Der frühe Vogel fängt den Bug

IDC-Umfrage: Qualitätsprobleme kosten Softwarefirmen viel Geld

Die Ergebnisse, der von der Softwareschmiede Coverity mitfinanzierten IDC-Studie sprechen eine deutliche Sprache: Fast drei Viertel haben „ernste Probleme“ mit Softwaredefekten – auch nach dem Testen des Codes. In Zahlen ausgedrückt: 14 Mio. Euro pro Jahr müssen Softwarefirmen berappen, um die Fehler auszumerzen – wenigstens teilweise unnötig, wie die Experten feststellen.

Bild: Fotolia, anbk

Die Maßnahmen zur Sicherung der Softwarequalität sind angesichts der durch Softwaredefekte auftretenden internen und externen Kosten in den meisten Firmen unzureichend, lautet das Fazit der jüngsten Studie des Marktforschungsunternehmens International Data Corporation (IDC). Die mit „Improving Software Quality to Drive Business Agility“ betitelte und bei 139 amerikanischen und kanadischen Firmen durchgeführte Umfrage belegt, dass die Software für sorgenvolle Falten in den Gesichtern der Entwickler und Qualitätsbeauftragen in Unternehmen sorgt: Denn auch nach Durchführung der häufig recht umfangreichen Qualitätssicherungsmaßnahmen sowie der zeit- und kostenträchtigen Beseitigung von gefundenen Defekten bleibt die Softwareentwicklung ein latentes Problem. Sieben von zehn Unternehmen bestätigen, dass ihre Codebasis heute komplexer ist als noch vor zwei Jahren, und 72 Prozent bezeichnen ihr Debugging als „problematisch“. 40 Prozent der Anwendungen werden mit einem bis zu zehn Fehlern freigegeben. Je nach Zeitpunkt der Fehlerbehebung summiert sich der Aufwand in astronomische Höhen: Den Teilnehmern der von Coverity gesponserten IDC-Umfrage (Dokumentnummer 212971, Juni 2008) zufolge kostet die Fehlerbeseitigung die Unternehmen je nach Größe bis zu ca. 14 Mio. Euro oder 22 Mio. Dollar pro Jahr.

Die Studie präsentiert die Ergebnisse einer im zweiten Quartal 2008 unter Firmen in den USA und Kanada mit 250 bis 10.000 Angestellten durchgeführten Umfrage. Etwa 57 Prozent dieser Firmen beschäftigen 1.000 und mehr Mitarbeiter. Zielgruppe der Befragung waren dabei Beschäftigte aus den Bereichen des IT-Managements, der IT-Abteilungen und der Softwareentwicklung. 62 Prozent der Befragten zählen sich überdies als Direktoren und Manager zur oberen Führungsriege. Das heißt, dass sie ausreichend mit Qualitätsbelangen befasst sind, so dass sie detaillierte Auskunft über Test, Defekte und Code-Analyse geben können. Überdies will IDC herausgefunden haben, dass 53 Prozent der Unternehmen ihre eigene Software entwickeln, während 47 Prozent auf externe Ressourcen, Open Source und Dienstleister zugreifen.

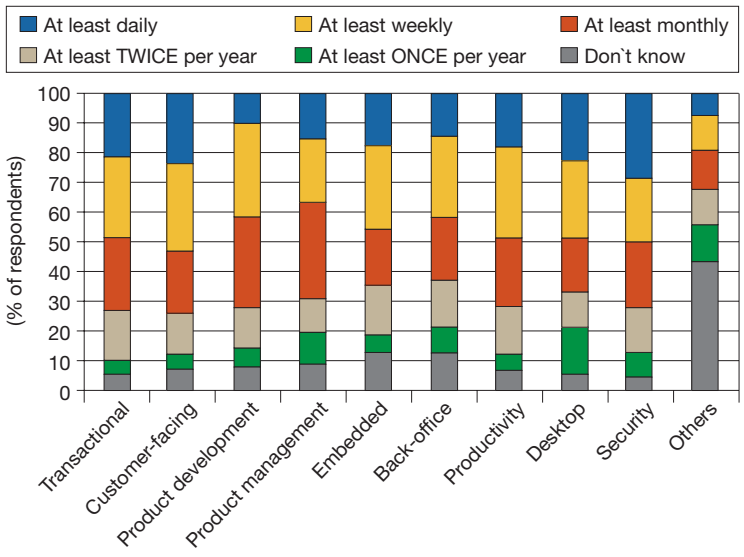
Verlauf der Studie

Demnach wird der Quellcode von Software gegenwärtig immer komplexer, insbesondere durch die Entwicklung von Multicore-Anwendungen, das zumindest berichten 71 Prozent der Unternehmen. Laut 63 Prozent der Befragten soll sich dieser Trend im kommenden Jahr weiter verschärfen. 72 Prozent der Befragten sind der Ansicht, dass das Debugging angesichts der Folgen dieser Entwicklung auch weiterhin problematisch bleibt.

Dennoch zeigen sich 62 Prozent optimistisch hinsichtlich ihrer internen Fehlerbehebungsprozesse und Testansätze („keine Verbesserung erforderlich“ bzw. „trotz Problemen keine Veränderung

Frequency of Application Updates Rolled into Production

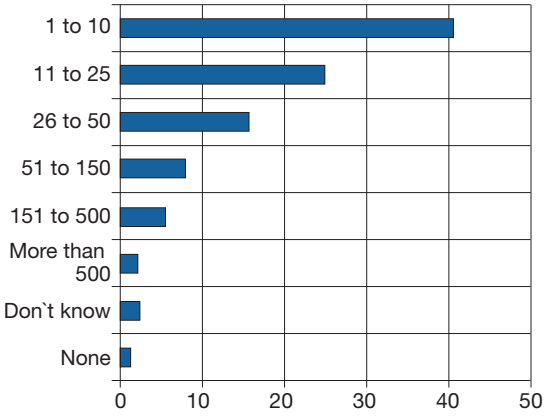
How often are the following applications updated and rolled into production?



Das Umfrageergebnis zum Thema „Anzahl der Applikations-Updates in der Produktion“ zeigt anschaulich, dass der Fehlerteufel schwer aufzuspüren ist.

Number of Critical Bugs Found Within 12 Months of Release

On average, how many critical bugs requiring patches are discovered in the 12-month period following release of the software into production?



Fehler in der Softwareentwicklung sind scheinbar nicht einfach zu finden, wie die Auswertung der Frage zur „Anzahl der gefundenen kritischen Bugs innerhalb von 12 Monaten“ zeigt

im Ansatz möglich“). Diese optimistische Einstellung hinsichtlich Anzahl, Konsequenzen und Ansatz der internen Prozesse zum Identifizieren und Reparieren von Codedefekten erklärt Melinda Ballou, Leiterin des Forschungsprogramms zum Application Life-Cycle Management bei IDC, mit mangelnder Aufklärung: „Während sich die Unternehmen die steigenden Komplexitätslevels und Arbeitsaufwand zum Beheben von Softwarefehlern eingestehen, unterschätzen sie gegenwärtige und laufende Aufwendungen für Reparatur und die Auswirkungen auf das Geschäft und Image.“ Auch in Zukunft entscheidet Software über den wirtschaftlichen Erfolg eines Unternehmens, ist sie überzeugt und mahnt: „Firmen sollten ihre derzeitigen Prozesse und Strukturen zur Sicherung der Softwarequalität auf den Prüfstand stellen sowie das Analysieren und Testen von Code automatisieren, um Software erfolgreicher und besser gesteuert zu implementieren.“

Problembewältigung: einfach und schnell?

Die IDC-Umfrage sieht die Probleme mit der Softwarequalität in mehreren Faktoren begründet: wachsende Komplexität von Code, auf unterschiedliche Standorte aufgeteilte Teams, Outsourcing, veralteter Code, Rückgriff auf Open-Source-Code und das vermehrte Aufkommen von Multi-Thread-Anwendungen. Untermauert wird das Ergebnis durch die Tatsache, dass über 50 Prozent der befragen Unternehmen bereits in den ersten 12 Monaten nach Freigabe der Software bis zu zehn kritische Defekte finden, für die Patches erforderlich sind. Laut Studie beträgt die Zeit zum Beheben der so genannten „field defects“ lediglich bei 16 Prozent einen Tag oder weniger, bei 66 Prozent sind es zwei bis zehn Tage und bei 11 Prozent sogar von 11 bis 30 Tage. Insgesamt schätzten die Befragten, dass die Beseitigung aller Defekte bereits vor dem Einsatz der Software eine Kosteneinsparung von 32 Prozent erzielen würde.

Um Fehler zu finden, verwenden 69 Prozent manuelle Code-Reviews, während 60 Prozent auf statische Analysen zurückgreifen und weitere 57 Prozent sich dynamische Analysetools zunutze machen (bei dieser Frage waren mehrere Auswahlmöglichkeiten erlaubt). Dass IDC jedem Entwickler rät, sich mit effizienten Tools bei der Fehlersuche zu befassen, ist nahe liegend, um frühzeitig Fehler zu erkennen und zu beheben.

Einige Befragten äußerten, dass die Debugging-Kosten erheblich seien. Um dies zu veranschaulichen, greift IDC auf ein Rechenbeispiel zurück: Nimmt man den durchschnittlichen Stundensatz eines Entwicklers in Höhe von 68 Dollar und multipliziert ihn mit einem angenommen Zeitaufwand von 30 Stunden, um einen Softwarefehler zu finden und zu beheben, summieren sich die Kosten schnell auf 2.040 Dollar. Zudem räumte ein Großteil der auskunftsfreudigen Befragten ein, dass sie jährlich 11,25 und mehr Bugs finden, deren Behebung gleich mehrere Entwickler beschäftigt, wodurch die Kosten weiter in die Höhe schnellen.

Überlebenswichtig: Probleme früh erkennen

Was dabei ziemlich außer Acht gelassen wird, betont Melinda Ballou von IDC in eindringlicher Weise: „Solcherlei Kosten können der Marke und der Reputation eines Unternehmens schaden.“ Sie lenkt die Aufmerksamkeit auf Produkt-Rückrufaktionen und beträchtliche Umsatzeinbußen besonders bei kritischen Anwendungen.

In dem Maße, in dem die wirtschaftliche Lage schwieriger wird und Unternehmen Personal abbauen, werden die Kosten für Ausfälle und kritische Defekte völlig untragbar. Mit dünnerer Personaldecke und allgemein geringeren Ressourcen lassen sich Fehler immer weniger tolerieren. Die frühe Erkennung von Problemen wird überlebenswichtig, denn: Werden Fehler erst in der Implementierungsphase behoben, kann das bis zu hundertmal teurer kommen, als wenn sie bereits am Anfang erkannt worden wären. „Gestützt auf unsere Zusammenarbeit mit IDC verfügt die ➜

Auf einen Blick

Automatisierung ist Pflicht!

Die IDC-Studie die Coverity gesponsert hat, dürfte für die Softwareschmiede eine gute Grundlage für diverse Weiterentwicklungen sein. Ungeachtet dessen, will IDC darauf aufmerksam machen, dass Softwarefehler einen langen Rattenschwanz an Kosten und Zeit für die Fehlerbehebung mit sich bringen. Vermeiden ließe sich dies bis zu einem gewissen Grad mit neuartigen Tools, welche automatisiert die Fehlersuche angehen. Ein Allheilmittel konnte IDC allerdings nicht benennen.

Branche jetzt über harte Fakten, die belegen, was nahezu alle Softwareunternehmen schon seit Jahren vermuteten – Qualitätsprobleme binden beträchtliche Ressourcen und bedrohen nach wie vor die Funktionsfähigkeit von Software im praktischen Einsatz", erläutert Ben Chelf, CTO von Coverity. „Firmen benötigen heute innovative Tools, die den langwierigen Prozess der Suche von Defekten automatisieren, damit sich die hochqualifizierten Entwickler auf die Weiterentwicklung des Produkts konzentrieren können, statt Fehler in alten Versionen beseitigen zu müssen." (*rob*) ■

Anmerkung: Die von Coverity mit unterstützte IDC-Studie „Improving Software Quality to Drive Business Agility", Dokumentnummer 212971, Juni 2008, steht zum Download bereit unter: www.coverity.com/library/pdf/IDC_Improving_Software_Quality_June_2008.pdf

infoDIREKT www.elektronikjournal.de 322ejl1108
Link zu IDC und Coverity

VORTEIL Aufschlussreiche und lesenswerte Studie bei der man allerdings den Sponsor nicht ganz aus den Augen verlieren sollte.

Kurzinterview mit Melinda Ballou, Leiterin des Forschungsprogramms zum Application Life-Cycle Management bei IDC und Ben Chelf, CTO von Coverity

Die komplexe Welt der Software

Laut der IDC-Umfrage rühren die Probleme mit der Softwarequalität von folgenden Faktoren her: wachsende Komplexität von Code, auf verschiedene Standorte aufgeteilte Teammitglieder, Outsourcing, veralteter Code, Rückgriff auf Open-Source-Code und das vermehrte Aufkommen von Multi-Thread-Anwendungen.

Melinda Ballou vom IDC aus Framingham, Massachusetts (USA)

Ben Chelf von Coverity aus San Francisco (USA)

Die Komplexität des Codes steigt seit Charles Babbage an, wenigstens jedoch seit Konrad Zuse – hat für dieses uralte Problem noch niemand eine Antwort entdeckt? Gibt es hier neue Entwicklungen, die das Problem verschärfen, oder hat die Industrie geschlafen?
Melinda Ballou: Die Antwort lautet: Nein! Ich bin nicht sicher, ob man überhaupt eine Antwort auf das Problem Softwarekomplexität finden kann. Die Arten, wie die Komplexität von Code heute – also von 2008 bis 2010 – unterscheiden sich durch folgende Faktoren von früheren Herausforderungen:
1. Probleme aus komplexem Sourcing, einschließlich Off-shoring, Outsourcing, verteiltes internes Mitarbeiterteam und dem Einsatz von Open-Source
2. Aus der Befolgung von und Anpassung an internationale Besonderheiten resultierende Herausforderungen.
3. Problemstellungen, die sowohl aus neuen Entwicklungen wie SOA, Web-2.0, etc. als auch aus Belangen der Sicherheit entstehen.
Ben Chelf: Außerdem ist Software heute sehr viel weiter verbreitet als noch vor 10 Jahren. Als Treiber sind sicherlich Mobile Computing, Internet, e-Kioske an Flughäfen und vieles mehr zu nennen. Folglich antworten die Anbieter mit Outsourcing und Internationalisierung und nutzen bestehende Frameworks. Fast jedes Unternehmen muss sich heute mit Softwareentwicklung beschäftigen oder ist davon betroffen.

Die geographisch verteilte Entwicklung ist ebenfalls nichts Neues. Für Outsourcing gilt dasselbe. Wieso stellen diese Dinge, die der Industrie häufig zum Vorteil gereichen, heute so große Probleme dar?
Melinda Ballou: Das Aufkommen einer globalen Wirtschaft bringt einen noch nie dagewesenen Übernahmegrad von komplexem Software-Sourcing mit sich. Die Koordination der Codeentwicklung über unterschiedliche Gruppen im selben Unternehmen, zum Beispiel Stakeholder, IT oder Operations, etc. ist für Unternehmen Herausforderung genug. Fügt man noch externe Outsourcer und Offshore-Anbieter hinzu, kommt man zu bisher unerreichten Komplexitätsleveln.
Ben Chelf: Bedenken Sie auch die Menge an Code, die geschrieben wird. In den 1970er Jahren waren zehn- oder hunderttausende Codezeilen die Obergrenze und heute sind Millionen an Codezeilen keine Seltenheit. Alleine in einem Auto existieren heute 10 Mio. Codezeilen. Die Koordination der Entwicklung ist ganz einfach ein anderes Problem als für eine Entwicklung kleineren Ausmaßes.

Was heißt „veralteter Code"? Wenn ein Codeabschnitt eine bestimmte Aufgabe erfüllt, wie kann er da „veralten"? Oder ist er in Basic statt C++ geschrieben?
Melinda Ballou: Veralteter Code bedeutet Code, der keine relevante Aufgabe hat. Dann wird er nutzlos oder eben veraltet.
Ben Chelf: Nehmen wir als Beispiel ein Telefongerät und dessen Software zur Steuerung. Von einer Version zur nächsten verändern sich die Tasten auf dem Telefon: einige neue Tasten kommen dazu, andere fallen weg. Oft zögern Entwickler ‚Code zu berühren', aus Angst sie könnten etwas zerstören. Deshalb bleibt der Code der in der nächsten Telefongeneration entfernten Tasten wahrscheinlich in der Software enthalten. Dieser Code ist definitiv veraltet.

Niemand zwingt Unternehmen dazu, auf Open-Source-Code zurückzugreifen. Wenn sie damit tatsächlich nicht zufrieden sind: Warum tun sie es?
Melinda Ballou: Open-Source bietet den Unternehmen eine Möglichkeit zur Zusammenarbeit, um Code zu entwickeln und ihr Intellectual Property gemeinsam zu benutzen, so dass die Unternehmen im Bestfall von einer breiten Entwicklungsgemeinschaft und von noch zahlreicheren und – wenn man so will – ‚kostenfreien' Softwarelösungen profitieren.
Ben Chelf: Ganz genau – oft ist das einfacher, auch wenn es nicht die perfekte Lösung darstellt.

Dass die Studie von Coverity gefördert wurde, wird in weiten Teilen sehr deutlich. Wieso werden ganze Bereiche der Qualitätssicherung mit keinem Wort erwähnt, etwa CMMI, also Capability-Maturity-Model-Integration, und Rational-Unified-Process mit UML? Agile Methoden sind gerade mal in einem Halbsatz erwähnt. Bezieht sich die Studie ausschließlich auf Coveritys Methoden?
Melinda Ballou: Diese Studie fokussiert weder Arbeitsabläufe noch Dauer. Wenn sie das täte, wären die Themen wie CMMI, agile, Scrum, Eclipse Process Framework, RUP und deren Einsatz erwähnt. Hauptsächlich lag der Fokus auf Defekten, auf der Sichtweise von Unternehmen und ihren eigenen Ansätzen, und auf maßgeblichen Kostenbeispielen, um Defekte in ihrem Code zu beheben. Wahrscheinlich ist einer der interessantesten Punkte die unterschiedliche Sichtweise, wie Unternehmen ihre eigenen Qualitätsinitiativen betrachten.

Die Fragen stellte Marisa Robles Consée