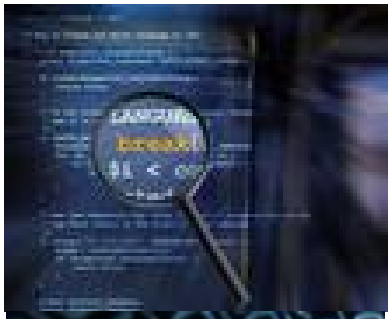

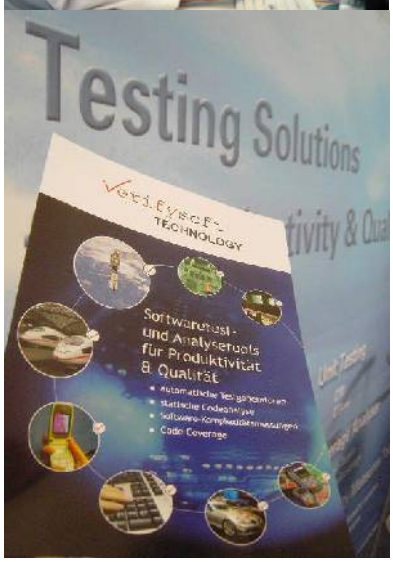

Softwarefehler verursacht das teuerste „Feuerwerk“ aller Zeiten

Softwaretests können in Europa über 100 Milliarden Euro jährlich einsparen

von Klaus Lambertz, Verifysoft Technology GmbH

„In 300 Metern bitte wenden“ zeigt das Navigationsgerät mitten auf der Autobahn an, Windows wird blau oder die Software ihres Handys hängt sich beim Fotografieren auf. Ursache sind meist Softwarefehler.
Seit in den 1940er-Jahren erstmals eine Motte (engl. Bug) ein Schaltrelais eines Computers störte, wird das Fehlverhalten von Computerprogrammen im Fachjargon als Bug bezeichnet. Insekten spielen in der heutigen Softwaretechnik keine Rolle mehr. Programmierfehler werden durch den Menschen verursacht und sind leider nicht immer harmlos. Bugs können Telefonnetze lahm legen, Raketen zum Absturz bringen oder Menschenleben kosten.

Anfang des Jahres konnten T-Mobile-Kunden für mehrere Stunden mit ihren Handys weder telefonieren noch SMS verschicken. Ursache war ein Problem im Homelocationregister, welches die Telefonnummern den einzelnen SIM-Karten zuordnet.
Die europäische Trägerrakete Ariane 5 wurde bei ihren ersten Flug 1996 zerstört, da die Ingenieure die Software der Vorgängerrakete ohne ausreichende Prüfung übernommen hatten. Das aus der Ariane 4 übernommene Modul zur Konvertierung von Zahlenwerten konnte die Berechnungen für die schnellere Ariane 5 nicht mehr bewältigen und brachte den Bordcomputer zum Absturz. 40 Sekunden nach dem Start explodierte die Rakete. Ein einziger Softwarefehler verursachte das mit etwa 500 Millionen Dollar Materialschäden wohl „teuerste Feuerwerk aller Zeiten“.
Nicht in Zahlen fassen lassen sich Verletzungen und Menschenleben. Ein Programm zur Berechnung von Bestrahlungen im Medizinbereich gab im Jahr 2000 inkorrekte Werte aus, was zum Tod von acht Patienten und zu etwa zwanzig Schwerstverletzten führte.

Leider handelt es sich bei fehlerhafter Software nicht um Einzelfälle. Die in der Industrie eingesetzten Programme entsprechen sehr oft nicht den Anforderungen. Konsequenzen sind mangelnde Produktivität, Rückrufaktionen und Rufschädigungen.

Software im Auto so komplex wie in einer Saturn V -Rakete

Software finden wir heute in fast allen Bereichen. Die Komplexität der Programme verdoppelt sich etwa alle 18 Monate. Mit durchschnittlich 50 Steuergeräten wie ABS und EPS ist ein Mittelklassewagen heute mit so viel Software bestückt wie die Saturn V, die in den 60-er Jahren die ersten Menschen zum Mond brachte. Schon heute stecken bis zu 40 Prozent des Wertes eines Wagens in Software und Elektronik. Tendenz: steigend.

Mit der höheren Softwarekomplexität wächst gleichzeitig die Zahl der Programmierfehler.
Bei einer „guten“ Software geht man von etwa 2 Bugs pro 1000 Programmzeilen aus. Die Software des Space Shuttles der NASA wurde besonders gründlich geprüft. Man schätzt das die drei Millionen Codezeilen trotzdem noch etwa 300 Fehler enthalten. Einer davon kann genügen, um eine Katastrophe auszulösen.
Laut Intel hat jeder Pentium-Prozessor etwas 80 bis 90 Bugs. Bei einer Handysoftware mit durchschnittlich 200.000 Zeilen Programmcode kann man mit etwa 600 Fehlern rechnen. Programme wie Windows XP haben etwa 40 Millionen Codezeilen. Auf DIN A 4-Seiten ausgedruckt ergibt das einen rund 60 Meter hohen Stapel Papier, in dem sich nach Statistik etwa 800.000 Bugs „eingeschlichen“ haben.

150 Millarden Euro Schäden jährlich in Europa

Professor Les Hatton von der Kingston University London schätzt die Kosten, die in Europa durch Softwarefehler entstehen auf jährlich bis zu 150 Milliarden Euro.
Durch richtiges Testen und frühes Finden von Fehlern könnte der Schaden deutlich verringert werden. Während Bugs während der Programmierung der Software noch relativ leicht und billig behoben werden können, steigen die Kosten einer Korrektur je später diese im Entwicklungsprozess vorgenommen wird. Im fertigen Produkt kostet der Fehler mindestens das Dreißigfache.

In der so genannten sicherheitskritischen Softwareentwicklung in den Bereichen Luft- und Raumfahrt, Automotive und Medizintechnik versucht man die Qualität durch Zertifizierungen abzusichern. Internationale Normen schreiben bestimmte Prozesse bei der Entwicklung und dem Test von Software vor.

Bananensoftware als Problem

In weniger kritischen Bereichen herrscht leider immer noch das Prinzip der „Bananensoftware“ vor. Die Software wird trotz unzureichender Tests sozusagen „grün“ auf dem Markt gebracht und reift dann beim Kunden aus. Die vermeintliche Kosteneinsparung nach dem Motto „der Kunde sagt uns schon wenn was nicht geht“ kann allerdings sehr teuer werden. Viele solcher „Wald-und-Wiesen“-Entwicklungsfirmen sind bereits vom Markt verschwunden. Zahlreiche Firmen mit schlechter Software werden wohl noch folgen.

Leistungsfähige Tools vereinfachen den Softwaretest

Testen ist ein wichtiger Bestandteil der Softwareentwicklung. Die Ausgaben für Test und Korrektur von Software belaufen sich auf etwa 40 bis 60 Prozent der gesamten Projektausgaben.
Mit spezieller Testsoftware wird das Prüfen von Computerprogrammen vereinfacht und effizient. Testtools decken Fehler früh im Softwareentwicklungsprozess auf und helfen Probleme im fertigen Produkt weitestgehend zu vermieden. Ergebnisse sind höhere Qualität und geringere Gesamtkosten.
Softwaretests sind also kein Kostenfaktor, sondern mittel- und langfristig ein Beitrag zur Kostensenkung.

Tools zur Analyse der **Softwarekomplexität** (Code Complexity Measures Tools) analysieren die Software und zeigen Umfang und Programmierstil der einzelnen Softwarebestandteile an. Teile, die zu komplex (also zu kompliziert) programmiert sind, haben nicht nur potentiell mehr Fehler, sondern sind auch schwieriger zu testen. Sollen zu komplexe Softwareteile für ein neues Produkt wiederverwendet werden, können Probleme auftreten. Hier gibt der Wartbarkeitsindex (Maintainability Index) an, wann es sinnvoller ist, den betreffenden Teil der Software neu zu schreiben. Komplexitätsmesstools geben ebenfalls die Mindestanzahl der Tests an, die für eine vollständige Prüfung der Software erforderlich ist.
Werden Drittfirmen mit der Entwicklung der Software beauftragt, helfen die durch die Code-Complexity-Measures-Tools angegebenen Metriken bei der Beurteilung des Arbeitsaufwands.

Mit **Unit-Test-Tools** können einzelne Teile der Software bereits früh im Entwicklungsprozess getestet werden, ohne dass hierfür das gesamte Projekt vorliegen muß. Fehlende Teile werden durch so genannte Stubs simuliert. Wenn Fehler gefunden werden, können diese logischerweise nur im getesteten Codeteil sein. Hierdurch wird die Lokalisierung der Fehlerquelle vereinfacht. Die einzelnen Funktionen der Software sind nach dem Unit-Test schon „sauber“ bevor Sie in die Software eingebaut werden.
Immer noch wird in vielen Softwareprojekten noch nur zum Schluß getestet. Das ist vergleichbar mit einem Autobauer, der alle Einzelteile wie Anlasser etc. ohne vorherigen Test in den Motor einbaut und erst bei der Probefahrt merkt, dass das Fahrzeug nicht funktioniert. Bei der Herstellung von Autos unvorstellbar - im Softwarebereich leider nicht ungewöhnlich.

Test Coverage (auch Code Coverage oder Testabdeckung genannt) zeigt die Teile der Software, die bereits getestet worden sind und die Softwarebereiche, für welche noch Tests geschrieben werden müssen. Es handelt sich also nicht um einen Test im eigentlichen Sinne, sondern um eine Analyse der Testqualität. Bei sicherheitskiritscher Software müssen bis zu 100% aller „Verhaltens-Möglichkeiten“ analysiert und die Tests entsprechend dokumentiert werden.

Werkzeuge für die **statische Codeanalyse** prüfen die Software ohne dass diese wie bei der dynamischen Codeanalyse ausgeführt werden muss. Es ist daher nicht erforderlich, dass die gesamte Software vorliegt. Mit der statischen Analyse können Defekte früh im Entwicklungsprozess gefunden werden. Die Analysetools „scannen“ den Programmcode. Das Schreiben von Tests ist hierfür nicht erforderlich.
Nach der automatischen Analyse zeigt das statische Testtool die Defekte im Code an.
Ein Teil der statischen Analysewerkzeuge beschränkt sich auf die Überprüfung von Programmierrichtlinien. Durch die Beachtung dieser Coding-Standards soll Softwarefehlern während der Programmierung vorgebeugt werden.

Beim **funktionalen Test** wird aus Sicht des Nutzers geprüft, ob sich die Software entsprechend den Anforderungen (Requirementts) verhält. Hierbei wird nicht der Programmiercode der Software analysiert, sondern ihr „externes“ Verhalten aus Sicht des Nutzers. Man spricht daher auch von **Black-Box-Tests**. Seit einigen Jahren sind für diese Testart automatische Testfallgeneratoren erhältlich. Die Nutzer-Anforderungen an die Software werden vor dem Test in einem grafischen Modell beschrieben. Der Testgenerator analysiert dieses Modell und generiert für alle möglichen Kombinationen automatisch die notwendigen Tests.

Klaus Lambertz ist geschäftsführender Gesellschafter und Mitgründer des im Offenburger Technologiepark ansässigen Softwaretestspezialisten Verifysoft Technology GmbH. Seit Aufbau der Firma im Jahr 2003 hatte er Einblick in sicherheitskritische Softwareprojekte bei über 200 Kunden. Bereits vor Gründung der Firma Verifysoft Technology konnte er umfangreiche Erfahrungen im Bereich des Softwaretests bei Testtoolherstellern in Deutschland, Frankreich und den USA sammeln. *Kontakt: lambertz@verifysoft.com*

Verifysoft Technology GmbH
Technologiepark
In der Spöck 10
77656 Offenburg (Germany)
Tel. +49 781 63 92 027

www.verifysoft.com

Stand Juni 2009