. Zum einen liegt das an der Vielzahl der Eingangsdaten, die für diese Tests benötigt werden, zum zweiten an der Vielzahl der möglichen Störungsszenarien (der Strom kann in jedem Programmzustand ausfallen). Zusätzlich sind Störungen in der Regel nicht reproduzierbar.
Für solche Tests wird ein Referenzsystem verwendet, das die Haupteigenschaften des realen Systems besitzt, jedoch nicht so dimensioniert ist wie dieses.



Zum Referenzsystem gehört ein (Input-)Simulator, der eine beliebige Last erzeugen kann. Die simulierten Informationen können dabei Testfälle sein, zufällig erzeugt werden oder aus realen Situationen abgeleitete Informationen enthalten.
Ausgangsseitig existieren ein Simulator, der eine gewisse Menge von Benutzern simulieren kann (incl. einfacher Use Cases), sowie einige Benutzerstationen, die der eigenen visuellen Kontrolle dienen und für manuelle Tests benutzt werden können.
Ziel dieser Tests ist es nicht nur, die Funktion des Gesamtsystems nachzuweisen, es soll auch die korrekte Funktion jeder einzelnen Komponente nachgewiesen werden. Eine ständig ausfallende Komponente kann vielleicht von anderen Komponenten kompensiert werden, stellt jedoch einen Fehler dar.
Daher müssen es die Tests erlauben, die Funktion jeder einzelnen Komponente zu prüfen. Das heißt nicht, dass in jedem Test jede Komponente überwacht werden muss, aber es muss die prinzipielle Möglichkeit dazu bestehen. Für die Überwachung wird ein eigenes System an das Referenzsystem angeschlossen.



Mit Hilfe des Monitoring Systems können die Informationen an allen (beobachtbaren) Schnittstellen aufgezeichnet werden, Datenbankinhalte können periodisch abgefragt werden oder Benutzeraktionen aufgezeichnet werden. So instrumentiert können die Tests auf dem Referenzsystem beginnen.
Zu beachten ist jedoch, dass jede Beobachtung auch einen Einfluss auf das beobachtete System hat. So können Schnittstellenbeobachtungen die Ausführung bestimmter Prozesse verlangsamen, erhöhte Log-Level können dazu führen, dass die lokalen Speicherressourcen erschöpft werden und Datenbankabfragen könnten andere Datenbankzugriffe blockieren.

Ablauf eines Tests

Der Test untergliedert sich in drei Phasen: die Vorbereitung, die Durchführung und die Auswertung.
Die **Testvorbereitung** umfasst folgende Aufgaben:

- Einrichten des Referenzsystems (Initialzustand festlegen, Log-Level der Komponenten festlegen)
- Einrichten der Beobachter (wo, was und wie detailliert muss beobachtet werden?, Erstellen von Beobachtungsskripten, Skript für Statusbeschreibung)
- Einrichten der UI-Beobachtung (Wo lassen sich Fehler am einfachsten erkennen?)
- Einrichten der Simulation (Simulationskripte erstellen für den Vortest und den Haupttest)
- UI-Aktionen vorbereiten (Planung der Aktionen, Nutzer/Rechte/Ordner einrichten)
- Vorbereiten von Systemaktionen (Planung der Aktionen, mögliche Konsequenzen auch außerhalb des Referenzsystems)

Während der Durchführung des Tests sind alle unternommenen Schritte mit Uhrzeit zu protokollieren. Wenn dabei im Rahmen von Systemaktionen einzelne Komponenten gestört werden, so ist darauf zu achten, dass sich die Zeitpunkte der Störung/Entstörung deutlich voneinander unterscheiden, so dass in den Log-Dateien

protokollierte Ereignisse eindeutig einem Systemzustand zugeordnet werden können.
Die **Testdurchführung** besteht aus zwei Hauptschritten:

- im Vortest wird geprüft, ob alle eingerichteten Beobachter arbeiten und ob auch das System in der Weise arbeitet, die für den eigentlichen Test notwendig ist. Im Prinzip wird die korrekte Durchführung der Testvorbereitung geprüft.
- Der Haupttest enthält dann die eigentlichen Tests.

Vor, zwischen und nach den Tests soll das Referenzsystem „zur Ruhe“ kommen, in diesen Phasen werden weder vom Simulator noch auf der User-Interface-Seite Informationen in das Referenzsystem eingegeben, die zu einer Statusänderung führen.
Der (statische) Status des Referenzsystems wird in dieser „Ruhe“ bestimmt.
Bestandteil des Haupttests sind i.d.R. Eingaben am User Interface bzw. Beobachtungen desselben. Auch hier sind alle Tätigkeiten und alle besonderen Beobachtungen zu dokumentieren. Wenn möglich sollten Screenshots besonderer Situationen angefertigt werden.
Die Ziele der **Testauswertung** bestehen in der Erkennung fehlerhafter Situationen und ihrer Ursachen. Idealerweise lässt sich aus den so gewonnenen Erkenntnissen ein einfacher Testfall konstruieren, mit dem der Fehler reproduziert werden kann und der zur Prüfung der Fehlerbeseitigung verwendet werden kann.
Grundlage der Auswertung sind die Log-Dateien, deren Gesamtgröße in einem Test schon mal über einem Gigabyte liegen kann. Im Vergleich dazu sind die Textfassungen von Goethes Faust I+II mit 540 kB oder des Alten und Neuen Testaments auf Schwedisch mit 4,7 MB (im Projekt Gutenberg) verhältnismäßig kurze Werke. Aber leider muss man die Log-Dateien in wesentlich kürzerer Zeit analysieren als die schöngeistige Literatur.
Ohne geeignete Werkzeuge ist das nicht zu schaffen (der Drucker ist übrigens ein absolut ungeeignetes Werkzeug). Die wichtigsten Werkzeuge sind noch immer die einfachsten:

- **grep** hilft, die Dateien zu finden, in denen bestimmte Informationen enthalten sind
- **less** hilft, diese Dateien anzusehen und schnell darin zu navigieren (vor und zurück) unabhängig von der Größe der Datei
- **perl** hilft, wesentliche Informationen aus den Dateien zu extrahieren und in einer für die weitere Auswertung geeigneten Weise neu zusammenzustellen.

Zumindest während der ersten Phase der Testauswertung sollte sich das Referenzsystem in dem Zustand befinden, den es am Ende des Tests hatte, es sollen also keine neuen Inputs in das System erfolgen. Dies erlaubt es zum einen, Fehler detaillierter zu untersuchen, zum anderen besteht die Möglichkeit, dass sich Fehlersituationen selbsttätig bereinigen, weil z.B. einige länger laufende Timeouts dafür sorgen.

Wunschzettel

Den größten Aufwand der manuellen Tests nimmt die Testauswertung ein, die Testvorbereitung wird im Laufe der Zeit zur Routine. Wünschenswert ist es, dass Aufgaben beider Phasen durch Werkzeuge teilweise oder vollständig **automatisiert** werden:

- Herstellen des Grundzustandes
- Aufzeichnen des momentanen Zustandes
- Beobachter aus Schnittstellenbeschreibung generieren

- Erkennen von Fehlerzuständen während des Tests
- Erkennen von Fehlerindizien in Logfiles
- Automatisierte Bewertung des Zustandes von Systems und von Komponenten über die Zeit incl. geeigneter Darstellung

Eine vollständige Automatisierung erscheint jedoch nur dort sinnvoll, wo dieselben Tests immer auf identischen Systemen durchgeführt werden müssen. In der Regel unterscheiden sich die Systeme jedoch mehr oder weniger voneinander und auch die Ziele der Tests differieren. Oft ist dann eine Unterstützung der Tätigkeiten sinnvoller als die Automatisierung derselben.
Auch für die **manuelle** Auswertung sind Werkzeuge denkbar:

- Logfile-Navigator zur Unterstützung der Navigation und Auswertung der Log-Dateien
- Verbindung mit Testtools z.B. zur Erkennung und Bewertung von Testfällen in den Logdateien
- Generierung von Sequenzdiagrammen oder anderen Darstellungen aus den Logfiles für Systeme, Teilsysteme, begrenzte Zeiträume oder andere beliebige Aspekte

Auch wenn die Tools zum Testen immer umfangreicher und mächtiger werden, manche Tests entziehen sich der Automatisierung (und sei es nur deshalb, weil das Formulieren der konkreten Testbedingungen länger dauert als der eigentliche Test). Und im Laufe der Zeit werden das nicht die einfachsten Tests sein, sondern eher die komplexer Szenarien in großen Systemen. Der vorliegende Erfahrungsbericht soll Anregungen geben, wie solche Tests am besten organisiert und durchgeführt werden und an welchen Stellen dabei noch Unterstützung durch (noch nicht existierende) Werkzeuge benötigt wird.
Der Autor ist gern bereit, seine Erfahrungen für die Entwicklung und Erprobung solcher Tools zur Verfügung zu stellen. Vielen Dank.