Testbegriffe und Glossar

1. Software Qualität
2. Unit-Test
3. Stress-Test
4. Explorativer Test
5. Branch Coverage
6. Condition Coverage

Software Qualität

Bedeutung:
Qualität bezieht sich auf Produkte und Prozesse. Somit wird Produktqualität und Prozessqualität unterschieden. Softwarequalität ist die Gesamtheit von Merkmalen und Merkmalswerten (Kenngrößen), die die festgelegten Anforderungen eines Software-Produktes erfüllen. Sie beziehen sich auf die auf dessen Eignung, festgelegte oder vorausgesetzte Erfordernisse zu erfüllen (nach ISO/IEC 9126). Qualitätsmerkmale von Software sind nach ISO/IEC 9126 Funktionalität, Zuverlässigkeit, Benutzbarkeit, Effizienz, Wartbarkeit/Wartungsfreundlichkeit und Portabilität/Übertragbarkeit. Diese Merkmale bestimmen den Grad der Eignung des Software-Produktes für einen bestimmten Verwendungszweck. Korrelation von Merkmalen: Es gibt einen Widerspruch zwischen hohen Anforderungen bezüglich der Effizienz und einer guten Wartbarkeit bzw. Übertragbarkeit. Funktionale Korrektheit verbessert automatisch die Zuverlässigkeit und Wartbarkeit. Eine gute Bedienbarkeit beeinflusst kein weiteres Qualitätsmerkmal negativ. Eine allgemeine Definition ist für die praktische Anwendung nicht ausreichend. Durch ein Qualitätsmodell werden Merkmale auf der obersten Abstraktionsebene durch Teilmerkmale verfeinert. Die Teilmerkmale werden durch Qualitätsindizes bzw. -maße mess- und bewertbar gemacht.

Abgrenzung: Qualitätsmanagement, Qualitätssicherung

Unklarheiten: -

Querverweis: -

Quellen:

- Folien Uni Marburg (https://www.uni-marburg.de/fb12/swt/lehre/files/est1415/EST150120.pdf) , Stand: 03.04.2016. 20:34
- Enzyklopädie der Wirtschaftsinformatik (http://www.enzyklopaedie-der-wirtschaftsinformatik.de/lexikon/is-management/Systementwicklung/Management-der-Systementwicklung/Software-Qualitatsmanagement/Qualitatsmerkmale-von-Software/index.html) , Stand: 03.04.2016 20:44
- Softwarequalitätsmodelle, Clara Lange, Technische Universität München Fakultät für Informatik, Boltzmannstraße 3 85748 Garching-Forschungszentrum langecl@in.tum.de
- Folien TU Ilmenau (https://www.tu-

ilmenau.de/fileadmin/media/sspi/Lehre/Vorlesungen/SQS/VL_Scripte/SQS_1praes.pdf) , Stand: 03.04.2016 20:49

Unit-Test

Synonym: Modultest

Bedeutung:
Das Ziel von Unit-Tests ist das Finden von Fehlern, nicht das Nachweisen von Korrektheit. Die Anwendung wird in die kleinsten testbaren Software-Einheiten aufgeteilt und unabhängig von der restlichen Software auf ihre Funktionalität getestet. Jede Einheit wird separat getestet, bevor die Module integriert werden und ihre Schnittstellen getestet werden. Die Abhängigkeit zu anderen Komponenten wird nicht geprüft. Ein Unit-Test testet einen einzelnen Aspekt einer Komponente. Ein guter Unit-Test besteht nur aus einem Aufruf. Es ist ein Test, welcher sich auf Komponenten eines Systems bezieht. Diese Komponenten können Methode, Klasse, Subsystem oder das Gesamtsystem sein. Meistens: Klasse (CUT) bzw. Objekt (OUT) oder Software (SUT). Auf den Testling werden definierte Eingabewerte angewendet, die Ergebnisse werden mit den Erwartungswerten verglichen. Eigenschaften: wiederholt und automatisiert ausführbar, häufiges Wiederholen möglich, der Test läuft schnell durch, ist unabhängig von anderen Tests, kann immer wieder unter denselben Bedingungen ausgeführt werden, einfach zu implementieren, kompliziertes Setup weißt auf Integrationstests hin, dienen der Beschreibung von Funktionalität, demonstrieren Benutzung und erwartetes Verhalten, sollte nicht öfter geändert werden als der getestete Code, auch für andere Entwickler zugänglich. Die Automatisierung des Testprozesses wird ermöglicht. Schwierigkeiten beim Entdecken von Fehlern in komplexeren Teilen der Anwendung werden reduziert. Die Testabdeckung wird häufig verbessert, da der Fokus beim Testen auf jeder einzelnen Einheit liegt.

Abgrenzung: Integrationstest

Unklarheiten: -

Querverweis: -

Quellen:

- Folien HU Berlin (https://www2.informatik.hu-berlin.de/swt/lehre/MTI/seminars/PR_MTI_1112/resources/restricted/speeches/Unit%20Testing,%20SUnit%20and%20You.pdf) , Stand: 03.04.2016. 20:34
- Microsoft (https://msdn.microsoft.com/en-us/library/aa292197%28v=vs.71%29.aspx) , Stand: 03.04.2016 20:44
- Informatik aktuell (http://www.informatik-aktuell.de/entwicklung/methoden/gute-unit-tests-und-testgetriebene-entwicklung-tdd.html) , Stand: 03.04.2016 20:49

Stress-Test

Bedeutung:
Stresstests sind Tests, die das Verhalten eines Systems unter Ausnahmesituationen prüfen und analysieren. Stresstests können Anwendungsprobleme erkennen, die nur unter extremen Bedingungen auftreten. Crashtests sind Stresstests, die versuchen, das System zum Absturz zu bringen. Bei einem Stresstest wird die simulierte Last über das im Normalbetrieb erwartete Maß erhöht, bis funktionale Fehler auftreten oder das Antwortverhalten des getesteten Systems bestimmte Grenzen überschreitet. Hierfür wird oft eine kontinuierlich ansteigende Nutzerzahl simuliert. Stresstests verwendet man beispielsweise zur Ermittlung des Systemverhaltens in und

nach Überlastsituationen, zur Ermittlung der maximal akzeptierbaren Last oder zum Finden von Performance-Schwachstellen (Bottlenecks). Ein Stresstest ist eine Art von Performance – Test, der sich auf Robustheit, Verfügbarkeit und Zuverlässigkeit unter extremen Bedingungen für eine Anwendung konzentriert. Das Ziel der Stresstests ist es Probleme zu erkennen, die nur unter extremen Bedingungen entstehen oder erkennbar werden. Diese Bedingungen können zum Beispiel schwere Lasten, hohe Parallelität oder begrenzte Rechenressourcen sein. Stresstests sind bei der Suche nach Problemen bei Synchronisation und Timing Bugs, Interlock Problemen und Ressourcenverlust Bugs nützlich. Stresstests beinhalten in der Regel die Simulation eines oder mehrerer Schlüssel Produktionsszenarien unter einer Vielzahl von Stressbedingungen.
Abgrenzung: Lasttest, Dauerlasttest, Performancetest

Unklarheiten: -

Querverweis: -

Quellen:

- Xceptance (https://blog.xceptance.com/2009/09/16/begriffe-erklart-lasttest-stresstest/) , Stand: 03.04.2016 20:58
- Wikipedia (https://de.wikipedia.org/wiki/Softwaretest) , Stand: 03.04.2016 21.02
- Microsoft (https://msdn.microsoft.com/en-us/library/bb924374.aspx) , Stand: 03.04.2016 21:05

Explorativer Test

Bedeutung:
Exploratives Testen ist ein intuitives Vorgehen, welches als Ergänzung zum systematischen Testen dient. Hierbei dient die intuitive Fähigkeit und Erfahrung des Testers, Testfälle nach erwarteten Fehlerzuständen und -wirkungen auszuwählen. Beispielsweise basieren Testfälle auf den Erfahrungen, wo Fehler in der Vergangenheit aufgetreten sind, oder die Vermutung des Testers wo in Zukunft welche auftreten können. Diese Vermutungen können z.B. auf die Erfahrung des Testers basieren, die er bereits in ähnlichen Projekten gemacht hat. Bei schlecht oder sogar gar nicht dokumentierten Projekten kann das explorative Testen weiterhelfen. Bei diesem Testverfahren, werden die einzelnen Aufgaben und Funktionen der Anwendung „erforscht“. Anschließend wird entschieden welche Teile getestet werden sollen. Auf diese Weise wird das unbekannte Verhalten des Testobjekts weiter geklärt, dabei dienen Auffälligkeiten und weitere Informationen zur Erstellung der nächsten Testfälle. Ein sinnvolles Ergebnis des explorativen Testens kann sein, welche Testverfahren sinnvollerweise für die Anwendung einzusetzen sind.

Quellen:

- Buch: Andreas Spillner, Tilo Lenz: „Basiswissen Softwaretest“
- SWTest-Blog (http://swtest-blog.de/)
- Microsoft (http://msdn.microsoft.com)

Branch Coverage

Bedeutung:
Beim Branch Coverage auch Zweigüberdeckungstest genannt, werden Testfälle hergeleitet die sicherstellen, dass jeder Zweig eines Programms mindestens einmal durchlaufen wird. Mithilfe des Zweigüberdeckungstests lassen sich nicht ausführbare Programmzweige aufspüren.

Quellen:

- Andreas Spillner, Tilo Lenz: „Basiswissen Softwaretest“
- Wikipedia: Kontrollflussorientierte Testverfahren (http://de.wikipedia.org)
- Wiki FH WS: Kontrollflussorientierter Test (http://www.iwiki.de)

Condition Coverage

Bedeutung:
Beim Branch Coverage auch Bedingungsüberdeckungstest genannt, muss jede Teilbedingung (Bedingungen ohne logischen Operator) eines Programms mit „true“ und mit „false“ getestet werden. Dabei wird in Einfachbedingungsüberdeckungstest und in Mehrfachbedingungsüberdeckungstest. Ziel des Einfachbedingungsüberdeckungstest ist es das jede Teilbedingung einmal jeden der Wahrheitswerte annimmt, so gäbe es beifolgendem Beispiel nur zwei Tests: x>3 OR y<5, einmal der Test bei dem die Wahrheitswerte links „true“ und rechts „false“ wären und umgekehrt. Beim Mehrfachbedingungsüberdeckungstest gibt es stets 2^n Testfälle, wobei n die Anzahl der Teilbedingen darstellt da dort jeweils alle Kombinationen von Wahrheitswerten berücksichtigt werden.

Quellen:

- Andreas Spillner, Tilo Lenz: „Basiswissen Softwaretest“
- Wikipedia: Bedingungsüberdeckungstest (http://de.wikipedia.org)
- Tutorialspoint (www.tutorialspoint.com)