Certified Tester

Advanced Level Syllabus Testmanager

VERSION 2012

International Software Testing Qualifications Board

Deutschsprachige Ausgabe. Herausgegeben durch
Austrian Testing Board, German Testing Board e.V. &
Swiss Testing Board

Arbeitsuntergruppe Advanced Level Testmanager: Rex Black (Vorsitzender), Judy McKay (stellvertretende Vorsitzende), Graham Bath, Debra Friedenberg, Bernard Homès, Kenji Onishi, Mike Smith, Geoff Thompson, Tsuyoshi Yumoto; 2010-2012.

Änderungsübersicht

Version	Datum	Bemerkungen
ISEB v1.1	4. September 2001	ISEB Practitioner Syllabus
ISTQB 1.2E	September 2003	ISTQB® Advanced Level Syllabus von EOQ-SG
V2007	12. Oktober 20007	Certified Tester Advanced Level Syllabus Version 2007 (Lehrplan Aufbaukurs Certified Tester)
D100626	26. Juni 2010	Einarbeitung von den in 2009 angenommenen Änderungen, Aufteilung der jeweiligen Kapitel für die einzelnen Module
D101227	27. Dezember 2010	Annahme von Formatänderungen und Korrekturen, die keinen Einfluss auf den Sinngehalt haben
D2011	31. Oktober 2011	Übergang zum geteilten Lehrplan, Überarbeitung von Lernzielen und Textänderungen zum Angleich an die Lernziele; Hinzufügen der geschäftlichen Ergebnisse
Alpha 2012	9. Februar 2012	Einarbeitung aller Reviewbemerkungen der nationalen Boards, die zur Version Oktober 2011 eingingen
Beta 2012	26. März 2012	Einarbeitung von Reviewbemerkungen der nationalen Boards zur Alpha-Version, die rechtzeitig eingingen
Beta 2012	7. April 2012	Beta-Version zur Vorlage bei der Hauptversammlung
Beta 2012	8. Juni 2012	Beta-Freigabe für nationale Boards nach technischer Überarbeitung
Beta 2012	27. Juni 2012	Einarbeitung von EWG- und Glossar-Bemerkungen
RC 2012	15. August 2012	Freigabeversion für GA – finale techn. Überarbeitung
GA 2012	19. Oktober 2012	Letzte techn. Überarbeitung und Bereinigung für GA Freigabe
DACH 2012 German	19. April 2013	Lokalisierung nach Übersetzung durch ATB, GTB und STB

Inhaltsverzeichnis

Änderungsübersicht ... 3
Inhaltsverzeichnis ... 4
Dank ... 6
0. Einführung in den Advanced Level Syllabus ... 7
0.1 Zweck dieses Dokuments ... 7
0.2 Übersicht ... 7
0.3 Prüfungsrelevante Lernziele ... 7
1. Testprozess – [420 Minuten] ... 8
1.1 Einführung ... 9
1.2 Testplanung, -überwachung und -steuerung ... 9
1.2.1 Testplanung ... 9
1.2.2 Testüberwachung und -steuerung ... 11
1.3 Testanalyse ... 12
1.4 Testentwurf ... 13
1.5 Testrealisierung ... 14
1.6 Testdurchführung ... 15
1.7 Bewertung von Endekriterien und Bericht ... 15
1.8 Abschluss der Testaktivitäten ... 16
2. Testmanagement – [750 Minuten] ... 18
2.1 Einführung ... 20
2.2 Testmanagement im Kontext ... 20
2.2.1 Die Interessensvertreter (Stakeholder) des Testens verstehen ... 20
2.2.2 Zusätzliche Softwarelebenszyklus-Aktivitäten und Arbeitsergebnisse ... 21
2.2.3 Anpassungen der Testaktivitäten an andere Lebenszyklusaktivitäten ... 22
2.2.4 Management von nicht-funktionalen Tests ... 24
2.2.5 Management des erfahrungsbasierten Tests ... 25
2.3 Risikoorientierter Test und andere Ansätze zur Testpriorisierung und Aufwandszuweisung ... 26
2.3.1 Risikoorientiertes Testen ... 26
2.3.2 Risikoorientierte Testverfahren ... 31
2.3.3 Weitere Verfahren zur Testauswahl ... 34
2.3.4 Testpriorisierung und Aufwandszuweisung im Testprozess ... 35
2.4 Testdokumentation und weitere Arbeitsergebnisse ... 36
2.4.1 Testrichtlinie ... 36
2.4.2 Teststrategie ... 37
2.4.3 Mastertestkonzept ... 39
2.4.4 Stufentestkonzept ... 40
2.4.5 Projektrisikomanagement ... 40
2.4.6 Weitere Arbeitsergebnisse des Testens ... 41
2.5 Testaufwandsschätzung ... 41
2.6 Definieren und Anwenden von Testmetriken ... 43
2.7 Mehrwert des Testens ... 48
2.8 Verteiltes Testen, Outsourcing und Insourcing ... 49
2.9 Die Anwendung von Industriestandards managen ... 50
3. Reviews – [180 Minuten] ... 52
3.1 Einführung ... 53
3.2 Managementreviews und Audits ... 54
3.3 Management von Reviews ... 55
3.4 Metriken für Reviews ... 57
3.5 Management von formalen Reviews ... 58
4. Fehlermanagement – [150 Minuten] ... 59

4.1 Einführung ... 60
4.2 Fehlerlebenszyklus und Softwarelebenszyklus ... 60
4.2.1 Fehler-Workflow und Status von Fehlerzuständen ... 61
4.2.2 Ungültige und doppelte Fehler managen ... 61
4.2.3 Übergreifendes Fehlermanagement ... 62
4.3 Informationen im Fehlerbericht bzw. Fehlermeldung ... 62
4.4 Bewerten der Prozessreife anhand der Fehlerberichtsinformationen ... 64
5. Verbesserung des Testprozesses – [135 Minuten] ... 66
5.1 Einführung ... 67
5.2 Testverbesserungsprozess ... 67
5.2.1 Einführung in die Prozessverbesserung ... 67
5.2.2 Arten der Prozessverbesserung ... 68
5.3 Testprozess verbessern ... 68
5.4 Testprozess mit TMMi verbessern ... 69
5.5 Testprozess mit TPI Next verbessern ... 70
5.6 Testprozess mit CTP verbessern ... 70
5.7 Testprozess mit STEP verbessern ... 71
6. Testwerkzeuge und Automatisierung – [135 Minuten] ... 72
6.1 Einführung ... 73
6.2 Auswahl von Werkzeugen ... 73
6.2.1 Open-Source-Werkzeuge ... 73
6.2.2 Maßgeschneiderte Werkzeuge ... 74
6.2.3 Rentabilität ... 74
6.2.4 Auswahlverfahren ... 76
6.3 Werkzeuglebenszyklus ... 77
6.4 Werkzeugmetriken ... 78
7. Soziale Kompetenz und Teamzusammensetzung – [210 Minuten] ... 80
7.1 Einführung ... 81
7.2 Individuelle Fähigkeiten ... 81
7.3 Dynamik im Testteam ... 83
7.4 Testen in der Organisationsstruktur etablieren ... 84
7.5 Motivieren ... 86
7.6 Kommunikation ... 87
8. Referenzen ... 89
8.1 Standards ... 89
8.2 ISTQB Documents ... 89
8.3 Eingetragene Marken ... 89
8.4 Literatur ... 90
8.5 Sonstige Referenzen ... 91
8.6 Ergänzende deutschsprachige Literatur ... 91
9. Index ... 92

Dank

Dieses Dokument wurde von einem Kernteam der Arbeitsuntergruppe „Advanced Level Syllabus – Advanced Testmanager" des International Software Testing Qualifications Board erstellt. Dieser Arbeitsgruppe gehörten an: Rex Black (Vorsitzender), Judy McKay (stellvertretende Vorsitzende), Graham Bath, Debra Friedenberg, Bernard Homès, Paul Jorgensen, Kenji Onishi, Mike Smith, Geoff Thompson, Erik van Veenendaal, Tsuyoshi Yumoto.

Das Kernteam bedankt sich beim Reviewteam und bei den nationalen Boards für die konstruktiven Vorschläge und Beiträge.

Bei Fertigstellung des Advanced Level Lehrplans hatte die Arbeitsgruppe „Advanced Level Syllabus" die folgenden Mitglieder (in alphabetischer Reihenfolge):

Graham Bath, Rex Black, Maria Clara Choucair, Debra Friedenberg, Bernard Homès (stellvertretener Vorsitzender), Paul Jorgensen, Judy McKay, Jamie Mitchell, Thomas Mueller, Klaus Olsen, Kenji Onishi, Meile Posthuma, Eric Riou du Cosquer, Jan Sabak, Hans Schaefer, Mike Smith (Vorsitzender), Geoff Thompson, Erik van Veenendaal, Tsuyoshi Yumoto

Folgende Personen haben an Review, Kommentierung und der Abstimmung über diesen Lehrplan mitgearbeitet (in alphabetischer Reihenfolge):

Chris van Bael, Graham Bath, Kimmo Hakala, Rob Hendriks, Marcel Kwakernaak, Rik Marselis, Don Mills, Gary Mogyorodi, Thomas Mueller, Ingvar Nordstrom, Katja Piroué, Miele Posthuma, Nathalie Rooseboom de Vries, Geoff Thompson, Jamil Wahbeh, Hans Weiberg.

Deutschsprachige Fassung:
Das Swiss Testing Board (STB), das Austrian Testing Board (ATB) und das German Testing Board (GTB) danken ebenso dem Review-Team der deutschsprachigen Fassung 2012 (in alphabetischer Reihenfolge): Andra Calancea (ATB), Andreas Gäng (Leitung, STB), Martin Klonk (ATB), Clemens Mucker (ATB), Helmut Pichler (ATB), Horst Pohlmann (GTB), Thomas Puchter (ATB), Angela Schiehl (ATB), Maud Schlich (GTB), Stephan Weißleder (GTB), Mario Winter (GTB).

Dieses Dokument wurde von der Hauptversammlung des ISTQB® am 19.10.2012 offiziell freigegeben.

Die vorliegende Fassung ist die freigegebene Fassung der DACH-Boards (GTB, ATB und STB) und wird auf der jeweiligen Webseite veröffentlicht..

0. Einführung in den Advanced Level Syllabus

0.1 Zweck dieses Dokuments

Dieser Lehrplan bildet die Grundlage für das Softwaretest-Qualifizierungsprogramm der Aufbaustufe (Advanced Level) für Testmanager. Das ISTQB® stellt den Lehrplan folgenden Adressaten zur Verfügung:

1. Nationalen/regionalen Boards zur Übersetzung in die jeweilige Landessprache und zur Akkreditierung von Ausbildungsanbietern. Die nationalen Boards können den Lehrplan an die eigenen sprachlichen Anforderungen anpassen sowie die Querverweise ändern und an die bei ihnen vorliegenden Veröffentlichungen angleichen.
2. Prüfungsinstitutionen zur Erarbeitung von Prüfungsfragen in der jeweiligen Landessprache, die sich an den Lernzielen der jeweiligen Lehrpläne orientieren.
3. Ausbildungsanbietern zur Erstellung ihrer Kursunterlagen und zur Bestimmung einer geeigneten Unterrichtsmethodik.
4. Prüfungskandidaten zur Vorbereitung auf die Prüfung (als Teil des Ausbildungslehrgangs oder auch kursunabhängig).
5. Allen Personen, die im Bereich Software- und Systementwicklung tätig sind und die professionelle Kompetenz beim Testen von Software verbessern möchten, sowie als Grundlage für Bücher und Fachartikel.

Das ISTQB® kann auch anderen Personenkreisen oder Institutionen die Nutzung dieses Lehrplans für andere Zwecke genehmigen, wenn diese vorab eine entsprechende schriftliche Genehmigung einholen.

0.2 Übersicht

Der Advanced Level besteht aus drei separaten Lehrplänen:

- Testmanager
- Test Analyst
- Technical Test Analyst

Im Übersichtsdokument zum Advanced Level [ISTQB_AL_OVIEW] sind folgende Informationen enthalten:

- Mehrwert für jeden Lehrplan
- Zusammenfassung der einzelnen Lehrpläne
- Beziehungen zwischen den Lehrplänen
- Beschreibung der kognitiven Stufen des Wissens
- Anhänge

0.3 Prüfungsrelevante Lernziele

Die Lernziele unterstützen die geschäftlichen Ziele und dienen zur Ausarbeitung der Prüfung für Testmanager Advanced Level (CTAL-TM). Allgemein gilt, dass der gesamte Inhalt des vorliegenden Advanced Level Lehrplans entsprechend der Lernziele der kognitiven Stufe K1 geprüft werden kann. Dies bedeutet, dass die Prüfungskandidaten Begriffe oder Konzepte erkennen, sich an sie erinnern und sie wiedergeben können. Die relevanten Lernziele der kognitiven Stufen K2 (Verstehen), K3 (Anwenden) und K4 (Analysieren) werden immer zu Beginn des jeweiligen Kapitels angegeben.

1. Testprozess – [420 Minuten]

Begriffe

Abschluss der Testaktivitäten, Endekriterien, Testablauf, Testabschlussbericht, Testbedingung, Testdurchführung, Testentwurf, Testfall, Testplanung, Testprotokoll, Testrealisierung, Testskript, Teststeuerung

Lernziele für den Testprozess

1.2 Testplanung, -überwachung und -steuerung

TM-1.2.1 (K4) Sie können den Testbedarf für ein System analysieren, um die zur Erreichung der Testziele erforderlichen Testaktivitäten und Arbeitsergebnisse zu planen

1.3 Testanalyse

TM-1.3.1 (K3) Sie können das Prinzip der Rückverfolgbarkeit anwenden, um die Vollständigkeit und Konsistenz definierter Testbedingungen hinsichtlich der Testziele, Teststrategie und Testkonzept zu prüfen

TM-1.3.2 (K2) Sie können die Faktoren erklären, die sich auf den Detaillierungsgrad der zu spezifizierenden Testbedingungen auswirken können und Sie können die Vor- und Nachteile einer detaillierten Spezifizierung von Testbedingungen erläutern.

1.4 Testentwurf

TM-1.4.1 (K3) Sie können das Prinzip der Rückverfolgbarkeit anwenden, um die Vollständigkeit und Konsistenz von entworfenen Testfällen hinsichtlich der definierten Testbedingungen zu prüfen

1.5 Testrealisierung

TM-1.5.1 (K3) Sie sind in der Lage einen Testausführungsplan zu erstellen, der eventuelle Risiken, nötige Prioritäten, vorhandene Testumgebungen und bestehende Datenabhängigkeiten mit in Betracht zieht. Ihr Testausführungsplan berücksichtigt dabei auch die gegebenen Testziele, die Teststrategie und das Testkonzept und widerspricht ihnen nicht

1.6 Testdurchführung

TM-1.6.1 (K3) Sie können das Prinzip der Rückverfolgbarkeit anwenden, um den Testfortschritt hinsichtlich der Vollständigkeit und Konsistenz mit Testzielen, Teststrategie und Testkonzept zu überwachen

1.7 Bewertung von Endekriterien und Bericht

TM-1.7.1 (K2) Sie können erklären, warum das Sammeln genauer und aktueller Informationen im Laufe des Testprozesses für eine korrekte Testberichterstattung und Bewertung der Endekriterien wichtig ist

1.8 Abschluss der Testaktivitäten

TM-1.8.1 (K2) Sie können die vier Gruppen von Testabschlussaktivitäten zusammenfassen.

TM-1.8.2 (K3) Sie können eine Projektretrospektive durchführen, um Prozesse zu bewerten und Bereiche für eine Verbesserung zu identifizieren

1.1 Einführung

Der ISTQB® Foundation Level Lehrplan beschreibt den fundamentalen Testprozess, der aus folgenden Hauptaktivitäten besteht:

- Testplanung und Teststeuerung
- Testanalyse und Testentwurf
- Testrealisierung und Testdurchführung
- Bewertung von Endekriterien und Bericht
- Abschluss der Testaktivitäten

Bereits im Foundation Level Lehrplan wird festgelegt, dass die Testprozessaktivitäten in der Praxis zeitlich überlappend oder parallel stattfinden, auch wenn diese im Lehrplan logisch sequentiell aufgelistet sind. Gewöhnlich ist es nötig, Ausprägungen und Reihenfolge dieser Hauptaktivitäten jeweils dem zu testenden System und dem Projekt anzupassen.

In den Advanced Level Lehrplänen werden einige dieser Aktivitäten getrennt behandelt. Hierdurch soll eine weitere Verfeinerung und Optimierung der Prozesse, eine bessere Anpassung an den Softwareentwicklungs-Lebenszyklus, sowie eine effektive Testüberwachung und -steuerung ermöglicht werden. Die Testprozessaktivitäten werden nun wie folgt behandelt:

- Testplanung, -überwachung und -steuerung
- Testanalyse
- Testentwurf
- Testrealisierung
- Testdurchführung
- Bewertung von Endekriterien und Bericht
- Abschluss der Testaktivitäten

1.2 Testplanung, -überwachung und -steuerung

Dieses Kapitel behandelt die Prozesse der Testplanung, -überwachung und -steuerung. Wie im Foundation Level erwähnt, handelt es sich bei diesen Aktivitäten um Aufgaben des Testmanagers.

1.2.1 Testplanung

Für jede Teststufe beginnt die Testplanung mit der Einleitung des Testprozesses für die jeweilige Teststufe und wird während des gesamten Projekts fortgesetzt, bis die Testabschlussaktivitäten für die Teststufe abgeschlossen sind. Dazu gehört die Identifizierung der Aktivitäten und Ressourcen, die zur Erfüllung der in der Teststrategie festgelegten Testzielsetzung und Testziele nötig sind. Zur Testplanung gehört auch, Methoden zum Sammeln und Verfolgen der Metriken zu identifizieren, die eingesetzt werden sollen, um das Projekt zu führen, damit festgestellt werden kann, ob die Testplanung eingehalten wird und um zu bewerten, ob die Ziele erreicht sind. Die Bestimmung nützlicher Metriken in der Testplanungsphase ermöglicht es, Testwerkzeuge auszuwählen, Schulungsmaßnahmen zu planen und Dokumentationsrichtlinien zu erstellen.

Die für das Testprojekt gewählte Strategie (oder Strategien) ist hilfreich bei der Festlegung der Aufgaben, die in der Testplanungsphase erfolgen sollten. Wird beispielsweise eine risikoorientierte Teststrategie(n) verfolgt (siehe Kapitel 2), dann liefert die Risikoanalyse Informationen für den Testplanungsprozess über geeignete Maßnahmen zur Reduzierung der identifizierten Produktrisiken, sowie für die Planung für den Fall des Eintretens. Falls mehrere wahrscheinliche und schwerwiegende potenzielle Fehlerzustände in Zusammenhang mit der Sicherheit des Softwaresystems identifiziert werden, dann sollten große Anstrengungen für die Entwicklung und Durchführung von

Sicherheitstests eingeplant werden. Ebenso verhält es sich, wenn bei einem bestimmten Projekt festgestellt wird, dass die schwerwiegendsten Fehlerzustände normalerweise in der Testentwurfsspezifikation zu finden sind. In einem solchen Fall könnte der Testplanungsprozess zusätzliche statische Tests (Reviews) der Testentwurfsspezifikation festlegen.

Informationen über Risiken können auch für die Priorisierung der verschiedenen Testaktivitäten verwendet werden. Wird beispielsweise die Performanz des Softwaresystems als hohes Risiko eingestuft, dann könnten Performanztests bereits durchgeführt werden, sobald integrierter Programmcode verfügbar ist. Oder falls eine reaktive Teststrategie verfolgt werden soll, dann kann es durchaus gerechtfertigt sein, dass die Erstellung von Test-Chartas und Werkzeuge für dynamische Testverfahren, wie z.B. exploratives Testen, eingeplant werden.

Außerdem ist der Testplanungsprozess die Phase, in der die Testvorgehensweise vom Testmanager klar definiert wird, einschließlich der vorgesehenen Teststufen, der Ziele für jede Teststufe und der Testverfahren, die in den jeweiligen Teststufen eingesetzt werden sollen. Beim risikoorientierten Test bestimmter Avioniksysteme schreibt beispielsweise eine Risikobewertung die benötigte Codeüberdeckung vor und damit auch, welche Testverfahren eingesetzt werden sollen.

Zwischen Testbasis (z.B. spezifische Anforderungen oder Risiken), Testbedingungen und den Tests, die diese abdecken, kann ein komplexes Beziehungsgeflecht bestehen. Zwischen diesen Arbeitsergebnissen bestehen oft viele wechselseitige Beziehungen. Tester müssen diese wechselseitigen Beziehungen verstehen, um eine effektive Testplanung, Testüberwachung und Teststeuerung durchführen zu können. Auch Entscheidungen bezüglich der Werkzeuge können vom Verständnis der wechselseitigen Beziehungen zwischen den Arbeitsergebnissen abhängen.

Weitere Beziehungen können zwischen den Arbeitsergebnissen des Entwicklungsteams und des Testteams bestehen. So kann eine Rückverfolgbarkeitsmatrix hilfreich sein, die Beziehungen zwischen Elementen der detaillierten Entwurfsspezifikation der Systementwickler, den fachlichen Anforderungen der Fachanalytiker und den vom Testteam definierten Arbeitsergebnissen transparent zu machen. Falls konkrete Testfälle entworfen und verwendet werden sollen, so kann es der Testplanungsprozess eventuell verlangen, möglicherweise die Anforderung festlegen, dass die detaillierten Entwurfsdokumente der Entwicklungsabteilung genehmigt werden müssen, bevor mit der Erstellung der Testfälle begonnen wird. In einem agilen Lebenszyklus können in einem informellen Informationsaustausch die Informationen zwischen den Teams kommuniziert werden, bevor mit dem Testen begonnen wird.

Im Testkonzept können auch die spezifisch zu testenden Features aufgelistet sein (evtl. basierend auf der Risikoanalyse) und es können ausdrücklich jene Features identifiziert sein, die nicht getestet werden sollen. Je nach der Formalität des Projekts und Umfang und Tiefe der benötigten Dokumentation kann jedem zu testenden Testobjekt eine entsprechende Testentwurfsspezifikation zugewiesen werden.

In dieser Phase (Testplanung) ist es in manchen Fällen auch erforderlich, dass der Testmanager mit dem Softwarearchitekten im Projekt zusammen arbeitet, um folgendes zu erreichen: die Testumgebungen initial festzulegen, die Verfügbarkeit aller benötigten Ressourcen zu überprüfen, die Beauftragung von Administratoren für die Umgebungskonfiguration sicherzustellen, sich mit Kosten- und Terminvorgaben vertraut zu machen und alle relevanten Aufgaben zur Erstellung und Lieferung der Testumgebung auszuarbeiten.

Schließlich sollten alle externen Abhängigkeiten und die zugehörigen Service Level Vereinbarungen (SLVs, engl. Service Level Agreements (SLAs)) identifiziert und gegebenenfalls Kontakt aufgenommen werden. Beispiele für derartige Abhängigkeiten sind Ressourcen, die von externen Gruppen angefordert werden, sowie Abhängigkeiten von anderen Projekten (falls das Projekt Teil

eines Programms ist), von externen Lieferanten oder Entwicklungspartnern, vom Installationsteam und von Datenbankadministratoren.

1.2.2 Testüberwachung und -steuerung

Damit der Testmanager das Testen effektiv steuern kann, muss ein Testplanungs- und Überwachungsrahmen geschaffen werden, der es ermöglicht, die Arbeitsergebnisse und Ressourcen zu verfolgen und mit dem Plan zu vergleichen. Dieses Rahmenwerk sollte die detaillierten Maßnahmen und Zielvorgaben beinhalten, die benötigt werden, um den aktuellen Status der Testarbeitsergebnisse und Testaktivitäten mit dem Plan und den strategischen Zielen in Beziehung zu setzen.

Bei kleinen und weniger komplexen Projekten mag es relativ einfach sein, die Testarbeitsergebnisse und Testaktivitäten mit dem Plan und den strategischen Zielen in Beziehung zu setzen. Im Allgemeinen müssen jedoch detailliertere Ziele spezifiziert werden, um dies zu ermöglichen. Dies kann die Maßnahmen und Ziele zur Erreichung der Testziele und Überdeckung der Testbasis einschließen.

Von besonderer Bedeutung ist, dass der aktuelle Status der Testarbeitsergebnisse und Testaktivitäten mit der Testbasis in einer Art und Weise in Beziehung gesetzt wird, die für Projektbeteiligte und Stakeholder des Fachbereichs nachvollziehbar und relevant ist. Dabei hilft das Festlegen von Zielen und das Messen des Testfortschritts basierend auf Testbedingungen und Gruppen von Testbedingungen, wodurch andere Testarbeitsergebnisse mit der Testbasis über die Testbedingungen in Beziehung gesetzt werden. Richtig konfigurierte Rückverfolgbarkeit, einschließlich der Berichterstattung über den Status der Rückverfolgbarkeit, machen das komplexe Beziehungsgeflecht zwischen Entwicklungsarbeitsergebnissen, der Testbasis und den Testarbeitsergebnissen transparenter und verständlicher.

Manchmal stehen die detaillierten Messwerte und Ziele, die für die Stakeholder überwacht werden müssen, nicht in einem direkten Bezug zur Systemfunktionalität oder zu einer Spezifikation, besonders wenn nur wenig oder keine formale Dokumentation vorhanden ist. So kann es einem Stakeholder aus dem Fachbereich möglicherweise wichtiger sein, Überdeckung von gelebten Geschäftsvorfällen zu messen, als nach den ausgearbeiteten Systemfunktionalitäten in der Anforderungsspezifikation. Daher erweist sich die frühe Einbindung von Stakeholdern des Fachbereichs für die Festlegung solcher Messwerte und Ziele als sehr hilfreich. Diese können dann nicht nur zur besseren Teststeuerung im Projektverlauf verwendet werden, sondern ermöglichen auch, die Testaktivitäten während des Projektes voranzutreiben und zu beeinflussen. Beispielsweise können die von den Stakeholdern vorgegebenen Messwerte und Ziele verwendet werden, um die Arbeitsergebnisse aus Testentwurf, Testdurchführung und/oder Testausführungsplan zu strukturieren, was letztendlich die genaue Überwachung des Testfortschritts gegen diese Messwerte erleichtert. Schließlich ermöglicht das auch die Rückverfolgbarkeit der Information aus einer bestimmten Teststufe oder eventuell mehrerer Teststufen zugleich.

Die Teststeuerung ist eine fortlaufende Aktivität, bei der der tatsächliche Testfortschritt mit dem Plan verglichen wird und bei Abweichungen Korrekturmaßnahmen eingeleitet werden. Die Teststeuerung steuert den Testprozess, damit übergeordnete Zielsetzung, Teststrategie und Testziele erreicht werden. Hierzu gehört bei Bedarf auch eine Anpassung der Testplanungsaktivitäten. Angemessene Reaktionen auf die vorliegenden Kontrolldaten sind angewiesen auf detaillierte Testplanungsinformationen.

Der Inhalt der Testplanungsdokumente und die Teststeuerungsaktivitäten werden in Kapitel 2 behandelt.

1.3 Testanalyse

Anstatt Testanalyse und Testentwurf zusammen zu betrachten wie im Foundation Level Syllabus, werden diese in den Advanced Level Lehrplänen als zwei separate Aktivitäten behandelt, auch wenn anerkannt wird, dass diese zur Erstellung der Arbeitsergebnisse dieser Phase selbstverständlich auch als parallele, integrierte oder iterative Aktivitäten durchgeführt werden können.

Die Testanalyse ist die Aktivität, die in Form von Testbedingungen identifiziert, "was" zu testen ist. Die Testbedingungen lassen sich durch eine Analyse der Testbasis, der Testziele und der Produktrisiken identifizieren. Sie können als die detaillierten Maßnahmen und Ziele für den Erfolg (z.B. als Teil der Endekriterien) gelten und sie sollten rückverfolgbar sein zur Testbasis und zu den festgelegten strategischen Zielen, einschließlich der Testziele und weiterer von Projektbeteiligten und Stakeholdern des Fachbereichs vorgegebenen Erfolgskriterien. Die Testbedingungen sollten außerdem auch zum Testentwurf und anderen Arbeitsergebnissen verfolgbar sein, sobald diese erstellt werden.

Die Testanalyse für eine Teststufe kann erfolgen, sobald die Testbasis für die betroffene Stufe vorliegt. Zur Identifizierung der Testbedingungen können formale Testverfahren und sonstige analytische Verfahren (z.B. risikoorientierte oder anforderungsbasierte Analysestrategien) gewählt werden. In den Testbedingungen können Werte und Variablen spezifiziert werden. Dies ist jedoch nicht zwingend erforderlich, sondern ist abhängig von der Teststufe, den zum Zeitpunkt der Analyse vorliegenden Informationen und dem gewählten Detaillierungsgrad (d.h. der Granularität der Dokumentation).

Es gibt eine Reihe von Faktoren, die bei der Entscheidung bezüglich der Granularität der zu identifizierenden Testbedingungen berücksichtigt werden sollten. Dazu gehören:

- Teststufe
- Detaillierungsgrad und Qualität der Testbasis
- Komplexität von System bzw. Software
- Projekt- und Produktrisiken
- Das Beziehungsgeflecht zwischen Testbasis, dem was getestet werden soll und wie es getestet werden soll
- Verwendeter Softwarelebenszyklus
- Verwendetes Testmanagementwerkzeug
- Detaillierungsgrad, mit dem Testentwurf und weitere Arbeitsergebnisse spezifiziert und dokumentiert werden sollen
- Fähigkeiten und Wissen der Test Analysten
- Reifegrad des Testprozesses und des Unternehmens an sich (es ist zu beachten, dass ein höherer Reifegrad einen höheren Detaillierungsgrad erforderlich machen, oder einen geringeren Detaillierungsgrad erlauben kann)
- Verfügbarkeit anderer im Projekt involvierter Personen für Beratungszwecke

Wenn die Testbedingungen detailliert spezifiziert werden, dann resultiert dies gewöhnlich in einer größeren Anzahl von Testbedingungen. Für eine E-Commerce-Anwendung kann beispielsweise als eine einzige allgemeine Testbedingung "Kassenfunktion testen" formuliert sein. In einem Dokument mit detaillierten Testbedingungen kann diese Testbedingung jedoch in mehrere Testbedingungen aufgeteilt sein, z.B. eine Testbedingung für jede unterstützte Zahlungsmethode, eine Testbedingung für jedes Zielland usw.

Für die Spezifikation von detaillierten Testbedingungen sprechen einige Vorteile. Dazu gehören:

- Mehr Flexibilität, wenn andere Arbeitsergebnisse (z.B. Testfälle) mit Testbasis und Testzielen in Beziehung gebracht werden sollen. Dies ermöglicht dem Testmanager einer bessere und detailliertere Testüberwachung und -steuerung.

- Wie im Foundation Level Syllabus beschrieben, ist die Spezifikation detaillierter Testbedingungen für die Fehlervorbeugung förderlich, da dies bereits in den frühen Projektphasen für die höheren Teststufen erfolgen kann, sobald die Testbasis vorliegt und potenziell bevor Systemarchitektur und detaillierter Entwurf verfügbar sind.
- Detaillierte Testbedingungen stellen einen Bezug her zwischen Testarbeitsergebnissen und Stakeholdern und zwar in einer Form, die für diese verständlich ist (häufig ergeben Testarbeitsergebnisse für die Stakeholder des Fachbereichs keinen Sinn und einfache Metriken, wie z.B. die Anzahl ausgeführter Testfälle, geben keinen Aufschluss über die Überdeckungsanforderungen von Stakeholdern).
- Mit Hilfe von detaillierten Testbedingungen lassen sich nicht nur andere Testaktivitäten beeinflussen und lenken, sondern auch andere Entwicklungsaktivitäten.
- Detaillierte Testbedingungen ermöglichen eine Optimierung von Testentwurf, Testrealisierung und Testdurchführung, sowie der daraus resultierenden Arbeitsergebnissen durch eine effizientere Abdeckung der detaillierten Messwerte und Ziele.
- Detaillierte Testbedingungen schaffen die Basis für eine klarere horizontale Rückverfolgbarkeit innerhalb einer Teststufe.

Zu den Nachteilen einer detaillierten Spezifikation von Testbedingungen gehören:
- Potenziell hoher Zeitaufwand
- Die Wartbarkeit kann in einer sich verändernden Umgebung schwierig werden
- Der Grad der Formalität muss festgelegt und für das gesamte Team umgesetzt werden

In den folgenden Situationen kann die Spezifikation detaillierter Testbedingungen besonders effektiv sein:
- Es erfolgt nur eine minimale Testentwurfsdokumentation (z.B. mit Checklisten), um dem Entwicklungslebenszyklus entgegenzukommen, aufgrund von Kosten- und/oder Zeitzwängen, oder aufgrund anderer Faktoren.
- Es stehen nur wenig oder gar keine formalen Anforderungen oder Arbeitsergebnisse der Entwicklung als Testbasis zur Verfügung.
- Das Projekt ist groß, komplex oder das Risiko ist hoch und es wird daher mehr Testüberwachung und Teststeuerung benötigt, die nicht möglich ist, wenn einfach nur die Testfälle mit Entwicklungsarbeitsprodukten in Beziehung gebracht werden.

Testbedingungen können weniger detailliert spezifiziert werden, wenn die Testbasis einfach und direkt mit den Arbeitsergebnissen des Testentwurfs in Beziehung gebracht werden kann. Dies ist in den folgenden Fällen eher wahrscheinlich:
- Komponententeststufe
- Bei weniger komplexen Projekten, in denen es einfache hierarchische Beziehungen gibt zwischen dem was getestet werden soll und wie es getestet werden soll
- Im Abnahmetest, wo Anwendungsfälle zur Festlegung der Tests dienen können

1.4 Testentwurf

Der Testentwurf ist die Aktivität, die definiert, “wie” etwas getestet werden soll. Dies beinhaltet die Identifizierung von Testfällen durch die schrittweise Ausarbeitung und Verfeinerung der identifizierten Testbedingungen oder der Testbasis mit Hilfe der Testverfahren, die in der Teststrategie und/oder dem Testkonzept festgelegt sind.

Je nach den Verfahren, die für Testüberwachung, Teststeuerung und Rückverfolgbarkeit zum Einsatz kommen, können die Testfälle direkt (oder indirekt über die Testbedingungen) mit der Testbasis und den festgelegten Zielen in Beziehung gebracht werden. Diese Ziele beinhalten strategische Ziele,

Testziele und weitere, von Projektbeteiligten und Involvierten des Fachbereichs vorgegebene Erfolgskriterien.

Der Testentwurf für eine Teststufe kann erfolgen, sobald die Testbedingungen identifiziert und ausreichend Informationen zur Verfügung stehen, damit Testfälle entworfen werden können. Diese können entweder konkret oder abstrakt sein, je nach dem für den Testentwurf angewendeten Verfahren. Für die höheren Teststufen ist der Testentwurf wahrscheinlich eine separate Aktivität, der die Testanalyse vorausgeht. Für die niedrigeren Teststufen ist es wahrscheinlich, dass Testanalyse und Testentwurf als eine integrierte Aktivität durchgeführt werden.

Es ist außerdem wahrscheinlich, dass einige Aufgaben, die in Zusammenhang mit der Aktivität der Testrealisierung beschrieben sind, in den Testentwurfsprozess integriert werden, z.B. die Erzeugung von Testdaten. Dies ist der Fall, wenn ein iterativer Ansatz zur Erstellung der für die Durchführung benötigten Tests verfolgt wird. Mit einer solchen Vorgehensweise lässt sich die Überdeckung von Testbedingungen sogar optimieren, indem dabei konkrete oder abstrakte Testfälle entworfen werden.

1.5 Testrealisierung

Die Testrealisierung ist die Aktivität, bei der die Tests von den Test Analysten organisiert und priorisiert werden. In einem formal dokumentierten Kontext ist die Testrealisierung die Aktivität, bei der Testentwürfe in konkrete Testfälle, Testabläufe und Testdaten umgesetzt werden. In Unternehmen, die den IEEE Standard 829 [IEEE 829] befolgen, werden die Testeingaben und die zugehörigen erwarteten Ergebnisse in Testfallspezifikationen und die Testschritte in Testablaufspezifikationen erfasst. Häufiger werden jedoch die Eingaben, erwarteten Ergebnisse und Testschritte eines Tests zusammen dokumentiert. Die Testrealisierung schließt auch die Erstellung gespeicherter Testdaten mit ein (z.B. in „einfachen“ Textdateien oder Datenbanktabellen).

Zur Testrealisierung gehört auch, endgültig zu prüfen und sicherzustellen, dass das Testteam für die Testdurchführung bereitsteht. In diesem Zusammenhang kann außerdem auch die Lieferung der benötigten Testumgebung, Testdaten und Programmcode sichergestellt werden (eventuell erfolgen einige Abnahmetests für Testumgebung und/oder Programmcode) und es wird überprüft, ob alle Testfälle erstellt, geprüft und ausgeführt werden können. Zur Testrealisierung kann auch eine Überprüfung anhand von expliziten und impliziten Eingangskriterien für die betroffene Teststufe gehören (siehe Abschnitt 1.7). Auch eine detaillierte Beschreibung der Testumgebung und Testdaten kann Teil der Testrealisierung sein.

Der in der Testrealisierung notwendige Detaillierungsgrad und die damit verbundene Komplexität der Aufgaben können durch den Detaillierungsgrad anderer Testarbeitsergebnisse (z.B. Testfälle und Testbedingungen) beeinflusst werden. In manchen Fällen, besonders wenn Tests für eine langfristige Wiederverwendung im Regressionstest archiviert werden sollen, können die Tests eine detaillierte Beschreibung der für die Durchführung des Tests notwendigen Schritte enthalten. Damit soll eine zuverlässige und konsistente Testdurchführung sichergestellt werden und zwar unabhängig vom Tester, der die Tests durchführt. Falls gesetzliche Bestimmungen gelten, müssen die Tests den Nachweis erbringen, dass die gültigen Normen und Standards tatsächlich erfüllt sind (siehe Abschnitt 2.9).

In der Testrealisierungsphase sollten die Tester die Reihenfolge der durchzuführenden manuellen und automatisierten Tests in einem Testausführungsplan endgültig festlegen. Dabei müssen die Testmanager etwaige Einschränkungen, einschließlich der Risiken und Priorisierung, genau beachten, die möglicherweise eine bestimmte Reihenfolge der Tests oder eine bestimmte Ausrüstung für die Durchführung vorgeben. Abhängigkeiten von bestimmten Testumgebungen oder Testdaten müssen bekannt sein und überprüft werden.

Eine frühe Testrealisierung kann auch Nachteile haben. Bei einem agilen Lebenszyklus kann sich beispielsweise der Programmcode von Iteration zu Iteration dramatisch ändern, wodurch viele der Testrealisierungsaufgaben wieder überholt sind. Aber auch bei einem anderen iterativen oder inkrementellen Lebenszyklus, der nicht ganz so stark von Änderungen geprägt ist wie der agile, kann es zwischen den Iterationen zu signifikanten Änderungen kommen. Dies macht das Testen mit Testskripten unzuverlässig, oder es ist viel Wartungsaufwand erforderlich. Das Gleiche gilt für schlecht geführte sequenzielle Lebenszyklen, bei denen sich die Anforderungen häufig ändern, sogar noch in den späten Projektphasen. Bevor sehr viel Aufwand für die Testrealisierung betrieben wird, macht es Sinn, den Softwarelebenszyklus zu verstehen und wie berechenbar die Features einzuschätzen sind, die zum Testen zur Verfügung stehen.

Eine frühe Testrealisierung kann allerdings auch Vorteile haben. Beispielsweise liefern konkrete Tests Beispiele dafür, wie sich die Software verhalten sollte, wenn diese der Testbasis entspricht. Für Experten des Fachbereiches ist die Verifizierung von konkreten Tests wahrscheinlich einfacher als die Verifizierung abstrakter Geschäftsregeln und sie können dabei weitere Schwächen in den Softwarespezifikationen identifizieren. Die verifizierten Tests können den Softwareentwicklern und -designern das benötigte Verhalten der Software deutlich vor Augen führen.

1.6 Testdurchführung

Die Testdurchführung beginnt, sobald das Testobjekt zur Verfügung steht und die Eingangskriterien für die Testdurchführung erfüllt sind. Die Tests sollten vor der Testdurchführung entworfen oder aber wenigstens definiert sein. Die vorgesehenen Werkzeuge sollten einsatzbereit sein, insbesondere die Werkzeuge für das Testmanagement, Fehlerverfolgung und (falls zutreffend) für die automatisierte Testdurchführung. Das Verfolgen der Testergebnisse (einschließlich der Metriken) sollte funktionieren und die dabei gesammelten Daten sollten von allen verstanden werden. Die Standards für die Testprotokollierung und Fehlerberichte sollten zur Verfügung stehen und bekannt gegeben werden. Wenn sichergestellt wird, dass all dies vor der Testdurchführung bereitsteht, können die Tests effizient durchgeführt werden.

Die Tests sollten gemäß den Testfällen durchgeführt werden, obwohl der Testmanager den Testern einen gewissen Spielraum einräumen sollte, damit sie zusätzliche interessante Testszenarien und Testverhalten verfolgen können, die sich erst während des Testens ergeben. Falls eine Teststrategie verfolgt wird, die zumindest in Teilen reaktiv ist, dann sollte Zeit für Testsitzungen mit erfahrungsbasierten und fehlerbasierten Testverfahren eingeräumt werden. Werden bei einer Abweichung vom festgelegten Testvorgehen Fehlerwirkungen aufgedeckt, müssen die Änderungen des ursprünglichen Testfalls so dokumentiert werden, dass sich die Fehlerwirkungen reproduzieren lassen. Automatisierte Tests laufen ohne Abweichung von den definierten Vorgaben ab.

Die Hauptaufgabe des Testmanagers bei der Testdurchführung besteht darin, den Fortschritt anhand des Testkonzepts zu überwachen und bei Bedarf geeignete Korrekturmaßnahmen zu initiieren und durchzuführen, um einen erfolgreichen Abschluss der Tests hinsichtlich der übergeordneten Zielsetzung, Teststrategie und Testziele herbeizuführen. Zu diesem Zweck kann der Testmanager die Rückverfolgbarkeit von den Testergebnissen zu den Testbedingungen, der Testbasis und letztlich zu den Testzielen verwenden und auch umgekehrt von den Testzielen zu den Testergebnissen. Dieser Prozess wird in Abschnitt 2.6 genau beschrieben.

1.7 Bewertung von Endekriterien und Bericht

Weitere Informationen über Dokumentation und Bericht der Testfortschrittsüberwachung enthält Abschnitt 2.6.

Für den Testprozess geht es bei der Überwachung des Testfortschritts vor allem darum, dass effektive Prozesse existieren, die die für die Bewertung der Endekriterien und die Berichte erforderlichen Informationen liefern.

Die Spezifikation der benötigten Informationen und der Methoden für das Sammeln dieser Informationen sind Teil der Testplanung, Testüberwachung und Teststeuerung. Bei der Testanalyse, Testentwurf, Testrealisierung und Testdurchführung sollte der Testmanager sicherstellen, dass die Mitglieder des Testteams, die dafür zuständig sind, die benötigten Informationen pünktlich und genau zur Verfügung stellen, um eine effektive Testberichterstattung und Bewertung der Endekriterien zu ermöglichen.

Die Häufigkeit und der Detaillierungsgrad der Testberichterstattung sind abhängig vom Projekt und von dem Unternehmen. Dies sollte während der Testplanungsphase ausgehandelt werden, wobei auch eine Beratung mit relevanten Projektbetroffenen erfolgen sollte.

1.8 Abschluss der Testaktivitäten

Nachdem die Testdurchführung als abgeschlossen erklärt wurde, sollten die wichtigsten Ergebnisse festgehalten und entweder an die zuständige Person weitergeleitet oder archiviert werden. Man bezeichnet das als den Abschluss der Testaktivitäten. Testabschlussaktivitäten können in die folgenden vier Hauptgruppen eingeteilt werden:

1. Testendeprüfung - Diese soll sicherstellen, dass alle geplanten Testaufgaben tatsächlich abgeschlossen sind. So sollten beispielsweise alle geplanten Tests entweder durchgeführt oder gezielt übersprungen worden sein. Alle bekannten Fehlerzustände sollten entweder behoben und nachgetestet, oder auf einen zukünftigen Release verschoben worden sein, oder aber als permanente Einschränkung akzeptiert sein.
2. Übergabe der Testmittel - Die nützlichen Arbeitsergebnisse werden den Personen oder Stellen geliefert, die sie benötigen. Beispielsweise sollten diejenigen, die das System benutzen oder seine Benutzung betreuen werden, erfahren, welche bekannten Fehlerzustände verschoben oder akzeptiert wurden. Die für die Wartungstests zuständigen Personen sollten Tests und Testumgebung erhalten. Außerdem sollte das Wartungsteam einen Satz manueller oder automatisierter Regressionstests erhalten.
3. Bewertungssitzungen - Besprechungen über die Erfahrungen aus dem Testen (engl. “lessons learned”) halten oder an ihnen teilnehmen. Hier können wichtige Erkenntnisse, sowohl aus dem Testprojekt als auch aus dem gesamten Softwarelebenszyklus, dokumentiert werden. In diesen Sitzungen werden Maßnahmen identifiziert, die sicherstellen, dass gute Praktiken wiederholt werden und schlechte in Zukunft vermieden werden. Falls Probleme sich nicht lösen lassen, können sie zumindest in zukünftigen Projektplänen berücksichtigt werden. Es können folgende Bereiche betrachtet werden:
    a. Weil unerwartete Fehleranhäufungen erst relativ spät aufgedeckt werden konnten, kommt das Team eventuell zur Erkenntnis, dass an den Besprechungen zur Qualitätsrisikoanalyse bei zukünftigen Projekten eine breitere Auswahl von Benutzern teilnehmen sollte.
    b. Möglicherweise war die Testschätzung nicht sehr zutreffend, sodass sich hieraus Lehren für zukünftige Schätzaktivitäten ziehen lassen, vor allem aus den Ursachen. So kann es an ineffektivem Testen gelegen haben, oder die Schätzung war von vornherein zu niedrig angesetzt.
    c. Die Tester können Tendenzen und Ergebnisse der Ursache–Wirkungs-Analyse der Fehlerzustände bewerten. Sie können bewerten, ob späte Änderungsanträge die Qualität von Analyse und Entwicklung beeinflusst haben. Die Tester können auch nach ungeeigneten Praktiken suchen und wie viel Zeit diese gekostet haben, wie z.B. das Auslassen einer Teststufe, die die Fehlerzustände früher und daher kosteneffektiver

aufgedeckt hätte. Einige Fehlertendenzen lassen sich auch mit Rahmenbedingungen in Verbindung bringen, wie dem Einsatz neuer Technologien, Personalwechsel oder fehlender fachlicher Qualifikation.
   d. Mögliche Verbesserungen für den Testprozess identifizieren.
   e. Gab es unerwartete Abweichungen vom Plan, die bei zukünftigen Planungen berücksichtigt werden sollten?
4. Archivierung: Ergebnisse, Protokolle, Berichte und andere Dokumente und Arbeitsergebnisse werden im Konfigurationsmanagementsystem archiviert. So sollten beispielsweise Testkonzept und Projektplan im Planungsarchiv archiviert werden und eindeutig auf das System und die Version verweisen, für die sie eingesetzt wurden.

All diese Aufgaben sind wichtig, auch wenn sie oft vergessen werden und sie sollten explizit ein Teil des Testkonzepts sein.
Oft werden eine oder mehrere Aufgaben in diesem Zusammenhang ausgelassen, meist weil das Testteam zu früh aufgelöst wird, weil Zeit und Ressourcen für nachfolgende Projekte benötigt werden, oder weil das Projektteam überarbeitet und erschöpft ist. Bei Vertragsprojekten, z.B. bei kundenindividuellen Entwicklungsaufträgen, sollten die notwendigen Aufgaben im Vertrag spezifiziert sein.

2. Testmanagement – [750 Minuten]

Begriffe

Mastertestkonzept, Produktrisiko, Projektrisiko, Qualitätsrisiko, Risiko, Risikoanalyse, Risikobeherrschung[1], Risikobewertung, Risikoidentifizierung, Risikomanagement, risikoorientierter Test, Risikostufe, Stufentestkonzept, Testbedingungen, Leitender Testmanager, Testkonzept, Testleiter, Testmanagement, Testrichtlinie, Testschätzung, Teststeuerung, Teststrategie, Teststufe, Testüberwachung, Testvorgehensweise, Breitband-Delphi

Lernziele für das Testmanagement

2.2 Testmanagement im Kontext

TM-2.2.1 (K4) Sie können die relevanten Stakeholder, die Rahmenbedingung und die Erfordernisse eines Softwareprojekts oder -programms ermitteln, wobei sie auf das jeweilige Softwarelebenszyklusmodell Bezug nehmen und daraus die optimalen Testaktivitäten ableiten

TM-2.2.2 (K2) Sie verstehen, wie die Aktivitäten und Arbeitsergebnisse im Softwarelebenszyklus das Testen beeinflussen und wie sich umgekehrt das Testen auf die Aktivitäten im Softwarelebenszyklus und die Arbeitsergebnisse auswirkt

TM-2.2.3 (K2) Sie können erklären, was für Besonderheiten beim Testmanagement im Zusammenhang mit erfahrungsbasiertem Test und nicht-funktionalem Test zu beachten sind und wie man sie am Besten in den Griff bekommt

2.3 Risikoorientierter Test und andere Ansätze für Testpriorisierung und Aufwandszuweisung

TM-2.3.1 (K2) Sie können erläutern, auf welche unterschiedliche Art und Weise risikoorientiertes Testen auf Risiken reagiert

TM-2.3.2 (K2) Sie können anhand von Beispielen die unterschiedlichen Verfahren zur Erstellung einer Produktrisikoanalyse erklären

TM-2.3.3 (K4) Sie können Produktrisiken identifizieren und bewerten und sie können die Risiken sowie deren ermittelte Risikostufe aus der Sicht der wichtigsten Stakeholder des Projekts zusammenfassen

TM-2.3.4 (K2) Sie können die für die ermittelte Risikostufe angemessenen Maßnahmen zur Risikobeherrschung und zum Management von identifizierten Produktrisiken darlegen, und zwar der Produktrisiken, die während des gesamten Softwarelebenszyklus und im Testprozess auftreten können

TM-2.3.5 (K2) Sie können anhand von Beispielen die unterschiedlichen Möglichkeiten der Testauswahl, Testpriorisierung und Aufwandszuweisung erläutern

[1] Gebräuchliche Synonyme in diesem Kontext: Risikovermeidung, Risikosteuerung und –bewältigung, Risikoüberwachung

2.4 Testdokumentation und weitere Arbeitsergebnisse

TM-2.4.1 (K4) Sie können vorgegebene Testrichtlinien und Teststrategien analysieren und Mastertestkonzepte, Stufentestkonzepte sowie weitere Testarbeitsergebnisse erstellen, in denen die Vorgaben dieser Dokumente vollständig und konsistent umgesetzt sind.

TM-2.4.2 (K4) Sie können Projektrisiken für ein Projekt analysieren und geeignete Maßnahmen des Risikomanagements bestimmen (d.h. das Risiko beherrschen, Notfallpläne erstellen, das Risiko auslagern und/oder das Risiko akzeptieren)

TM-2.4.3 (K2) Sie können anhand von Beispielen erläutern, wie sich Teststrategien auf die Testaktivitäten auswirken

TM-2.4.4 (K3) Sie können Dokumentationsstandards und -vorlagen für Testarbeitsergebnisse definieren und diese an den Bedarf eines Unternehmens/einer Organisation, an den Softwarelebenszyklus und an ein bestimmtes Projekt anpassen. Bei Bedarf können Sie vorhandene Vorlagen von Normungsorganisationen für den eigenen Zweck adaptieren

2.5 Testschätzung

TM-2.5.1 (K3) Sie können eine Testschätzung für alle Testprozessaktivitäten eines Projekts durchführen und je nach Bedarf alle geeigneten Verfahren zur Testschätzung anwenden.

TM-2.5.2 (K2) Sie kennen die Faktoren, die die Testschätzung beeinflussen können und können diese anhand von Beispielen erläutern

2.6 Definieren und Anwenden von Testmetriken

TM-2.6.1 (K2) Sie können typische testbezogene Metriken beschreiben und vergleichen

TM-2.6.2 (K2) Sie können die verschiedenen Dimensionen zur Überwachung und Steuerung des Testfortschritts vergleichen

TM-2.6.3 (K4) Sie können im Bezug auf Restrisiken, Fehlerstatus, den aktuellen Status der Testdurchführung, Testüberdeckung und das Vertrauen in die Software Testergebnisse auswerten und über sie berichten, damit Projekt-Stakeholder durch das Gewähren von Einblicken und das Aussprechen von Empfehlungen, Release-Entscheidungen treffen können

2.7 Mehrwert des Testens

TM-2.7.1 (K2) Sie können Beispiele für jede der vier Kategorien geben, die die Qualitätskosten bestimmen

TM-2.7.2 (K3) Sie können den Mehrwert des Testens auf Basis der Qualitätskosten sowie anderer quantitativen und qualitativen Parameter schätzen und Stakeholder über den geschätzten Nutzen des Testens informieren

2.8 Verteiltes Testen, Outsourcing und Insourcing

TM-2.8.1 (K2) Sie können die Faktoren eines erfolgreichen Einsatzes der verschiedenen Personalorganisations-Modelle (verteiltes Testen, Outsourcing und Insourcing des Testens) nennen

2.9 Die Anwendung von Industriestandards managen

TM-2.9.1 (K2) Sie können die Quellen und Anwendungen von Industriestandards für das Softwaretesten zusammenfassen

2.1 Einführung

Beim Advanced Level hat eine Spezialisierung in der Karriere der professionellen Testexperten eingesetzt. In diesem Kapitel werden schwerpunktmäßig Wissensbereiche behandelt, die für diejenigen professionellen Tester wichtig sind, die eine Position als Testleiter, Testmanager oder leitender Testmanager anstreben. Der Syllabus verwendet für all diese Positionen die Bezeichnung „Testmanager“ als Oberbegriff, wohlwissend, dass unterschiedliche Unternehmen für Positionen, Zuständigkeiten und Verantwortungsbereiche unterschiedliche Bezeichnungen verwenden.

2.2 Testmanagement im Kontext

Eine zentrale Verantwortung eines Managers besteht darin, die Schaffung und Nutzung von benötigten Ressourcen (Personen, Software, Hardware, Infrastruktur usw.) sicherzustellen, um für Mehrwert schaffende Prozesse zu sorgen. Für Manager in der Software- und IT-Industrie sind diese Prozesse oft Teil eines Projekts oder Programms, das eine Software oder ein System für interne oder externe Zwecke liefern soll. Für Testmanager geht es bei den Prozessen um das Testen, insbesondere um die Aktivitäten in Zusammenhang mit dem fundamentalen Testprozess, der im Foundation Level Syllabus und in Kapitel 1 des vorliegenden Lehrplans beschrieben ist. Da Testprozesse nur Mehrwert schaffen, wenn sie zum Erfolg des Projekts bzw. Programms beitragen (oder schwerere Fehlerwirkungen oder Ausfälle verhindern), muss der Testmanager die Prozesse entsprechend planen und steuern. Konkret heißt das, dass der Testmanager die Testprozesse, einschließlich der damit verbundenen Aktivitäten und Arbeitsergebnisse, entsprechend der Erfordernisse und Umstände der anderen Stakeholder, deren Aktivitäten (z.B. Softwareentwicklungslebenszyklus, in dem getestet wird) und deren Arbeitsergebnisse (z.B. Anforderungsspezifikationen) gestaltet.

2.2.1 Die Interessensvertreter (Stakeholder) des Testens verstehen

Stakeholder sind Interessensvertreter des Testens, wenn sie ein Interesse an den Testaktivitäten, den Testarbeitsergebnissen oder an der Qualität des zu liefernden Systems haben. Das Interesse der Stakeholder kann in einer direkten oder indirekten Beteiligung an den Testaktivitäten bestehen, oder die Testarbeitsergebnisse sind direkt oder indirekt für sie bestimmt, oder sie sind direkt oder indirekt von der Qualität der zu liefernden Arbeitsergebnisse des Projekts oder Programms betroffen.

Die Stakeholder des Testens können je nach Projekt, Produkt, Unternehmen oder anderen Faktoren variieren; es können jedoch folgende Rollen zu den Stakeholdern gehören:

- Entwickler, Entwicklungsleiter und Entwicklungsmanager: Diese Stakeholder entwickeln die zu testende Software, erhalten die Testergebnisse und müssen darauf basierend dann ihrerseits wieder tätig werden (z.B. berichtete Fehlerzustände beheben).
- Datenbank-Architekten, Systemarchitekten und -designer: Diese Stakeholder entwerfen die Software, erhalten die Testergebnisse und müssen darauf basierend dann ihrerseits wieder tätig werden Maßnahmen aufgrund dieser Ergebnisse zu ergreifen.
- Mitarbeiter von Fachbereichen (z.B. Business Analysten): Diese Stakeholder legen die Features und deren innenwohnende Qualität fest, die in der Software vorhanden sein müssen. Sie sind auch häufig involviert, wenn die benötigte Testüberdeckung festgelegt wird, wenn Testergebnisse bewertet und wenn Entscheidungen auf Grundlage der Testergebnisse getroffen werden.
- Geschäftsleitung, Produktmanager und Projektsponsoren: Diese Stakeholder sind häufig beteiligt, wenn die benötigte Testüberdeckung festgelegt wird, wenn Testergebnisse bewertet und wenn Entscheidungen auf Grundlage der Testergebnisse getroffen werden.

- Projektmanager: Diese Stakeholder sind dafür verantwortlich, die Projekte zum Erfolg zu führen und in diesem Zusammenhang für einen Ausgleich zwischen verschiedenen Prioritäten (Qualität, Zeitplan, Features und Budget) zu sorgen. Häufig beschaffen sie die für die Testaktivitäten benötigten Ressourcen und arbeiten mit dem Testmanager bei Testplanung und Teststeuerung zusammen.
- Technischer Support, Kundenbetreuung und Helpdesk-Mitarbeiter: Diese Stakeholder unterstützen die Benutzer und Kunden, die von den Merkmalen und der Qualität der gelieferten Software profitieren.
- Direkte und indirekte Benutzer: Diese Stakeholder verwenden die Software direkt (als Endanwender), oder sie erhalten die von der Software erstellten Ausgaben oder Dienstleistungen.

Weitere Informationen über Interessensvertreter (Stakeholder) des Testens, siehe Kapitel 2 von [Goucher09].

Die Auflistung der Stakeholder erhebt keinen Anspruch auf Vollständigkeit, weshalb die Testmanager die einzelnen Stakeholder für ihr Projekt oder Programm identifizieren müssen. Die Testmanager müssen außerdem die genaue Beziehung der jeweiligen Stakeholder zum Testen verstehen und wie das Testteam deren Erfordernisse bedient. Zusätzlich zur Identifizierung der Stakeholder des Testens sollten die Testmanager die anderen Softwarelebenszyklus-Aktivitäten und Arbeitsergebnisse identifizieren, die das Testen beeinflussen bzw. die vom Testen beeinflusst werden. Ohne diese Maßnahmen wird der Testprozess möglicherweise nicht die optimale Effektivität und Effizienz erzielen (siehe Abschnitt 2.2.3).

2.2.2 Zusätzliche Softwarelebenszyklus-Aktivitäten und Arbeitsergebnisse

Da es sich beim Softwaretest letztlich um eine Bewertung der Qualität eines oder mehrerer Arbeitsergebnisse handelt, die außerhalb der Testaktivitäten erstellt wurden, ist das Testen gewöhnlich in den Kontext einer größeren Menge von Softwarelebenszyklus-Aktivitäten eingebettet. Der Testmanager muss bei Planung und Leitung der Testaktivitäten verstehen, wie sich diese anderen Aktivitäten und deren Arbeitsergebnisse auf das Testen auswirken (siehe Foundation Level Syllabus) und wie sich das Testen auf die anderen Aktivitäten und deren Arbeitsergebnisse auswirkt.

In Unternehmenen, in denen agile Entwicklungspraktiken angewandt werden, wird oft eine testgetriebene Softwareentwicklung durchgeführt, es werden automatisierte Modultests erstellt und der Code (einschließlich der Tests für diesen Code) wird kontinuierlich in das Konfigurationsmanagementsystem integriert. Der Testmanager sollte also mit dem Entwicklungsleiter zusammenarbeiten, um sicherzustellen, dass die Tester in diese Aktivitäten integriert sind bzw. die Aktivitäten mit den Testern koordiniert werden. Tester können die Modultests überprüfen, um sowohl Vorschläge für eine bessere Überdeckung und Effektivität dieser Tests zu machen, als auch um die Software und deren Implementierung besser zu verstehen. Tester sollten auch Möglichkeiten suchen, ihre eigenen automatisierten Tests, insbesondere funktionale Regressionstests, in das Konfigurationsmanagementsystem zu integrieren.

Während das Beziehungsgeflecht zwischen Testaktivitäten, den anderen Stakeholdern des Tests, den Aktivitäten des Softwareentwicklungslebenszyklus und den Arbeitsergebnissen je nach Projekt, gewähltem Softwareentwicklungslebenszyklus und weiteren unterschiedlichen Faktoren variiert, ist das Testen generell eng mit folgendem Tätigkeiten verbunden:

- Anforderungserstellung und -management. Bei Festlegung des Testumfangs und bei der Schätzung des Testaufwands muss der Testmanager die Anforderungen berücksichtigen und sich stets bewusst sein, dass sich diese ändern können. In diesem Fall müssen steuernde Maßnahmen zur Anpassung an die Änderungen ergriffen werden. Technische Test Analysten und Test Analysten sollten am Review der Anforderungen teilnehmen.

- Projektmanagement. Der Testmanager muss in Zusammenarbeit mit Test Analysten und Technischen Test Analysten den Projektmanager über Zeitplanung und die benötigten Ressourcen informieren. Der Testmanager muss mit dem Projektmanager zusammenarbeiten, um über Änderungen im Projektplan Bescheid zu wissen und ggf. dann rechtzeitig den Test danach neu auszurichten und steuernde Maßnahmen zur Anpassung an die Änderungen zu ergreifen.
- Konfigurations-, Release- und Änderungsmanagement. Der Testmanager muss in Zusammenarbeit mit dem Testteam die Prozesse und Mechanismen zur Lieferung der Testobjekte ausarbeiten und im Testkonzept festhalten. In diesem Zusammenhang kann der Testmanager Test Analysten und technische Test Analysten darum bitten, Tests zur Verifizierung von Softwareversionen zu erstellen und die Versionskontrolle während der Testdurchführung sicherzustellen.
- Softwareentwicklung und -wartung. Der Testmanager sollte mit Entwicklungsmanagern zusammenarbeiten, um die Lieferung von Testobjekten, einschließlich Inhalt und Termine für die einzelnen Test-Freigaben, zu koordinieren. Der Testmanager sollte am Fehlermanagement beteiligt sein (siehe Kapitel 4).
- Technischer Support. Der Testmanager sollte mit dem Technischen Support Manager zusammenarbeiten, um die korrekte Lieferung der Testergebnisse bei Abschluss der Testaktivitäten sicherzustellen, damit diejenigen, die für den Support der Software nach der Freigabe zuständig sind, über bekannte Fehlerwirkungen und Umgehungslösungen Bescheid wissen. Außerdem sollte der Testmanager zusammen mit dem Technischen Support Manager Produktionsausfälle analysieren, um Testprozessverbesserungen zu implementieren.
- Erstellung der technischen Dokumentation. Der Testmanager sollte mit dem Technischen Dokumentationsmanager zusammenarbeiten, um sicherzustellen, dass die Dokumentation für das Testen rechtzeitig geliefert wird. Außerdem muss sich der Testmanager in diesem Zusammenhang um das Management von Fehlerzuständen kümmern, die in den Dokumenten gefunden wurden.

Zusätzlich zur Identifizierung der Stakeholder des Testens (wie oben beschrieben) müssen die Testmanager die anderen Softwarelebenszyklus-Aktivitäten und Arbeitsergebnisse identifizieren, die das Testen beeinflussen bzw. die vom Testen beeinflusst werden. Ohne diese Maßnahmen wird der Testprozess nicht die optimale Effektivität und Effizienz erzielen.

2.2.3 Anpassungen der Testaktivitäten an andere Lebenszyklusaktivitäten

Testen sollte ein integraler Bestandteil des Projekts sein, egal welches der folgenden Softwareentwicklungsmodelle eingesetzt wird:

- Sequenzielle Modelle (Wasserfallmodell, V-Modell und W-Modell). In einem sequenziellen Modell werden alle Arbeitsergebnisse und Aktivitäten einer Phase (z.B. Anforderungen, Entwurf, Realisierung, Modultest, Integrationstest, Systemtest und Abnahmetest) vor Beginn der nächsten Phase abgeschlossen. Testplanung, Testanalyse, Testentwurf und Testrealisierung können sich mit Projektplanung, Analyse der Kundenanforderungen, Software- und Datenbankentwurf und Programmierung überlappen, wobei die genaue Art der Überlappung von der spezifischen Teststufe abhängt. Bei der Testdurchführung wird sequenziell entsprechend der einzelnen Teststufen vorgegangen, wie im Foundation Level Lehrplan und im vorliegenden Advanced Level Lehrplan beschrieben.
- Iterative oder inkrementelle Modelle (Rapid Application Development (RAD), Rational Unified Process (RUP)). Bei einem iterativen oder inkrementellen Modell werden die zu implementierenden Merkmale in Gruppen zusammengefasst (z.B. nach Geschäftspriorität oder nach Risiko) und dann laufen die verschiedenen Projektphasen, einschließlich der zugehörigen Arbeitsergebnisse und Aktivitäten, für die jeweiligen Gruppen von Merkmalen ab. Diese Phasen können sequenziell durchgeführt werden oder sich zeitlich überlappen und auch die Iterationen können sequenziell oder zeitlich überlappend ablaufen. In der Anfangsphase des Projekts werden die übergeordnete Testplanung und Testanalyse parallel

zur Projektplanung und Analyse der geschäftlichen Anforderungen durchgeführt. Detaillierte Testplanung, Testanalyse, Testentwurf und Testrealisierung erfolgen überlappend zu Beginn einer jeden Iteration. Jede Teststufe beginnt so früh wie möglich und kann nach Beginn nachfolgender, höherer Teststufen noch andauern.

- Agile Vorgehensweisen, wie z.B. SCRUM und Extreme Programming (XP). Hierbei handelt es sich um iterative Lebenszyklen mit sehr kurzen Iterationen (meist zwei bis vier Wochen). Die Arbeitsergebnisse und Aktivitäten einer jeden Iteration sind immer vor Beginn der nächsten Iteration abgeschlossen (d.h. die Iterationen sind sequenziell). Das Testen läuft ähnlich ab wie bei den iterativen Modellen, jedoch überlappen sich Test- und Entwicklungsaktivitäten weitaus mehr; es gibt sogar erhebliche Überlappungen zwischen Testdurchführung (auf verschiedenen Stufen) und Entwicklungsaktivitäten. All diese Aktivitäten einer Iteration, einschließlich der Testaktivitäten, sollten abgeschlossen sein, bevor die nächste Iteration beginnt. Bei agilen Projekten ändert sich die Rolle des Testmanagers von einer direkten Managerrolle zur Rolle eines technisch kompetenten Beraters.
- Spiralmodell. Beim Spiralmodell werden in frühen Projektphasen Prototypen eingesetzt, um die Machbarkeit zu bestätigen und mit Entwurfs- und Implementierungsentscheidungen zu experimentieren. Die Reihenfolge, in der die Prototyping Experimente durchgeführt werden, wird anhand von Geschäftspriorität und technischem Risiko festgelegt. Die Prototypen werden getestet, um herauszufinden, welche technischen Probleme noch nicht behoben sind. Nachdem die wichtigsten technischen Probleme gelöst sind, wird das Projekt nach einem sequenziellen oder iterativen Modell weitergeführt.

Um die Testaktivitäten richtig an den Lebenszyklus anzupassen, muss der Testmanager detaillierte Kenntnisse über die im Unternehmen verwendeten Lebenszyklusmodelle haben. Im V-Modell lässt sich beispielsweise der fundamentale Testprozess des ISTQB® auf der Stufe Systemtest folgendermaßen anpassen:

- Der Systemtest wird gleichzeitig mit dem Projekt geplant und die Teststeuerung dauert an, bis Testdurchführung und Abschluss der Testaktivitäten des Systemtests erfolgt sind.
- Systemtestanalyse und -entwurf des Systemtests finden parallel mit der Erstellung der Anforderungsspezifikation und (abstraktem) Architekturentwurf statt und (auf einer niedrigeren Ebene) gleichzeitig mit dem Komponentenentwurf.
- Aktivitäten in Bezug auf die Systemtestrealisierung (bspw. Bereitstellung der Testumgebung, des Testrahmens,) kann während des funktionalen Systementwurfs beginnen, obwohl der größte Aufwand normalerweise gleichzeitig mit der Programmierung und dem Komponententest anfällt. Die Arbeit an der Testrealisierung dauert dagegen oft bis wenige Tage vor Beginn der Testdurchführung. Die Testdurchführung beginnt, wenn alle Eingangskriterien für den Systemtest erfüllt sind (oder auf sie verzichtet wird), was normalerweise bedeutet, dass zumindest der Komponententest und oft auch der Komponentenintegrationstest abgeschlossen sind. Die Testdurchführung wird fortgesetzt, bis die Endekriterien des Systemtests erfüllt sind.
- Während der gesamten Testdurchführung werden die Endekriterien bewertet und die Testergebnisse berichtet, normalerweise mit größerer Häufigkeit und Dringlichkeit, je näher bestimmte Projekttermine rücken.
- Die Testabschlussaktivitäten folgen, wenn die Endekriterien des Systemtests erfüllt sind und die Testdurchführung als abgeschlossen erklärt wird. Manchmal können diese Aktivitäten jedoch verschoben werden, bis auch der Abnahmetest abgeschlossen ist und alle Projektaktivitäten beendet sind.

In einem iterativen oder inkrementellen Lebenszyklus müssen die gleichen Aufgaben durchgeführt werden, die zeitliche Abfolge und der Umfang können sich jedoch unterscheiden. Anstatt beispielsweise die gesamte Testumgebung zu Beginn des Projekts bereitstellen zu können, kann es effizienter sein, nur den für die aktuelle Iteration benötigten Teil zu implementieren. Für jedes der

iterativen oder inkrementellen Lebenszyklusmodelle gilt jedoch, dass je frühzeitiger vorausgeplant wird, desto weiter kann der Einfluss des fundamentalen Testprozesses reichen.

Außer den Planungsphasen, die in jedem Projekt stattfinden, können auch die Testdurchführung und die Berichterstattung vom eingesetzten Lebenszyklusmodell beeinflusst werden. In einem iterativen Lebenszyklus kann es beispielsweise effektiv sein, komplette Berichte zu erstellen und am Ende einer und vor Beginn der nächsten Iteration abschließende Review-Sitzungen zu halten. Wenn jede Iteration wie ein Miniprojekt behandelt wird, dann hat das Team Gelegenheit, auf die Ereignisse der vorangegangenen Iteration zu reagieren und Korrekturen/Anpassungen vorzunehmen. Da die Iterationen jedoch kurz sind und in einem engen Zeitrahmen ablaufen, macht es Sinn, Zeit und Aufwand für das Berichten und Bewerten zu kürzen. Trotzdem müssen die Aufgaben in einer Art und Weise erfolgen, dass der Gesamtfortschritt des Testens verfolgt und Problembereiche schnellstmöglich identifiziert werden können. Wenn es in einer Iteration Probleme mit dem Prozess gab, ist es leicht möglich, dass sich diese auf die nächste Iteration auswirken oder wieder auftauchen, falls keine Korrekturmaßnahmen ergriffen werden.

Allgemeine Informationen darüber, wie das Testen an die anderen Lebenszyklusaktivitäten angepasst wird, können in der Teststrategie beschrieben werden (siehe Abschnitt 2.4.2). Bei der Testplanung und/oder Projektplanung muss der Testmanager diese projektspezifische Anpassung für jede Teststufe und für jede ausgewählte Kombination von Softwarelebenszyklus und Testprozess vornehmen.

Je nach den Erfordernissen der Organisation, Projektkontext und Produkt können zusätzlich zu den im Foundation Level Syllabus definierten Teststufen noch weitere benötigt werden, beispielsweise:

- Hardware-Software-Integrationstest
- Systemintegrationstest
- Interaktionstest der Features
- Produktintegrationstest beim Kunden

Für jede Teststufe sollten die folgenden Merkmale klar definiert werden:

- Testziele mit erreichbaren Zielen
- Testumfang und Testelemente
- Testbasis und wie die Überdeckung dieser Testbasis gemessen wird (d.h. Rückverfolgbarkeit)
- Eingangs- und Endekriterien
- Testarbeitsergebnisse und Berichterstattung
- Testverfahren und wie durch diese Testverfahren die benötigte Überdeckung sicher erreicht wird
- Messungen und Metriken, die für Testziele, Eingangs- und Endekriterien und Berichten der Testergebnisse relevant sind (einschließlich der Überdeckungsmessung)
- Testwerkzeuge, die gegebenenfalls für bestimmte Aufgaben eingesetzt werden sollen
- Ressourcen (z.B. Testumgebungen)
- Zuständige Personen oder Gruppen, sowohl innerhalb als auch außerhalb des Teams
- Einhaltung von organisationsinternen, gesetzlichen und sonstigen Standards (falls zutreffend)

Wie später in diesem Kapitel erläutert, hat es sich bewährt, diese Elemente kohärent für sämtliche Teststufen zu definieren, um unnötige und riskante Lücken über mehrere Stufen mit ähnlichen Tests zu vermeiden.

2.2.4 Management von nicht-funktionalen Tests

Wenn nicht-funktionale Tests nicht eingeplant werden, kann es zu schwerwiegenden, manchmal sogar katastrophalen, Qualitätsproblemen kommen, die womöglich erst nach der Systemfreigabe entdeckt werden. Andererseits sind viele Arten nicht-funktionaler Tests mit erheblichen Kosten

verbunden. Es ist daher Aufgabe des Testmanagers, die notwendigen nicht-funktionalen Tests auszuwählen und dabei Kosten und Risiken abzuwägen. Außerdem gibt es viele unterschiedliche nicht-funktionale Tests und nicht alle sind unbedingt für eine bestimmte Anwendung geeignet.

Da der Testmanager eventuell nicht über ausreichend Erfahrung verfügt, um alle Faktoren bei der Planung zu berücksichtigen, sollte er einen Teil der Verantwortung in Zusammenhang mit der Testplanung an die technischen Test Analysten (bzw. in manchen Fällen an die Test Analysten) delegieren, die für die nicht-funktionalen Testaktivitäten eingeteilt sind. Diese sollten bei Planung und Durchführung der nicht-funktionalen Tests die folgenden Faktoren berücksichtigen:

- Anforderungen der Stakeholder
- benötigte Werkzeuge
- Testumgebung
- organisatorische Faktoren
- Sicherheit

Weitere detaillierte Informationen, siehe viertes Kapitel des Advanced Level Syllabus - Technical Test Analyst [ISTQB ATTA SYL].

Ein weiterer wichtiger Aspekt, den die Testmanager berücksichtigen müssen, ist, wie die nicht-funktionalen Tests in den Softwarelebenszyklus integriert werden. Ein Fehler, der häufig gemacht wird, ist zu warten, bis alle funktionalen Tests abschlossen sind, bevor mit den nicht-funktionalen Tests begonnen wird. Dies kann dazu führen, dass kritische nicht-funktionale Defekte erst spät aufgedeckt werden. Stattdessen sollten die Priorität und Reihenfolge der nicht-funktionalen Tests nach dem Risiko bestimmt werden. Es gibt häufig die Möglichkeit, nicht-funktionale Risiken bereits in frühen Teststufen oder sogar während der Entwicklung zu reduzieren. Beispielsweise kann ein Benutzbarkeits-Review eines Prototyps der Bedienerschnittstelle, das während des Systementwurfs erfolgt, sehr effektiv Benutzbarkeitsfehlerzustände identifizieren, die erhebliche Zeitprobleme verursachen würden, wenn sie erst gegen Ende des Systemtests aufgedeckt werden.

In iterativen Lebenszyklen kann es durch die schnell aufeinander folgenden Änderungen und Iterationen schwierig sein, sich auf bestimmte nicht-funktionale Tests zu konzentrieren, die anspruchsvolle Testrahmen benötigen. Daher sollten Testentwurf und Testrealisierung für Aktivitäten, die länger als eine Iteration benötigen, ausgegliedert und als separate Aktivitäten außerhalb der Iterationen organisiert werden.

2.2.5 Management des erfahrungsbasierten Tests

Obwohl der erfahrungsbasierte Test nützlich ist, sowohl beim Aufdecken von Fehlerzuständen, die mit anderen Testverfahren eventuell übersehen werden, als auch zur Kontrolle der Vollständigkeit dieser Verfahren, stellt er für das Testmanagement eine Herausforderung dar. Der Testmanager sollte sowohl über die Herausforderungen als auch über den Nutzen der erfahrungsbasierten Testentwurfsverfahren, insbesondere des explorativen Testens, Bescheid wissen. Es ist schwierig, bei dieser Art des Testens die Überdeckung zu bestimmen, da normalerweise nur sparsam protokolliert und die Tests nur minimal vorausgeplant werden. Das Management sollte außerdem besondere Aufmerksamkeit auf die Reproduzierbarkeit der Testergebnisse richten, besonders wenn mehrere Tester beteiligt sind.

Eine Möglichkeit, wie der erfahrungsbasierte Test, insbesondere das explorative Testen, gemanagt werden kann, ist die Aufteilung der Testaufgaben in kurze, 30 bis 120 Minuten lange Zeiträume, die manchmal auch als Testsitzungen bezeichnet werden. Diese Zeitfenster begrenzen und fokussieren die Aufgaben, die in einer Sitzung durchgeführt werden und ermöglichen eine gewisse Überwachung und Planung. Jede Sitzung konzentriert sich auf eine Test-Charta, die der Testmanager dem Tester schriftlich oder mündlich kommuniziert. In der Test-Charta sind die in der Testsitzung zu

überdeckenden Testbedingungen spezifiziert. Auch dies ist hilfreich für die Fokussierung und verhindert Überlappungen, wenn mehrere Tester gleichzeitig explorativ testen.

Eine weitere Möglichkeit erfahrungsbasierten Testens ist es, solche selbstgesteuerten und spontanen Testaktivitäten in traditionellere, vorgeplante Testsitzungen zu integrieren. So kann es den Testern beispielsweise ausdrücklich erlaubt (und entsprechend Zeit eingeplant) werden, in den vordefinierten Tests auch jenseits der ausdrücklich spezifizierten Schritte, Eingaben und vorausgesagten Ergebnissen neue Möglichkeiten und Themenbereiche zu erkunden. Auch können solche selbstgesteuerten Testsitzungen der Tester als Teil ihrer täglichen Testaufgaben eingeplant werden, die vor, während oder nach den täglichen vordefinierten Tests ausgeführt werden. Wenn in diesen Testsitzungen Fehlerzustände oder interessante Bereiche für zukünftige Tests gefunden werden, dann können die vordefinierten Tests entsprechend aktualisiert werden.

Zu Beginn der explorativen Testsitzung ermittelt der Tester die notwendigen Aufgaben zur Vorbereitung der Tests. Während der Testsitzung lernt der Tester die zu testende Anwendung kennen, entwirft und führt die Tests entsprechend dem eingesetzten Verfahren und den gewonnenen Erkenntnissen über die Applikation aus; er untersucht Fehlerzustände und zeichnet die Ergebnisse in einem Testprotokoll auf. (Falls die Tests wiederholbar sein müssen, sollten die Tester auch die Eingaben, Tätigkeiten und Ereignisse aufzeichnen.) Nach der Testsitzung kann eine Abschlussbesprechung stattfinden, die die Richtung für die nächsten Testsitzungen festlegt.

2.3 Risikoorientierter Test und andere Ansätze zur Testpriorisierung und Aufwandszuweisung

Eine universelle Herausforderung des Testmanagements besteht in der richtigen Auswahl, Zuweisung und Priorisierung von Tests. Dies bedeutet, dass das Testteam aus unendlich vielen Testbedingungen und Kombinationen von Bedingungen, die im Test abgedeckt werden könnten, eine endliche Anzahl von Bedingungen auswählen und den angemessenen Aufwand zuweisen muss, damit jede dieser Bedingungen durch Testfälle abgedeckt werden kann. Letztlich müssen die resultierenden Testfälle so priorisiert werden, dass die Testaufgaben möglichst effektiv und effizient erledigt werden können. Eine Identifizierung und Analyse der Risiken kann dem Testmanager, zusammen mit einigen anderen Faktoren, helfen dieses Problem zu lösen, obwohl zahlreiche wechselseitige Beschränkungen und Variablen möglicherweise eine Kompromisslösung notwendig machen.

2.3.1 Risikoorientiertes Testen

Risiko ist die Möglichkeit, dass es zu einem unerwünschten Ergebnis oder Ereignis kommt. Ein Risiko besteht grundsätzlich immer, wenn Probleme auftauchen, die die Produktqualität oder den Projekterfolg mindern könnten, wie sie von Kunden, Anwendern, Beteiligten oder Stakeholdern wahrgenommen werden. Wenn sich das potenzielle Problem in erster Linie auf die Produktqualität auswirkt, dann bezeichnet man derartige Probleme als Produktrisiken (oder Qualitätsrisiken). Wenn sich das potenzielle Problem dagegen vor allem auf den Projekterfolg auswirkt, dann bezeichnet man derartige Probleme als Projektrisiken (oder Planungsrisiken).

Beim risikoorientierten Testen werden Produktrisiken in einer Produktrisikoanalyse zusammen mit den Stakeholdern identifiziert und bewertet. Das Testteam entwirft, realisiert und führt die Tests dann aus, um die Qualitätsrisiken zu beherrschen. Die Qualität beinhaltet die Gesamtheit aller Features, Eigenschaften, Verhalten und Attribute, die sich auf die Zufriedenstellung von Kunden, Benutzern und Stakeholdern auswirken. Demzufolge ist ein Qualitätsrisiko nichts Anderes als der potenzielle Zustand, in dem im Produkt Qualitätsprobleme vorhanden sein könnten. Beispiele für Qualitätsrisiken sind falsche Berechnungen in Berichten (ein funktionales Risiko in Zusammenhang mit der Richtigkeit), langsame Reaktion auf Benutzereingaben (ein nicht-funktionales Risiko in

Zusammenhang mit Effizienz und Antwortzeiten) und schwer verständliche Bildschirme und Eingabefelder (ein nicht-funktionales Risiko in Zusammenhang mit Benutzbarkeit und Verständlichkeit). Wenn die Tester Fehlerzustände finden, reduzieren sie das Qualitätsrisiko, weil sie das Bewusstsein für vorhandene Fehlerzustände schärfen und die Möglichkeit schaffen, sie vor Release des Systems zu beheben. Auch wenn die Tester keine Fehlerzustände finden, reduzieren sie beim Testen das Risiko, denn sie stellen sicher, dass das System unter bestimmten (den getesteten) Bedingungen richtig funktioniert.

Beim risikoorientierten Testen dienen die Produktrisiken als Grundlage für die Auswahl der Testbedingungen, Zuweisung von Testaufwand zu den jeweiligen Testbedingungen und zur Priorisierung der erstellten Testfälle. Für den risikoorientierten Test gibt es eine Reihe von Verfahren, die sich sowohl in Art und Umfang der erstellten Dokumentation als auch im Grad der Formalität deutlich unterscheiden. Risikoorientiertes Testen hat immer implizit oder explizit das Ziel, durch das Testen das Produktrisiko insgesamt zu reduzieren, bzw. das Risiko auf ein akzeptables Maß zu reduzieren.

Risikoorientiertes Testen besteht aus vier Hauptaktivitäten:
- Risikoidentifizierung
- Risikobewertung
- Risikobeherrschung
- Risikomanagement

Diese Aktivitäten überlappen sich. In den nachfolgenden Abschnitten werden die einzelnen Aktivitäten beschrieben.

Die Risikoidentifizierung und -bewertung sind am effektivsten, wenn alle Stakeholder im Projekt daran beteiligt werden. In der Projektrealität werden Stakeholder manchmal durch Stellvertreter ersetzt. Bei der Entwicklung von Standardsoftware für den Massenmarkt kann es beispielsweise eine kleine Gruppe möglicher Kunden sein, die dabei hilft, die Fehlerzustände auszumachen, die ihre Nutzung der Software am meisten beeinträchtigen würden. Diese kleine Gruppe möglicher Kunden steht dann für den gesamten möglichen Kundenkreis. Wegen ihres speziellen Fachwissens sollten die Tester aktiv an Risikoidentifizierung und Risikoanalyse beteiligt werden.

2.3.1.1 Risikoidentifizierung

Die Stakeholder können sowohl Produkt- als auch Projektrisiken durch ein oder mehrere der folgenden Verfahren identifizieren:
- Experten-Interviews
- unabhängige Bewertungen
- Verwendung von Risikovorlagen
- Projektretrospektiven
- Risiko-Workshops
- Brainstorming
- Checklisten
- Erfahrungen aus der Vergangenheit

Je breiter die Basis der Stakeholder im Risikoidentifizierungs-Prozess ist, desto höher die Wahrscheinlichkeit, dass dabei die größtmögliche Menge an wichtigen Produktrisiken aufgedeckt wird.

Die Risikoidentifizierung erzeugt häufig Nebenprodukte; d.h. es werden Probleme identifiziert, die keine Produktrisiken sind. Beispiele dafür sind allgemeine Fragen oder Punkte zum Produkt oder Projekt, oder Probleme in Referenzdokumenten, z.B. Anforderungs- oder Entwurfsspezifikationen. Auch Projektrisiken werden häufig zwar als ein Nebenprodukt bei der Identifikation von Produktrisiken aufgedeckt, sind jedoch kein Schwerpunkt des risikoorientierten Testens. Dennoch ist das

Management von Projektrisiken für das Testen insgesamt und nicht nur beim risikoorientierten Testen, wichtig. Weitere Informationen sind in Abschnitt 2.4 enthalten.

2.3.1.2 Risikobewertung

Nach der Risikoidentifizierung kann die Risikobewertung erfolgen; hierbei werden die identifizierten Risiken genauer untersucht. Dabei werden die identifizierten Risiken kategorisiert und die Eintrittswahrscheinlichkeit und das Schadensausmaß für jedes Risiko bestimmt. Bei der Risikobewertung können auch weitere Eigenschaften für die einzelnen Risiken bewertet oder festgelegt werden, z.B. wer der Verantwortliche für das Risiko ist.

Risiken zu kategorisieren bedeutet, eine Einteilung in Risikotypen, wie z.B. Performanz, Zuverlässigkeit, Funktionalität usw. vorzunehmen. Manche Unternehmen verwenden zur Kategorisierung die Qualitätsmerkmale nach ISO Standard 9126 [ISO 9126] (wird gerade durch den ISO Standard 25000 [ISO 25000] abgelöst), viele arbeiten jedoch mit anderen Schemata. Häufig wird zur Kategorisierung auch dieselbe Checkliste verwendet wie bei der Risikoidentifizierung. Es existieren allgemeine Risiko-Checklisten; viele Unternehmen passen diese den eigenen Erfordernissen an. Bei einer Risikoidentifizierung auf Basis von Checklisten wird oft beim Identifizieren das Risiko einem bestimmten Risikotyp zugeordnet.

Für die Festlegung der Risikostufe werden für jedes einzelne Risiko die Eintrittswahrscheinlichkeit des Risikos und das Schadensausmaß eingeschätzt. Die Eintrittswahrscheinlichkeit des Risikos bezeichnet oft die Wahrscheinlichkeit, dass das potenzielle Problem im getesteten System vorhanden ist. Bei der Eintrittswahrscheinlichkeit geht es also um die Bewertung eines technischen Risikos. Folgende Faktoren beeinflussen die Eintrittswahrscheinlichkeit von Produkt- und Projektrisiken:

- Komplexität der Technologie und Zusammensetzung des Teams
- Ausbildung und Erfahrungen der Testbeteiligten (z.B. Mitarbeiter von Fachbereichen, Systementwickler, Programmierer)
- Konflikte im Team
- vertragliche Probleme mit Zulieferern
- räumlich verteilte Teams
- Altsysteme versus Neuentwicklung
- Werkzeuge und Technologie
- schwaches Management oder technische Leitung
- Zeitdruck, knappe Ressourcen, begrenztes Budget und Druck durch das Management
- Fehlen bisheriger Qualitätssicherungsmaßnahmen
- häufige Änderungen
- hohe bisherige Fehlerraten
- Schnittstellen-/Integrationsproblematik

Das Schadensausmaß bezeichnet oft das Ausmaß der Wirkung auf Anwender, Kunden oder andere Stakeholder. Folgende Faktoren beeinflussen das Schadensausmaß von Projekt- und Produktrisiken:

- Häufigkeit der Nutzung der betroffenen Funktion
- Kritikalität der Funktion für die Erreichung eines Geschäftsziels
- Schaden für die Reputation
- entgangene Geschäfte
- mögliche finanzielle, ökologische oder soziale Verluste oder Haftungsansprüche
- zivil- oder strafrechtliche Maßnahmen
- Aberkennung der Lizenz
- fehlende vernünftige Lösungsalternativen
- das negative Erscheinungsbild, wenn Fehlerwirkungen bekannt werden, bzw. Negativschlagzeilen
- Sicherheit

Die Risikostufe lässt sich entweder quantitativ oder qualitativ betrachten. Wenn Eintrittswahrscheinlichkeit und Schadensausmaß des Risikos quantitativ bestimmt werden können, ergibt ein einfaches Multiplizieren der beiden Werte die Kosten einer Gefährdung, also den erwarteten Verlust bei einem bestimmten Risiko.

In der Regel lässt sich die Risikostufe aber nur qualitativ bestimmen. Dabei kann die Eintrittswahrscheinlichkeit zwar mit sehr hoch, hoch, mittel, gering oder sehr gering angegeben, aber nicht als ein genauer Prozentsatz spezifiziert werden. Ähnlich verhält es sich beim Schadensausmaß: Auch dieses kann mit sehr hoch, hoch, mittel, gering oder sehr gering angegeben, aber nicht umfassend und präzise in finanzieller Hinsicht spezifiziert werden. Wenn nämlich quantitative Methoden unangemessen verwendet werden, täuscht das die Stakeholder in ihrer Einschätzung darin, wieweit sie das Risiko wirklich begriffen und im Griff haben. Wenn das quantitative Vorgehen zur Bestimmung der Risikostufe nicht fachgerecht eingesetzt wird, könnte es die Stakeholder sogar darüber täuschen, inwieweit sich Risiko tatsächlich verstehen und managen lässt. Qualitative Vorgehen zur Bestimmung der Risikostufe werden oft untereinander kombiniert, mittels Multiplikation oder Addition, um ein Gesamtergebnis zu erhalten. Dieses Ergebnis kann als eine Risikoprioritäts-Kennzahl gelten, sollte jedoch als eine qualitative, relative Bewertung auf einer Ordinalskala interpretiert werden.

Wenn die Risikoanalyse nicht auf umfangreichen und statistisch korrekten Risikodaten beruht, dann wird sie, ob qualitativ oder quantitativ, immer auf der jeweiligen Wahrnehmung von Eintrittswahrscheinlichkeit und Schadensausmaß basieren. Naturgemäß haben Projektmanager, Programmierer, Anwender, Fachanalytiker, Systemarchitekten und Tester unterschiedliche Wahrnehmungen und deshalb vielleicht auch ganz andere Ansichten über die Risikostufe des jeweiligen Risikos. Die Risikoanalyse sollte daher Wege finden, wie ein Konsens zu erzielen oder – im ungünstigsten Fall – die aggregierte Risikostufe vorzugeben ist (z.B. Festlegung durch das Management, Bestimmung des Durchschnitts-, Median- oder Modalwerts für das einzelne Risiko). Außerdem sollte geprüft werden, ob die Risikostufen gut über die gesamte Skala verteilt sind, damit sie Aussagekraft für weitere Aktivitäten zur Risikobeherrschung haben (z.B. für Ablaufplanung, Priorisierung von Tests und Zuweisung von Aufwand). Nur dann können die Risikostufen eine Richtschnur für Aktivitäten zur Risikobeherrschung sein.

2.3.1.3 Risikobeherrschung

Risikoorientiertes Testen beginnt mit einer Qualitätsrisikoanalyse, bei der Produkt- oder Qualitätsrisiken identifiziert und bewertet werden. Dies ist Grundlage des Mastertestkonzepts und anderer Testkonzepte. Entsprechend der Festlegungen dieser Testdokumente erfolgen Testentwurf, -realisierung und -durchführung. Um die identifizierten Risiken abzudecken. Der Aufwand für Erstellung und Ausführung eines Tests ist proportional zur Risikostufe. Dies bedeutet, dass für die höheren Risiken besonders gründliche Testverfahren (z.B. paarweises Testen) eingesetzt werden und für geringere Risiken weniger akribische Testverfahren (z.B. Äquivalenzklassenbildung oder exploratives Testen mit zeitlicher Eingrenzung). Die Risikostufe für das jeweilige Risiko entscheidet außerdem über die Priorität bei Entwicklung und Testausführung. An der Sicherheit orientierte Standards, wie FAA DO-178B/ED 12B oder IEC 61508, schreiben Testverfahren und Überdeckung je nach Risikostufe vor. Außerdem sollte die Risikostufe gewisse Entscheidungen beeinflussen. Hierzu gehören beispielsweise Entscheidungen über den Einsatz von Reviews der Arbeitsergebnisse des Projekts (einschließlich des Tests), über den Unabhängigkeitsgrad der Testorganisation, über die Erfahrung der Tester und über den Umfang von Fehlernachtests sowie Regressionstests

Im Projektverlauf sollte das Testteam bei Vorliegen zusätzlicher Informationen beachten, dass sich aufgrund dieser die Produktrisiken und/oder die Risikostufe bekannter Risiken ändern können. Es ist also eine regelmäßige Anpassung der Qualitätsrisikoanalyse und damit des Testens, erforderlich. Diese Aktualisierung sollte zumindest zu jedem Meilenstein stattfinden. Dies schließt die Identifikation neuer Risiken, erneutes Bewerten der Risikostufen für bestehende Risiken und die Bewertung ein, ob Risikobeherrschungsmaßnahmen effektiv waren. Hierzu ein Beispiel: Wurden während der

Anforderungsanalyse die Risikoidentifizierung und -bewertung auf Basis der Anforderungsspezifikation durchgeführt, dann sollten die Risiken nach Vorliegen des Systementwurfs neu bewertet werden. Ein weiteres Beispiel: Falls beim Testen weit mehr Fehlerzustände in einer Komponente gefunden wurden als erwartet, dann kann man annehmen, dass die Fehlerwahrscheinlichkeit in diesem Bereich höher ist als angenommen und die Eintrittswahrscheinlichkeit und Risikostufe nach oben korrigieren. Für diese Komponente könnte das zu einem erhöhten Testaufwand führen.

Produkt- oder Qualitätsrisiken lassen sich auch vor Beginn der Testausführung steuern. Werden beispielsweise bei der Risikoidentifizierung Probleme mit den Anforderungen festgestellt, dann sind gründliche Reviews der Anforderungsspezifikation sicherlich ein probates Mittel zur Risikobeherrschung. Auch hierdurch lässt sich das Risiko verringern, was bedeuten kann, dass weniger Tests zur Beherrschung des Restrisikos erforderlich sind.

2.3.1.4 Risikomanagement im Softwarelebenszyklus

Im Idealfall begleitet das Risikomanagement den gesamten Softwarelebenszyklus. Wenn eine Organisation eine Testrichtlinie und/oder eine Teststrategie hat, dann sollten sie den grundlegenden Prozess beschreiben, nach dem Produkt- und Projektrisiken im Testen behandelt werden. Sie sollten zeigen, dass Risikomanagement ein integraler Bestandteil aller Teststufen ist und wie es sie beeinflusst.

In einer reifen Organisation, wo viel Risikobewusstsein im Projektteam vorhanden ist, findet das Risikomanagement auf vielen Ebenen statt, nicht nur beim Testen. Wichtige Risiken werden nicht nur in der jeweiligen Teststufe frühzeitig behandelt, sondern auch in früheren Teststufen. (Beispiel: Wenn die Performanz als wichtiges Qualitätsrisiko identifiziert wurde, dann beginnen die Performanztests nicht nur frühzeitig im Systemtest, sondern Performanzetests werden bereits in den Teststufen Komponenten- und Integrationstest durchgeführt. Reife Organisationen identifizieren nicht nur Risiken, sondern auch die Risikoquellen und die Konsequenzen der Risiken, sollten diese dann vorkommen. Fehlerzustände, die vorkommen, werden mithilfe einer Grundursachenanalyse genauer betrachtet, um Prozessverbesserungen vorzunehmen, die diese Fehlerzustände von vornherein verhindern. Die Risikobeherrschung begleitet den gesamten Softwarelebenszyklus. Die Risikoanalyse ist stets gut mit Informationen versorgt und bezieht die damit zusammenhängende Aufgaben, wie die Analyse des Systemverhaltens, die kostenbasierte Risikobewertung, die Produktrisikoanalyse, die Endnutzer-Risikoanalyse und die Analyse von Haftungsrisiken mit ein. Die Risikoanalyse geht über das Testen hinaus, da das Testteam damit an der übergeordneten Risikoanalyse beteiligt ist und Einfluss nehmen kann.

Die meisten risikoorientierten Methoden beinhalten auch Verfahren, die die Risikostufe zur Priorisierung und Ablaufplanung des Tests verwenden. Dadurch wird sichergestellt, dass die wichtigsten Bereiche frühzeitig abgedeckt und die wichtigsten Fehlerzustände bei der Testausführung aufgedeckt werden. Manchmal werden alle hohen Risikostufen vor den niedrigeren Risikostufen getestet und die Tests erfolgen streng nach der Reihenfolge der Risikos (Depth-first, d.h. Testen in die Tiefe). In anderen Fällen machen die Tester Stichproben von identifizierten Risiken aller Risikostufen, gewichten die Risiken und wählen Tests aus, wobei sie dafür sorgen, dass jedes Risiko mindestens einmal abgedeckt wird (Breadth-first, d.h. Testen in die Breite).

Unabhängig davon, ob beim risikoorientierten Testen in die Tiefe oder in die Breite getestet wird, ist es möglich, dass die Zeit für das Testen abläuft, ohne dass alle Tests durchgeführt wurden. Beim risikoorientierten Testen können die Tester in solchen Fällen das Management über das verbleibende Restrisiko informieren und das Management kann entscheiden, ob weiter getestet wird, oder ob das Restrisiko an Anwender, Kunden, Helpdesks, technischen Support und/oder Betriebspersonal weitergegeben wird.

Die am meisten fortgeschrittenen und risikoorientierten Testverfahren — müssen nicht zwangsläufig sehr formal oder „schwer“ sein - ermöglichen den Projektmitarbeitern, Projekt- und Produktmanagern, Führungskräften und dem gehobenen Management, sowie den anderen Stakeholdern den Softwarelebenszyklus auch während der Testausführung zu überwachen und zu steuern und auf Basis des verbleibenden Restrisikos Freigabeentscheidungen zu treffen. Dies bedingt, dass der Testmanager die Testergebnisse hinsichtlich des Risikos so kommuniziert, dass sie für jeden Stakeholder verständlich sind.

2.3.2 Risikoorientierte Testverfahren

Es gibt mehrere risikoorientierte Testverfahren. Einige sind sehr informell, z.B. wenn Tester Qualitätsrisiken beim explorativen Testen analysieren [Whittaker09]. Das kann eine Richtschnur für das Testen sein, kann aber auch zu einer übermäßigen Fokussierung auf die Wahrscheinlichkeit von Fehlerzuständen führen und weniger auf deren Auswirkungen. Außerdem ist dabei kein Input der unterschiedlichen Stakeholder vorhanden. Hinzu kommt, dass diese Ansätze subjektiv sind und von den Fähigkeiten, Erfahrung und Vorlieben einzelner Tester abhängen. Somit erzielen sie auch selten den größtmöglichen Nutzen des risikoorientierten Testens.

Um vom Nutzen des risikoorientierten Testens zu profitieren und dabei die Kosten gering zu halten, greifen viele Praktiker zu „leichten“ risikoorientierten Testvorgehensweisen. Dabei vermischt sich die Reaktionsfähigkeit und Flexibilität informeller Vorgehensweisen mit der Leistungsfähigkeit und Konsensbildung formalerer Vorgehensweisen. Zu den „leichten“ Vorgehensweisen gehören: Pragmatische Risikoanalyse und Management (PRAM) [Black09], Systematic Software Testing (SST) [Craig02], sowie Produkt-Risiko-Management (PRisMa) [vanVeenendaal12]. Zusätzlich zu den allgemein üblichen Eigenschaften des risikoorientierten Testens haben diese Vorgehensweisen folgende Merkmale:

- Sie wurden im Laufe der Zeit aus den in der Praxis gemachten Erfahrungen in Zusammenhang mit dem risikoorientierten Testen entwickelt, insbesondere in Industrien, in denen Effizienz besonders wichtig ist.
- Sie stützen sich auf die umfassende Beteiligung eines funktionsübergreifenden Teams von Stakeholdern, die bei der anfänglichen Risikoidentifizierung und -bewertung sowohl geschäftliche als auch technische Sichtweisen einbringen.
- Vorzugsweise werden sie in den ganz frühen Projektphasen schon eingesetzt, wo es noch viele Möglichkeiten zur Beherrschung von Qualitätsrisiken gibt und wo die Arbeitsergebnisse der Risikoanalyse (sowohl das Haupt-, als auch die Nebenergebnisse) noch die Spezifikation und die Implementierung des Produkts risikominimierend beeinflussen können.
- Sie verwenden die erstellten Arbeitsergebnisse (Risikomatrix oder Risikoanalysetabelle) als Basis für Testkonzept und Testbedingungen und somit für alle nachfolgenden Testmanagement- und Analyseaktivitäten.
- Sie unterstützen das Berichten der Testergebnisse, insbesondere bezüglich der Restrisiken, an alle anderen Stakeholder des Tests.

Einige der Verfahren (z.B. SST) benötigen die Anforderungsspezifikation als Eingabe für die Risikoanalyse und können ohne Anforderungsspezifikation nicht angewandt werden. Andere Verfahren (z.B. PRisMa und PRAM) fördern eine Mischung aus risikoorientierter und anforderungsbasierter Strategie und verwenden die Anforderungs- und/oder sonstige Spezifikationen als Eingabe für die Risikoanalyse, können jedoch auch funktionieren, wenn sie komplett auf den Eingaben der Stakeholder basieren. Durch die Verwendung von Anforderungen als Eingaben kann eine gute Abdeckung der Anforderungen sichergestellt werden, aber der Testmanager muss dafür sorgen, dass wichtige Risiken, die in den Anforderungen nicht enthalten sind (besonders in nicht-funktionalen Bereichen), nicht übersehen werden. Wo gute, priorisierte Anforderungen als Input dienen, ist in der Regel ein starker Zusammenhang zwischen Risikostufen und Anforderungspriorität gegeben.

Bei vielen der Verfahren wird der Risikoidentifizierungs- und Bewertungsprozess als ein Mittel eingesetzt, um einen Konsens der Stakeholder hinsichtlich der Testvorgehensweise herbeizuführen. Dies ist ein bedeutender Nutzen; allerdings müssen die Stakeholder dafür Zeit investieren und entweder an Brainstorming-Runden oder an Einzelinterviews teilnehmen. Eine unzureichende Beteiligung der Stakeholder führt zu einer lückenhaften Risikoanalyse. Natürlich sind sich die Stakeholder nicht immer über die Risikostufe einig. Daher muss die Person, die die Risikoanalyse leitet, kreativ und konstruktiv mit den Stakeholdern daran arbeiten, um das bestmögliche Maß an Übereinstimmung herbeizuführen. Alle Fähigkeiten, die ein geschulter Moderator zur Leitung von Review-Sitzungen benötigt, gelten auch für die Person, die eine Qualitätsrisikoanalyse leitet.

Genau wie die formalen Verfahren ermöglichen auch die „leichten" Verfahren eine Gewichtung von Eintrittswahrscheinlichkeit und Schadensausmaß des Risikos und damit der geschäftlichen oder technischen Risiken (je nach Teststufe). Im Gegensatz zu den formalen Verfahren verwenden die "leichten" Verfahren jedoch:

- nur die beiden Faktoren Eintrittswahrscheinlichkeit und Schadensausmaß und
- einfache, qualitative Beurteilungen und Skalen.

Die „Leichtigkeit" dieser Verfahren verleiht ihnen Flexibilität, Anwendbarkeit in unterschiedlichen Bereichen und Zugänglichkeit für Teams mit unterschiedlicher Erfahrung und Qualifikation (selbst für Nachwuchskräfte ohne technischen Hintergrund).

Am anderen Ende der Skala, bei den formalen Verfahren, hat der Testmanager mehrere Optionen:

- Die Gefährdungsanalyse, bei der der Analyseprozess fortgesetzt und vertieft wird. Die Gefährdungsanalyse versucht die Gefährlichkeit jedes einzelnen Risikos zu identifizieren.
- Kosten einer Gefährdung. Hier bestimmt die Risikobewertung für jedes Qualitätsrisiko drei Faktoren: 1) die Eintrittswahrscheinlichkeit (in Prozent) einer Fehlerwirkung in Zusammenhang mit dem Risiko; 2) die Kosten des Verlusts (eine finanzielle Größe) verbunden mit einer typischen Fehlerwirkung in Zusammenhang mit dem Risiko, falls diese in der Produktion bzw. im Regelbetrieb auftritt; und 3) die Kosten des Testens auf solche Fehlerwirkungen.
- Die Fehlermöglichkeits- und Einfluss-Analyse (FMEA, Failure Mode and Effect Analysis) sowie deren Varianten [Stamatis03]. Dabei werden Qualitätsrisiken, deren potenzielle Ursachen und deren wahrscheinliche Auswirkungen identifiziert und sodann ein Rating der Fehlerschweregrads, Priorität und Fehlerfindung vorgenommen.
- Quality Function Deployment (QFD). Dabei handelt es sich um ein Risikomanagementverfahren mit Implikationen für das Testen, insbesondere weil sich das Verfahren mit Qualitätsrisiken befasst, die aus einem falschen oder mangelnden Verständnis der Kunden- oder Benutzeranforderungen resultieren.
- Fehlerbaum-Analyse (FBA). Hier erfolgt eine Ursachenanalyse von verschiedenen, tatsächlich (im Test oder in der Produktion bzw. im Regelbetrieb) beobachteten Fehlerwirkungen, oder von potenziellen Fehlerwirkungen (Qualitätsrisiken). Die Analyse beginnt mit Fehlerzuständen, die die Fehlerwirkung verursachen könnten und befasst sich dann mit Fehlhandlungen oder Fehlerzuständen, die jene Defekte verursachen könnten und wird fortgesetzt bis die verschiedenen Grundursachen identifiziert sind.

Die spezifischen Verfahren, die im risikoorientierten Testen eingesetzt werden sollten, sowie der Grad an Formalität der einzelnen Verfahren, hängen vom Projekt, Prozess und Produktüberlappungen ab. Informelle Ansätze, wie z.B. Whittaker's exploratives Verfahren, können für einen Patch oder „Hot-Fix" Anwendung finden. Bei agilen Projekten ist die Qualitätsrisikoanalyse voll in die frühe Phase eines jeden Sprints integriert und die Dokumentation der Risiken vermischt sich mit der Verfolgung der User-Stories. Systeme von Systemen benötigen eine Risikoanalyse der einzelnen Systeme sowie des gesamten Systems an Systemen. Sicherheitskritische Projekte benötigen generell mehr Formalität und Dokumentation.

Eingaben, Prozesse und Ausgaben des risikoorientierten Testens werden durch das gewählte Verfahren bestimmt. Zu den gängigen Eingaben gehören Erkenntnisse von Stakeholdern, Spezifikationen und historische Daten. Zu den gängigen Prozessen gehören Risikoidentifizierung, Risikobewertung und Risikosteuerung. Gängige Ausgaben sind eine Liste der Qualitätsrisiken (mit den jeweiligen Risikostufen und dem empfohlenen Testaufwand), Fehlerzustände, die in den Eingabedokumenten (z.B. Spezifikationen) aufgedeckt wurden, Fragen, Probleme oder offene Punkte bezüglich der Risiken, sowie Projektrisiken, die das Testen oder das Projekt insgesamt betreffen.

Der kritischste Erfolgsfaktor beim risikoorientierten Testen ist gewöhnlich das Einbinden der richtigen Stakeholder bei Risikoidentifizierung und -bewertung. Sämtliche Stakeholder haben ihre eigene Auffassung von dem, was die Produktqualität letztlich ausmacht, jeder hat eigene Prioritäten und Interessen in Zusammenhang mit der Qualität. Letztlich lassen sich Stakeholder jedoch in zwei Kategorien unterteilen: Stakeholder des Fachbereichs und Stakeholder des technischen Bereichs.

Stakeholder des Fachbereichs sind unter anderem Kunden, Anwender, Betriebspersonal, Mitarbeiter von Help-Desk und technischem Support. Diese Stakeholder verstehen die Kunden und Anwender und sie können daher aus Kundensicht Risiken identifizieren und das Schadensausmaß bewerten.

Stakeholder des technischen Bereichs sind unter anderem Entwickler, Systemarchitekten, Datenbank- und Netzwerkadministratoren. Diese Stakeholder haben ein grundlegendes Verständnis dafür, wie Software versagen kann und sie können daher aus technischer Perspektive Risiken identifizieren und das Schadensausmaß bewerten.

Bei manchen Stakeholdern kommen beide Perspektiven zusammen. Beispiele sind Experten des Fachbereichs, die im Testen tätig sind, oder Business Analysten, die aufgrund ihrer technischen und geschäftlichen Expertise oft eine breitere Sichtweise der Risiken haben.

Der Risikoidentifizierungsprozess erzeugt eine beträchtliche Liste von Risiken. Es ist unnötig, wenn sich die Stakeholder über die einzelnen Risiken streiten, denn solange auch nur ein Stakeholder etwas als ein Risiko wahrnimmt, ist es ein Risiko. Es ist allerdings wichtig, dass die Stakeholder einen Konsens bezüglich des Ratings für die Risikostufe erzielen. So muss man z.B. bei den "leichten" Methoden, die ja Eintrittswahrscheinlichkeit und Schadensausmaß als Ratingfaktoren verwenden, im Rahmen des Vorgehens erst einmal gemeinsam ein Ratingschema für eben diese Beiden finden. Alle Stakeholder, einschließlich der Testgruppe, müssen mit derselben Skala arbeiten und müssen sich für jedes Qualitätsrisiko einem einzigen Rating für Eintrittswahrscheinlichkeit und Schadensausmaß annähern.

Wenn risikoorientiertes Testen langfristig eingesetzt werden soll, dann muss der Testmanager die Stakeholder davon aktiv überzeugen und es mit ihnen zusammen erfolgreich einführen. Die funktionsübergreifende Gruppe muss den Wert der Risikoanalyse erkennen, damit das Verfahren nachhaltig eingesetzt wird. Dies setzt voraus, dass der Testmanager Verständnis hat für die Bedürfnisse, Erwartungen und die verfügbare Zeit der Stakeholder, um am Prozess teilzunehmen.

Großes Engagement der Stakeholder bei der Qualitätsrisikoanalyse bringt dem Testmanager einen wichtigen Nutzen, vor allem bei unzureichend spezifizierten Projekten mit schwachen bzw. fehlenden Anforderungen. Hier können die Stakeholder die Risiken immer noch mit Hilfe einer geeigneten Checkliste Risiken identifizieren. Der Nutzen wird offensichtlich, wenn sich nach Umsetzung des risikoorientierten Testens die Effektivität des Testteams bei der Fehlerfindung verbessert. Der Grund dafür ist, dass die verwendete Testbasis – in diesem Fall die Liste mit den Qualitätsrisiken – vollständiger ist.

Beim Abschluss der Testaktivitäten sollte das Team messen, inwieweit das risikoorientierte Testen den erwarteten Nutzen gebracht hat. In vielen Fällen bedeutet dies, dass einige oder alle vier der nachfolgenden Fragen durch Metriken und Gespräche im Testteam zu beantworten sind:

- Ist der prozentuale Anteil an wichtigen Fehlerzuständen, die das Testteam aufgedeckt hat, höher als der prozentuale Anteil an weniger wichtigen Fehlerzuständen?
- Wurden die meisten der wichtigen Fehlerzustände in einer frühen Phase der Testausführung gefunden?
- War das Testteam in der Lage, den Stakeholdern die Testergebnisse mit Bezug auf Risiken zu erklären?
- Falls das Testteam Tests ausgelassen hat, hatten diese eine geringere Risikostufe als die ausgeführten Tests?

In den meisten Fällen werden, wenn das risikoorientierte Testen erfolgreich war, alle vier Fragen bejaht. Langfristig sollte der Testmanager Prozessverbesserungsziele für diese Metriken festsetzen und danach streben, die Effizienz des Risikoanalyseprozesses zu steigern. Selbstverständlich können auch andere Metriken und Erfolgskriterien beim risikoorientierten Testen zur Anwendung kommen. Der Testmanager sollte jedoch die Zusammenhänge zwischen diesen Metriken, die strategischen Ziele des Testteams und das Verhalten, das aus einem Einsatz bestimmter Metriken und Erfolgskriterien resultiert, reiflich überlegen.

2.3.3 Weitere Verfahren zur Testauswahl

Obwohl viele Testmanager risikoorientiertes Testen als ein Element ihrer Teststrategie einsetzen, gibt es viele, die andere Verfahren einbeziehen.

Eines der wichtigsten alternativen Verfahren für die Entwicklung und Priorisierung von Testbedingungen ist das anforderungsbasierte Testen. Beim anforderungsbasierten Testen können mehrere Verfahren eingesetzt werden, wie z.B. Review auf Zweideutigkeit, Testbedingungsanalyse oder Ursache-Wirkungs-Graph-Analyse. Reviews auf Zweideutigkeit identifizieren Zweideutigkeiten und eliminieren Unklarheiten in den Anforderungen (die als Testbasis dienen). Oft wird dabei eine Checkliste mit häufigen Anforderungsfehlern verwendet (siehe [Wiegers03]).

Wie in [Craig02] beschrieben, geht es bei der Testbedingungsanalyse um ein genaues Lesen der Anforderungsspezifikation, um die abzudeckenden Testbedingungen zu identifizieren. Wenn diese Testbedingungen bereits priorisiert sind, dann können diese verwendet werden, um Aufwand zuzuweisen und die Testfälle zu priorisieren. Wenn keine Priorisierung vorgegeben ist, dann ist es schwierig, den angemessenen Aufwand und Reihenfolge des Testens zu bestimmen, ohne den anforderungsbasierten Test mit risikoorientiertem Testen zu kombinieren.

Die Ursache-Wirkungs-Graph-Analyse wird im Advanced Level Syllabus für Test Analysten in Zusammenhang mit der Überdeckung von Kombinationen von Testbedingungen als Teil der Testanalyse behandelt. Allerdings hat dieses Verfahren breitere Einsatzmöglichkeiten. So kann damit ein extrem schwieriges Testproblem auf eine überschaubare Anzahl von Testfällen reduziert werden bei 100% funktionaler Überdeckung der Testbasis. Die Ursache-Wirkungs-Graph-Analyse identifiziert außerdem beim Testfallentwurf Lücken in der Testbasis. Dadurch können Fehlerzustände früh im Softwarelebenszyklus aufgedeckt werden, wenn der Testentwurf auf Basis der groben Anforderungen beginnt. Ein wichtiger Hinderungsgrund in Zusammenhang mit dem Einsatz der Ursache-Wirkungs-Graph-Analyse ist, dass die Erstellung der Graphen eine komplexe Aufgabe ist. Da die manuelle Durchführung kompliziert sein kann, gibt es allerdings Werkzeuge, die diese Methode unterstützen.

Ein allgemeines Hindernis beim anforderungsbasierten Testen ist, dass Anforderungsspezifikationen häufig zweideutig, nicht testbar, unvollständig oder gar nicht vorhanden sind. Nicht alle Organisationen sind motiviert und willens, diese Probleme zu lösen. Tester, die mit einer solchen Situation konfrontiert sind, müssen zwangsläufig eine andere Teststrategie wählen.

Eine weitere Methode, die manchmal zusätzlich zur Verwendung der vorhandenen Anforderungen eingesetzt wird, ist die Erstellung von Nutzungs- oder Anwendungsprofilen. Hierbei handelt es sich um

eine modellorientierte Strategie, bei der eine Mischung aus Anwendungsfällen, Anwendern (manchmal als Persona bezeichnet), Eingaben und Ausgaben verwendet wird, um die tatsächliche Nutzung des Systems wirklichkeitsgetreu abzubilden. Dies ermöglicht nicht nur das Testen der Funktionalität, sondern auch anderer Qualitätsmerkmale wie Benutzbarkeit, Interoperabilität, Zuverlässigkeit, Sicherheit und Performanz.

Bei Testanalyse und Planung identifiziert das Testteam die Nutzungsprofile und versucht, diese mit Testfällen abzudecken. Bei einem Nutzungsprofil handelt es sich um eine Schätzung, die auf den verfügbaren Informationen über die realistische Nutzung der Software basiert. Wie schon beim risikoorientierten Testen bedeutet dies auch hier, dass die Nutzungsprofile die zukünftige Realität eventuell nicht perfekt abbilden. Wenn jedoch ausreichende Informationen und Angaben von Stakeholdern verfügbar sind, dann ist das erstellte Modell angemessen (siehe [Musa04]).

Manche Testmanager verwenden auch methodische Vorgehensweisen, wie z.B. Checklisten, um zu bestimmen, was getestet wird, wie viel getestet wird und in welcher Reihenfolge getestet wird. Für sehr stabile Produkte, kann eine Checkliste der wichtigsten zu testenden funktionalen und nicht-funktionalen Bereiche, die zu testen sind, in Kombination mit einem Repository bestehender Testfälle ausreichend sein. Die Checkliste liefert Heuristiken für die Zuweisung von Aufwand und für die Ablaufplanung der Tests, meist basierend auf Art und Anzahl der angefallenen Änderungen. Solche Vorgehensweisen verlieren jedoch an Aussagekraft, wenn mehr zu testen ist als ein paar geringfügige Änderungen.

Schließlich wird häufig auch noch die Methode des reaktiven Ansatzes angewandt (reaktiver Test). Bei einem reaktiven Ansatz wird vor der Testdurchführung nur wenig Testanalyse, Testentwurf oder Testrealisierung betrieben. Das beobachtete Verhalten des gelieferten Produkts und geeignete Tests als Reaktion darauf stehen im Fokus des Testteams. Fehlerschwerpunkte, sobald entdeckt, werden zum Schwerpunkt weiterer Tests. Priorisierung und Zuweisung sind komplett dynamisch. Der reaktive Ansatz kann als Ergänzung zu anderen Vorgehensweisen funktionieren. Ausschließlich eingesetzt führt reaktiver Test tendenziell dazu, dass wichtige Bereiche der Anwendung, so sie wenig Fehlerzustände haben, gerne übersehen werden.

2.3.4 Testpriorisierung und Aufwandszuweisung im Testprozess

Egal welches Verfahren – bzw. welche Kombination von Verfahren – Testmanager wählen, sie müssen diese Verfahren in die Projekt- und Testprozesse integrieren. Bei sequenziellen Lebenszyklen (z.B. V-Modell) wählt das Testteam in der Anforderungsphase die Tests aus, bestimmt den jeweiligen Aufwand und bestimmt eine erste Priorisierung. Im weiteren Verlauf folgen regelmäßige Anpassungen. Bei iterativen oder agilen Lebenszyklen dagegen basiert die Vorgehensweise auf aufeinanderfolgenden Iterationen. Testplanung und Teststeuerung müssen berücksichtigen, wie sich die Risiken, Anforderungen und/oder Nutzungsprofile entwickeln werden und entsprechend reagieren.

Bei Testanalyse, Testentwurf und Testrealisierung müssen Aufwandszuweisung und Priorisierung, die in der Testplanung bestimmt wurden, angewendet werden. Häufig kann der Testprozess gestört werden, wenn eine sorgfältige Analyse und/oder Modellierung stattfindet, ohne dass diese Informationen in die Gestaltung des Testprozesses einfließen. Zu solchen Missständen kommt es normalerweise während Testentwurf und -realisierung.

Bei der Testausführung sollten die Tests entsprechend der Priorisierung, die in der Testplanung erfolgte, durchgeführt werden. Es ist allerdings wichtig, dass die Priorisierung regelmäßig entsprechend der Erkenntnisse, die erst nach Erstellung des ursprünglichen Plans gewonnen wurden, aktualisiert wird. Wenn Testmanager Testergebnisse und Status von Endekriterien bewerten und berichten, dann müssen sie dies auch hinsichtlich der Risiken, Anforderungen, Nutzungsprofile, Checklisten und der sonstigen Hilfsmitteln tun, die für Auswahl und Priorisierung der Tests verwendet wurden. Bei Bedarf sollte eine Triage-Sitzung auf Grundlage des Testpriorisierungsschemas erfolgen.

Als Teil des Testergebnisberichts und der Auswertung der Endekriterien wird so mancher Testmanager erheben wollen, zu welchem Grad der Test bereits abgeschlossen ist. Dazu gehört auch, dass Testfälle und entdeckte Fehlerzustände zur relevanten Testbasis zurückverfolgt werden. Bei risikoorientiertem Testen, können Tester beispielsweise, sobald die Tests gelaufen und die Fehlerzustände gefunden sind, das verbliebene Restrisiko bestimmen. So kann man das risikoorientiertes Testen nutzen, um den richtigen Zeitpunkt für die Freigabe zu bestimmen. Die Testberichte sollten über die abgedeckten und die noch offenen Risiken genauso wie über den erreichten und den noch nicht erreichten Nutzen informieren.

Beim Abschluss der Testaktivitäten sollten Testmanager Metriken und Erfolgskriterien bewerten, die die Stakeholder des Testens benötigen und erwarten, einschließlich der Erfordernisse und Erwartungen von Kunden und Anwendern hinsichtlich der Qualität. Nur wenn das Testen diese Erfordernisse und Erwartungen erfüllt, kann das Testteam von sich behaupten, wirklich effektiv zu sein.

2.4 Testdokumentation und weitere Arbeitsergebnisse

Dokumentation entsteht oft als Teil des Testmanagements. Während Bezeichnungen und Umfang der Testmanagement-Dokumente oft variieren, so sind doch die folgenden typischen Testmanagement-Dokumente in Unternehmen und Projekten üblich:

- Testrichtlinie: Sie beschreibt die Unternehmensziele für das Testen.
- Teststrategie: Sie beschreibt die allgemeinen, projektunabhängigen Testmethoden des Unternehmens.
- Mastertestkonzept (bzw. Projekttestkonzept): Es beschreibt die Anwendung der Teststrategie für ein bestimmtes Projekt.
- Stufentestkonzept (bzw. Phasentestkonzept): Es beschreibt die in den jeweiligen Teststufen durchzuführenden Aktivitäten.

Die genaue Anordnung dieser verschiedenen Dokumente kann je nach Kontext variieren. In einigen Unternehmen und Projekten werden die oben angeführten Dokumente in einem einzigen Dokument zusammengefasst, oder sie sind in anderen Dokumenten enthalten, oder der Inhalt wird einfach als intuitive, ungeschriebene oder traditionelle Testmethodik zu Grunde gelegt. Größere, formal strukturierte Unternehmen und Projekte gestalten die Dokumente meist als Arbeitsergebnisse in schriftlicher Form, während in kleineren Unternehmen und Projekten meist weniger dokumentiert wird. Der Lehrplan beschreibt jedes der oben angeführten Dokumente einzeln, in der Praxis gibt aber der konkrete Unternehmens- und Projektzusammenhang vor, wie das jeweilige Dokument richtig anzuwenden ist.

2.4.1 Testrichtlinie

Die Testrichtlinie beschreibt, warum das Unternehmen testet. Sie definiert die allgemeinen Ziele, die das Unternehmen durch das Testen erreichen möchte. Die Richtlinie sollte vom leitenden Testmanagement des Unternehmens in Zusammenarbeit mit Führungskräften aller im Test involvierter Gruppen entwickelt werden.

Die Testrichtlinie kann entweder eine Ergänzung zur allgemeinen Qualitätsrichtlinie des Unternehmens sein oder ein Teil davon. Diese Qualitätsrichtlinie beschreibt die allgemeinen qualitätsbezogenen Werte und Ziele des Managements.

Wenn eine Testrichtlinie vorliegt, kann sie ein kurzes abstraktes Dokument sein, das

- den Wert des Testens für das Unternehmen zusammenfasst.

- die Ziele des Testens definiert, im Sinne einer vertrauensbildenden Maßnahme, dass das System wie beabsichtigt funktioniert, dass Fehlerzustände in der Software aufgedeckt werden, oder dass Qualitätsrisiken reduziert werden (siehe Abschnitt 2.3.1).
- beschreibt, wie Effektivität und Effizienz des Testens zur Erreichung dieser Ziele zu bewerten sind.
- einen Überblick über den typischen Testprozess gibt, eventuell basierend auf dem fundamentalen Testprozess des ISTQB®.
- beschreibt, wie das Unternehmen seinen Testprozess verbessern wird (siehe Kapitel 5).

Die Testrichtlinie sollte sich sowohl auf Testaktivitäten für Neuentwicklungen als auch auf die Wartung beziehen. Sie kann außerdem auf interne und/oder externe Standards für Testarbeitsprodukte und Testterminologie verweisen, die unternehmensweit verbindlich sind.

2.4.2 Teststrategie

Die Teststrategie beschreibt die Testmethodologie des Unternehmens. Beschrieben werden auch Produkt- und Projekt-Risikomanagement, die Aufteilung des Testens in Teststufen, sowie die übergeordneten testbezogenen Aktivitäten. (Ein Unternehmen kann durchaus verschiedene Strategien für unterschiedliche Situationen haben, wie z.B. unterschiedliche Softwarelebenszyklen, unterschiedliche Risikostufen, oder unterschiedliche gesetzliche Anforderungen). Der in der Teststrategie beschriebene Testprozess und die Testaktivitäten sollten der Testrichtlinie entsprechen. Die Teststrategie sollte die allgemeinen Testeingangs- und Endekriterien für das Unternehmen oder für ein oder mehrere Projekte liefern.

Wie bereits erwähnt, beschreibt die Teststrategie die Testmethodologie des Unternehmens. Diese beinhaltet normalerweise:

- Analytische Strategien, wie das risikoorientierte Testen. Hierbei analysiert das Testteam die Testbasis, um die abzudeckenden Testbedingungen zu identifizieren. Beim anforderungsbasierten Testen werden in der Testanalyse die Testbedingungen aus den Anforderungen abgeleitet, dann die Tests entworfen und realisiert, um diese Bedingungen abzudecken. Anschließend werden die Tests ausgeführt, wobei häufig die Priorität der vom jeweiligen Test abgedeckten Anforderung ausschlaggebend ist für die Reihenfolge der Tests. Testergebnisse werden hinsichtlich des Anforderungsstatus berichtet, z.B. Anforderung getestet und bestanden, Anforderung getestet und nicht bestanden, Anforderung noch nicht vollständig getestet, Testen der Anforderung geblockt, usw.
- Modellbasierte Strategien, wie das am Anwendungsprofil orientierte Testen. Hierbei entwickelt das Testteam (auf Basis tatsächlicher oder erwarteter Situationen) ein Modell der Umgebung des Systems, der Eingaben und Bedingungen des Systems und wie sich das System verhalten soll. Beispiel: Beim modellorientierten Performanztest einer schnell wachsenden Anwendung für mobile Endgeräte könnten Modelle entwickelt werden für ein- und ausgehenden Netzwerkverkehr, aktive und inaktive Nutzer, sowie die daraus resultierende Last, basierend auf der aktuellen Nutzung und dem zukünftigen Projektwachstum. Außerdem könnten Modelle entwickelt werden, die Hardware, Software, Datenkapazität, Netzwerk und Infrastruktur der derzeitigen Produktionsumgebung berücksichtigen. Es können auch Modelle für ideale, erwartete und minimale Durchsatzraten, Antwortzeiten und Ressourcenzuteilungen entwickelt werden.
- Systematische Strategien, wie z.B. die an Qualitätsmerkmalen orientierte Strategie. Hierbei verwendet das Testteam einen vorher festgelegten Satz von Testbedingungen, beispielsweise einen Qualitätsstandard (z.B. ISO 25000 [ISO 25000], ersetzt die ISO 9126 [ISO 9126]) eine Checkliste oder eine Sammlung generalisierter, abstrakter Testbedingungen, die einen bestimmten fachlichen Themenbereich, eine Applikation oder eine Testart (z.B. Sicherheitstest) betreffen. Diese Menge von Testbedingungen wird dann von Iteration zu Iteration, bzw. von Freigabe zu Freigabe verwendet. Beispiel: Für den Wartungstest einer einfachen, stabilen E-Commerce-Webseite kann eine Checkliste mit den Schlüsselfunktionen,

Attributen und Links der einzelnen Seiten verwendet werden. Die Tester würden die relevanten Punkte dieser Checkliste bei jeder Änderung an einer Seite abdecken.

- Prozess- oder standardkonforme Strategien, wie z.B. für medizinische Systeme, die dem Standard der amerikanischen Food & Drug Administration unterliegen. Hierbei befolgt das Testteam eine Menge von Prozessen, die von einem Normenausschuss oder einem Expertengremium definiert wurden. Die vorgeschriebenen Prozesse betreffen sowohl die Dokumentation, die korrekte Identifikation und Verwendung von Testbasis und Testorakel(n), als auch die Organisation des Testteams. Beispiel: In agilen Projekten, die Managementmethoden wie Scrum verwenden, analysieren die Tester in jeder Iteration User-Stories, die bestimmte Merkmale beschreiben, schätzen im Rahmen der Planung für die Iteration den Testaufwand für die einzelnen Features, identifizieren Testbedingungen (oft als Abnahmekriterien bezeichnet) für jede User-Story, führen die Tests zur Überdeckung dieser Bedingungen aus und berichten während der Testausführung den Status jeder User Story (ungetestet, nicht bestanden oder bestanden).
- Reagierende Strategien, wie z.B. fehlerbasierte Angriffe. Hierbei wartet das Testteam mit Entwurf und Realisierung, bis die Software vorliegt, reagiert also auf das tatsächliche System, das zu testen ist. Beispiel: Beim explorativen Testen einer menübasierten Anwendung könnte eine Menge von Test-Chartas entwickelt werden, die den Features, der Menüauswahl und den Bildschirmen entsprechen. Allen Testern werden Test-Chartas zugeteilt, anhand derer sie ihre explorativen Testsitzungen strukturieren. Tester berichten dem Testmanager regelmäßig über die Ergebnisse der Testsitzungen. Dieser kann die Test-Chartas aufgrund der Ergebnisse überarbeiten.
- Beratungsunterstützte Strategien, wie z.B. anwenderprofilgesteuertes Testen. Hierbei verlässt sich das Testteam auf die Eingaben eines oder mehrerer Stakeholder, um die abzudeckenden Testbedingungen zu bestimmen. Beispiel: Beim ausgelagerten Kompatibilitätstest einer webbasierten Applikation übergibt das Unternehmen dem Testdienstleister eine priorisierte Liste mit Browserversionen, Antivirus-Software, Betriebssystemen, Verbindungstypen und sonstigen Konfigurationsoptionen, die für ihre Anwendung bewertet werden sollen. Der Testdienstleister kann dann zur Erstellung der Tests verschiedene Verfahren einsetzen, z.B. paarweises Testen (für Optionen mit hoher Priorität) und Äquivalenzklassenbildung (für Optionen mit niedriger Priorität).
- Regressionstestorientierte Strategien, wie z.B. umfangreiche Automatisierung. Hier verwendet das Testteam unterschiedliche Verfahren für das Management des Regressionsrisikos, insbesondere der Automatisierung von funktionalen und/oder nicht-funktionalen Regressionstests auf einer oder mehreren Stufen. Beispiel: Beim Regressionstest einer webbasierten Anwendung können Tester ein GUI-basiertes Testautomatisierungswerkzeug einsetzen, um die Anwendungsfälle (ob typisch oder Ausnahme) der Applikation automatisiert zu testen. Diese Tests können dann jederzeit ausgeführt werden, wenn die Anwendung geändert wird.

Verschiedene Strategien lassen sich kombinieren. Die ausgewählte Strategie sollte sich an den Erfordernissen und Möglichkeiten des Unternehmens bzw. der Organisation orientieren und Unternehmen können die Strategien auf ihre eigenen speziellen Abläufe und Projekte zuschneiden.

Die Teststrategie kann die auszuführenden Teststufen beschreiben. Sie sollte dann eine grobe Leitlinie für die Eingangs- und Endekriterien jeder Teststufe sowie für die Beziehungen zwischen den Teststufen (beispielsweise Aufteilung von Testüberdeckungszielen) vorgeben.

Die Teststrategie kann außerdem beschreiben:

- Integrationsverfahren
- Testspezifikationsverfahren
- Unabhängigkeit des Testens (kann je nach Teststufe variieren)
- verpflichtend oder optional einzuhaltende Normen/Standards
- Testumgebungen

- Testautomatisierung
- Testwerkzeuge
- Wiederverwendbarkeit von Software und Arbeitsergebnissen des Testens
- Fehlernachtest (Bestätigungstest) und Regressionstests
- Teststeuerung und -berichte
- Testmaße und -metriken
- Fehlermanagement
- Konfigurationsmanagement der Testmittel
- Rollen und Zuständigkeiten

Möglicherweise müssen sowohl kurz- als auch langfristige Teststrategien definiert werden. Unterschiedliche Teststrategien eignen sich für unterschiedliche Unternehmen und unterschiedliche Projekte. Wenn es beispielsweise um sicherheitsrelevante oder sicherheitskritische Anwendungen geht, dann sind intensivere Strategien geeigneter. Außerdem variiert die Teststrategie je nach Entwicklungsmodell.

2.4.3 Mastertestkonzept

Das Mastertestkonzept deckt die gesamten Testaufgaben für ein spezifisches Projekt ab. Dazu gehören die auszuführenden Teststufen und die Beziehungen zwischen ihnen, sowie zwischen den Teststufen und den zugehörigen Entwicklungsaktivitäten. Das Mastertestkonzept sollte beschreiben, wie die generelle Teststrategie im spezifischen Projekt umzusetzen ist (d.h. die Testvorgehensweise). Das Mastertestkonzept sollte mit der Testrichtlinie und der Teststrategie konsistent sein. Falls es in bestimmten Bereichen davon abweicht, sollten die Abweichungen und Ausnahmen erklärt werden, auch hinsichtlich möglicher Auswirkungen. Falls die Teststrategie eines Unternehmens beispielsweise vorgibt, dass unmittelbar vor der Freigabe ein kompletter Regressionstest eines nicht geänderten Systems durchgeführt werden muss, im Projekt aber kein Regressionstest vorgesehen ist, dann sollte das Mastertestkonzept erklärt werden, warum dies so geplant ist und was unternommen wird, um das Risiko zu beherrschen, das aus der Abweichung von der normalen Strategie möglicherweise resultiert. Außerdem sollte das Mastertestkonzept etwaige weitere Auswirkungen erläutern, die aufgrund dieser Abweichungen zu erwarten sind. Ein Überspringen des Regressionstest könnte beispielsweise einen Wartungstest einen Monat nach der ersten Projektfreigabe erforderlich machen. Das Mastertestkonzept sollte den Projektplan oder das Betriebshandbuch ergänzen und den Testaufwand beschreiben, der Teil eines größeren Projekts oder einer größeren Operation ist.

Inhalt und Aufbau des Mastertestkonzepts können je nach Unternehmen, den im Unternehmen verwendeten Dokumentationsstandards und der im Projekt gelebten Formalität variieren. Typische Themen für ein Mastertestkonzept sind:

- Objekte, die getestet oder nicht getestet werden sollen
- Qualitätsmerkmale, die getestet oder nicht getestet werden sollen
- Testplan und Budget für das Testen (in Anlehnung an das Projekt- oder Gesamtbudget)
- Testdurchführungszyklen und deren Beziehung zum Release-Plan für die Software
- Beziehungen und Arbeitsergebnisse der Testabteilung zu anderen Personen oder Abteilungen
- definierte Abgrenzungen zwischen Testobjekten in den jeweiligen Teststufen
- Spezifikation der Eingangs-, Unterbrechungs- und Wiederaufnahmekriterien sowie der Endekriterien für jede Teststufe und der Beziehungen zwischen den Teststufen
- Testprojektrisiken
- Übergreifende Governance des Testens
- Zuständigkeit für die Durchführung der einzelnen Teststufen
- Testeingaben und Ausgaben der einzelnen Teststufen

In kleineren Projekten oder Unternehmen, in denen nur eine Teststufe formal getestet wird, werden Mastertestkonzept und Stufentestkonzept für diese Teststufe oft in einem Dokument zusammengefasst. Ist der Systemtest die einzige formale Teststufe, mit einem informellen, von den

Entwicklern durchgeführten Komponenten- und Integrationstest und einem informellen Abnahmetest, der vom Kunden als Teil eines Beta-Tests durchgeführt wird, so können all diese Elemente im Systemtestkonzept enthalten sein.

Meist hängt das Testen von anderen Aktivitäten im Projekt ab. Wenn diese Aktivitäten nicht ausreichend dokumentiert sind, vor allem was ihren Einfluss auf und ihre Beziehung zum Testen betrifft, dann kann dies im Mastertestkonzept (oder im entsprechenden Stufentestkonzept) abgedeckt werden. Ist beispielsweise der Konfigurationsmanagement-Prozess nicht dokumentiert, dann sollte im Mastertestkonzept festgehalten werden, wie Testobjekte an das Testteam geliefert werden.

2.4.4 Stufentestkonzept

Das Stufentestkonzept beschreibt die in jeder Teststufe auszuführenden Aktivitäten, manchmal auch die Testarten. Wenn nötig, kann es das Mastertestkonzept für die dokumentierte Teststufe oder Testart erweitern und beispielsweise Details zu Terminplan, Aufgaben und Meilensteinen spezifizieren, die im Mastertestkonzept nicht dokumentiert sind. Wenn unterschiedliche Standards und Dokumentvorlagen für die Spezifikation der Tests in verschiedenen Teststufen gelten, wird dies im Stufentestkonzept genau festgelegt.

In weniger formalen Projekten oder Unternehmen ist das Stufentestkonzept oft das einzige Testmanagementdokument. Es kann dann einige der in diesem Abschnitt bereits erwähnten Informationen enthalten.

Bei agilen Projekten können Sprint- oder Iterations-Testkonzepte den Stufentestplan ersetzen.

2.4.5 Projektrisikomanagement

Ein wichtiger Teil der Planungsaktivitäten betrifft den Umgang mit Projektrisiken. Diese lassen sich mit den in Abschnitt 2.3 beschriebenen Prozessen identifizieren. Wenn Projektrisiken identifiziert wurden, müssen die Projektmanager eventuell darüber informiert werden und handeln. Nicht alle Risiken lassen sich innerhalb der Testorganisation durch entsprechende Maßnahmen reduzieren. Es gibt aber auch Projektrisiken, die die Testmanager erfolgreich abschwächen können:

- einsatzbereite Testumgebung und Werkzeuge
- Verfügbarkeit und Qualifikation des Testpersonals
- fehlende Standards, Regeln und Verfahren für die Testaufgaben

Zum Vorgehen bei der Risikobeherrschung gehören: frühe Vorbereitung der Testmittel, Vorabtests der Testumgebung, Vorabtests früher Produktversionen, strengere Eingangskriterien, Anforderungen an die Testbarkeit, Teilnehmen an Reviews früher Projektergebnisse, Teilnehmen am Änderungsmanagement, Überwachung des Projektfortschritts und der Qualität.

Nachdem ein Projektrisiko identifiziert und analysiert wurde, gibt es vier Möglichkeiten damit umzugehen:

1. das Risiko durch vorbeugende Maßnahmen beherrschen, um Eintrittswahrscheinlichkeit und/oder -auswirkungen zu minimieren
2. Notfallpläne, um das mögliche Schadensausmaß bei einem Auftreten des Risikos zu reduzieren
3. das Risiko an eine andere Partei auslagern
4. das Risiko akzeptieren und nicht weiter beachten

Welche dieser Optionen tatsächlich gewählt wird, hängt sowohl von den Vorteilen und Chancen jeder Option, als auch von ihren Kosten und möglichen Folgerisiken ab. Wenn für ein Projektrisiko ein Notfallplan vorgesehen wird, dann ist es am besten, einen Auslöser zu identifizieren (d.h. zu bestimmen, wann und wie der Notfallplan greift) sowie einen Besitzer (der den Notfallplan durchführt).

2.4.6 Weitere Arbeitsergebnisse des Testens

Beim Testen gibt es auch weitere Arbeitsergebnisse, z.B. Fehlerberichte, Testfallspezifikationen und Testprotokolle; die meisten werden von Test Analysten und Technischen Test Analysten erstellt. Im Test Analyst-Modul des Advanced-Level-Lehrplans wird die Erstellung und Dokumentierung dieser Arbeitsergebnisse behandelt. Der Testmanager sollte durch die folgenden Aktivitäten die Konsistenz und Qualität dieser Arbeitsprodukte sicherstellen:

- Metriken bestimmen und überwachen, anhand derer die Qualität dieser Arbeitsergebnisse überwacht werden kann, z.B. der prozentuale Anteil zurückgewiesener Fehlerberichte
- Gemeinsam mit den Test Analysten und den Technischen Test Analysten geeignete Vorlagen für diese Arbeitsergebnisse auswählen und ggf. anpassen
- Gemeinsam mit den Test Analysten und den Technischen Test Analysten die Standards für diese Arbeitsergebnisse festlegen, z.B. den notwendigen Detaillierungsgrad für Tests, Protokolle und Berichte
- Arbeitsergebnisse überprüfen mit Hilfe der geeigneten Verfahren und der jeweiligen Teilnehmer und Stakeholder

Umfang, Art und spezifischer Zweck der Testdokumentation können unter anderem vom gewählten Softwarelebenszyklus, den einzuhaltenden Normen und Standards, sowie von den Produkt- und Projektrisiken in Zusammenhang mit dem System, das entwickelt wird, beeinflusst werden.

Es gibt unterschiedliche Vorlagen für Testdokumentation und Arbeitsergebnisse, z.B. IEEE Standard 829 [IEEE 829]. Testmanager sollten jedoch bedenken, dass die im IEEE Standard 829 beschriebenen Dokumente für sämtliche Branchen bestimmt sind und einen entsprechend hohen Detaillierungsgrad haben. Nicht alles ist für ein bestimmtes Unternehmen relevant. Es empfiehlt sich daher, die im IEEE Standard 829 beschriebenen Dokumente anzupassen und für das Unternehmen eigene Standardvorlagen zu erstellen. Eine durchgehende Anwendung dieser Vorlagen verringert den Schulungsaufwand und unterstützt die Vereinheitlichung der Prozesse im Unternehmen.

Außerdem werden beim Testen auch Testberichte erstellt. Dies ist in der Regel eine Aufgabe des Testmanagers und wird später in diesem Kapitel beschrieben.

2.5 Testaufwandsschätzung

Aufwandsschätzungen als Managementaktivität führen zu einem ungefähren Kosten- und Terminplan für die Aktivitäten in einem bestimmten Betrieb oder Projekt. Die besten Aufwandsschätzungen:

- repräsentieren die kollektive Erfahrung von erfahrenen Praktikern und werden von allen Stakeholdern getragen
- liefern detaillierte Aufstellungen der Kosten, Ressourcen, Aufgaben und beteiligten Mitarbeiter
- zeigen die wahrscheinlichen Kosten, Aufwand und Dauer für jede geschätzte Aktivität

Die Schätzung von Aufgaben in der Software- und Systementwicklung ist bekanntermaßen schwierig, sowohl technisch als auch politisch, obwohl es im Projektmanagement Best-Practice-Verfahren für die Schätzung gibt. Die Testschätzung ist die Anwendung dieser Verfahren auf die Testaktivitäten in einem Projekt oder Unternehmen.

Die Testschätzung sollte alle Aktivitäten im Testprozess einschließen, wie in Kapitel 1 beschrieben. Geschätzte Kosten, Aufwand und vor allem Dauer der Testdurchführung interessieren das Management oft besonders, weil die Testdurchführung typischerweise auf dem kritischen Pfad des Projekts liegt. Schätzungen der Testdurchführung sind aber schwierig und tendenziell unzuverlässig, wenn die Gesamtqualität der Software schlecht ist oder nicht bekannt. Außerdem wird die Qualität der Aufwandsschätzung auch entscheidend von Faktoren wie der Vertrautheit und Erfahrung mit dem

System beeinflusst. Es ist gängige Praxis, die Anzahl von erforderlichen Testfällen zu schätzen. Das funktioniert jedoch nur dann, wenn angenommen werden kann, dass die zu testende Software nur wenig Fehlerzustände enthält. Die getroffenen Annahmen bei der Aufwandsschätzung sollten immer als Teil der Aufwandsschätzung dokumentiert werden.

Zu diesen Faktoren gehören (ohne Anspruch auf Vollständigkeit):

- erforderliches Qualitätsniveau des Systems
- Größe und Komplexität des zu testenden Systems
- historische Daten aus früheren Testprojekten, sowie Branchendaten oder Benchmark-Daten anderer Unternehmen
- Prozessfaktoren, wie Teststrategie, Entwicklungs- und Wartungslebenszyklus, Prozessreife, Richtigkeit der Projektschätzung
- Wesentliche Faktoren, wie Testautomatisierung und Testwerkzeuge, Testumgebung(en), Testdaten, Entwicklungsumgebung(en); Projektdokumentation (beispielsweise Anforderungen, Entwürfe, usw.) und wiederverwendbare Arbeitsergebnisse
- Personalfaktoren, wie Manager und Technische Leiter; Engagement und Erwartungen der Geschäftsleitung und des oberen Managements; fachliches Können, Erfahrung und Motivation im Projektteam, Stabilität des Projektteams, Beziehungen der Projektmitglieder untereinander; Hilfe bei der Nutzung der Test- und Debugging-Umgebung; Verfügbarkeit qualifizierter Auftragnehmer und Berater; Fachwissen.
- Komplexität des Testprozesses; Technik, Organisation; Anzahl der Test Stakeholder; viele räumlich getrennte Teilteams und deren Zusammensetzung
- außergewöhnlicher Aufwand für den Aufbau des Testteams, Training und Einarbeitung
- Anpassung oder Entwicklung von neuen Testwerkzeugen, Technologien, Verfahren, kundenspezifische Hardware, große Anzahl von Testmitteln
- Anforderungen für eine außergewöhnlich detaillierte Testspezifikation, besonders bei Anwendung eines nicht vertrauten Dokumentationsstandards
- komplexe Terminierung der Komponentenbeschaffung, besonders für Integrationstest und Testentwicklung
- veränderliche Testdaten (beispielsweise Daten mit Verfallsfristen)

Auch die Qualität der zu testenden Software ist ein Hauptfaktor, den Testmanager bei der Aufwandsschätzung berücksichtigen sollten. Wenn beispielsweise in der Entwicklung Best-Practice-Verfahren wie automatisierte Modultests und kontinuierliche Integration verwendet wurden, dann sind bereits bis zu 50% der Fehlerzustände behoben, bevor der Code an das Testteam geliefert wird (siehe [Jones11] bezüglich weiterer Informationen über die Fehlerbehebungseffektivität dieser Praktiken). Es wurde auch berichtet, dass agile Methoden, einschließlich der testgetriebenen Entwicklung, Software mit einem höheren Qualitätsniveau an das Testen liefern.

Die Testaufwandsschätzung lässt sich Bottom-Up oder Top-Down durchführen, dabei können die folgenden Testaufwandsschätzverfahren eingesetzt bzw. miteinander kombiniert werden:

- Intuition und Raten, sowie Erfahrungen aus früheren Projekten
- Projektstrukturplan (PSP)
- gemeinsame Schätzungen im Team (beispielsweise Breitband-Delphi)
- Unternehmensstandards und Normen
- prozentualer Anteil am Gesamtprojekt oder an bestimmten Mitarbeitergruppen (beispielsweise das Verhältnis von Testern zu Entwicklern)
- Erfahrungen und Metriken, einschließlich darauf basierender Modelle, zum Schätzen der Anzahl von Fehlerwirkungen, Testzyklen, Testfällen, des durchschnittlichen Aufwands je Test und der Anzahl von Regressionstestzyklen

- Durchschnittswerte der Branche und Vorhersagemodelle, wie Funktionspunkte (function points), Anzahl der Codezeilen, geschätzter Aufwand der EntwicklerInnen oder andere Projektparameter (siehe beispielsweise [Jones07]).

Wenn die Testaufwandsschätzung erstellt ist, muss sie meist mit einer Begründung dem Management vorgelegt werden (siehe Abschnitt 2.7). Oft folgen Verhandlungen, die nicht selten zu einer Überarbeitung der vorgelegten Zahlen führen. Im Idealfall ist die endgültige Version der Testaufwandsschätzung der beste mögliche Kompromiss zwischen Unternehmenszielen und Projektzielen hinsichtlich Qualität, Zeitplan, Budgetierung und erwarteter Funktionalität.

Es muss daran erinnert werden, dass jede Schätzung auf den zum Zeitpunkt der Schätzung verfügbaren Informationen basiert. In frühen Projektphasen sind eventuell nur spärliche Informationen vorhanden. Außerdem können sich früh verfügbare Informationen im Verlauf eines Projekts ändern. Aus diesen Gründen müssen die Testaufwandsschätzungen aktualisiert und neue bzw. geänderte Informationen berücksichtigt werden.

2.6 Definieren und Anwenden von Testmetriken

Ein Managementklischee besagt, dass das, was gemessen wird, auch erledigt wird. Umgekehrt wird das, was nicht gemessen wird, nicht erledigt, weil es leicht ignoriert werden kann. Es ist daher wichtig, dass für alle Aufgaben, auch für das Testen, korrekte Metriken eingeführt werden.

Testmetriken lassen sich wie folgt klassifizieren und können zu einer oder mehreren der folgenden Kategorien gehören:

- Projektmetriken, die den Fortschritt gegen festgelegte ProjektEndekriterien messen, z.B. der prozentuale Anteil der ausgeführten, bestandenen und nicht bestandenen Testfälle
- Produktmetriken, die ein Attribut des Produkts messen, z.B. inwieweit das Produkt getestet wurde oder die Defektdichte
- Prozessmetriken, die die Prozessreife der Test- und Entwicklungsprozesse messen, z.B. der prozentuale Anteil der vom Testen aufgedeckten Fehlerzustände
- Personenbezogene Metriken, die die Fähigkeiten einzelner Mitarbeiter oder Gruppen messen, z.B. Anzahl der implementierten Testfälle in einem bestimmten Zeitraum.

Jede Metrik kann zu zwei, drei oder sogar zu allen vier Kategorien gehören. Tendenzdiagramme können beispielsweise die Anzahl der aufgetretenen Fehlerzustände pro Tag in Zusammenhang mit einem Ausgangskriterium (keine neuen Fehlerzustände in einer Woche) darstellen; oder in Zusammenhang mit dem Produkt (es können keine weiteren Fehlerzustände durch Testen gefunden werden); oder in Zusammenhang mit der Testprozessreife (das Testen identifiziert viele Fehlerzustände früh in der Testausführungsphase).

Besondere Vorsicht ist bei den personenbezogenen Metriken geboten. Es kommt manchmal vor, dass Manager Metriken, die eigentlich in erster Linie Prozessmetriken sind, als personenbezogene Metriken auffassen. Das kann dann katastrophale Auswirkungen haben, wenn Mitarbeiter die Metriken zu ihrem eigenen Vorteil manipulieren. Korrekte Motivierung und Beurteilung der Testmitarbeiter wird in Kapitel 7 dieses Lehrplans sowie im Expert-Level-Lehrplan „Testmanagement“ behandelt.

Im Advanced-Level-Lehrplan beschäftigen wir uns hauptsächlich mit dem Anwenden von Metriken zur Messung des Testfortschritts, d.h. mit Projektmetriken. Einige der Metriken, die für die Messung des Testfortschritts verwendet werden, betreffen auch Produkt und Prozess. Weitere Informationen über die Anwendung von Produkt- und Prozessmetriken durch das Management sind im Testmanager-Modul des Expert-Level-Lehrplans enthalten. Weitere Informationen über die Anwendung von Prozessmetriken sind im Modul Testprozessverbesserung des Expert Level Lehrplans enthalten.

Mit Metriken können Tester ihrem Management in gleichbleibender Art und Weise berichten und der Fortschritt im zeitlichen Verlauf lässt sich kohärent verfolgen. Häufig müssen sie diese in Besprechungen verschiedenen Ebenen von Stakeholdern, vom technischen Personal bis zur Geschäftsleitung, präsentieren. Da Metriken manchmal dazu dienen, den Erfolg eines Projekts zu bestimmen, sollte sehr sorgfältig überlegt werden, was verfolgt wird, wie oft berichtet wird und welche Methode zur Präsentation der Informationen eingesetzt wird. Insbesondere müssen Testmanager die folgenden Bereiche berücksichtigen:

- Metriken definieren: Eine begrenzte Menge von sinnvollen Metriken sollte definiert werden. Metriken können passend zu den Zielen eines Projekts, Prozesses oder eines Produkts festgelegt werden. Die definierten Metriken sollten ausgewogen sein, da einzelne Metriken einen irreführenden Eindruck über Status und Tendenzen vermitteln können. Sind die Metriken definiert, müssen sich alle Stakeholder auf ihre Interpretation einigen, um zukünftige Diskussionen zu vermeiden, wenn die Metrikwerte vorliegen. Oft werden tendenziell zu viele Metriken definiert, anstatt die aussagekräftigsten festzulegen.
- Metriken verfolgen: Berichte über und Zusammenstellungen von Metriken sollten möglichst weitgehend automatisiert werden, um den zeitlichen Aufwand für die Generierung der Metrikwerte niedrig zu halten. Abweichungen bei den Daten im zeitlichen Ablauf könnten auch andere Informationen widerspiegeln als die, über deren Interpretation in der Definitionsphase Einigkeit erzielt wurde. Testmanager sollten bereit sein, mögliche Divergenzen zwischen gemessenen und erwarteten Ergebnissen sorgfältig zu analysieren, einschließlich der Gründe für diese Abweichungen.
- Metriken berichten: Ziel ist es, die Information für das Management unmittelbar verständlich darzustellen. Präsentationen können eine Momentaufnahme der Metrik zu einem bestimmten Zeitpunkt wiedergeben oder die zeitliche Entwicklung der Metrik(en), sodass sich Tendenzen einschätzen lassen.
- Gültigkeit der Metriken: Testmanager müssen außerdem die Informationen verifizieren, die berichtet werden. Die Messungen einer bestimmten Metrik reflektieren möglicherweise nicht den tatsächlichen Status eines Projekts, oder vermitteln einen zu positiven oder negativen Trend. Bevor Testmanager Daten präsentieren, müssen diese sowohl auf Genauigkeit als auch auf die Botschaft geprüft werden, die damit vermittelt wird.

Die fünf wesentlichen Dimensionen für die Überwachung des Testfortschritts sind:

- Produkt-/Qualitätsrisiken
- Fehler
- Testfallstatus
- Überdeckungen
- Vertrauen in die Software

Während des Regelbetriebs oder eines Projekts werden Produktrisiken, Fehlerzustände, Testfälle und Überdeckungsgrad auf spezielle Art und Weise gemessen und berichtet (Testfortschrittsbericht). Wenn diese Messungen mit den im Testkonzept definierten Endekriterien verknüpft sind, liefern sie eine objektive Grundlage für die Beurteilung, ob der Testaufwand abgeschlossen ist. Vertrauen in die Software-Qualität ist allenfalls durch Umfragen messbar oder es werden Überdeckungsmetriken verwendet (als Surrogat); im Grunde wird Vertrauen jedoch auch nur subjektiv berichtet.

Mögliche Produktrisiken-basierte Metriken sind

- Prozentualer Anteil von Risiken, die komplett abgedeckt sind (Test bestanden)
- Prozentualer Anteil von Risiken, die teilweise abgedeckt sind (einige oder mehrere Tests nicht bestanden)
- Prozentualer Anteil von Risiken, die noch nicht komplett getestet sind
- Prozentualer Anteil der abgedeckten Risiken, sortiert nach Risikokategorie
- Prozentualer Anteil von Risiken, die nach der ersten Produktrisikoanalyse identifiziert wurden

Mögliche Fehlerbasierte Metriken sind

- Gesamtzahl der gemeldeten (identifizierten) gegenüber der Gesamtzahl behobener (abgeschlossener) Fehlerzustände
- durchschnittliche Zeit zwischen dem Auftreten von Fehlerwirkungen
- Aufschlüsselung der Anzahl oder des prozentualen Anteils von Fehlerzuständen nach den folgenden Kategorien:
  - bestimmte Testobjekte oder Komponenten
  - Grundursachen
  - Fehlerquellen (z.B. Anforderungsspezifikation, neue Funktionalität, Regression, usw.)
  - Testversionen
  - Phasen, in denen sie eingeführt, festgestellt oder entfernt wurden
  - Priorität/Schweregrad
  - Fehlerberichte, die zurückgewiesen oder dupliziert wurden
- Trends bei der Dauer zwischen Eingeben der Fehlermeldung und Behebung
- Anzahl der Fehlerbehebungen, die neue Fehlerzustände eingeschleust haben

Mögliche Testfallstatus-basierte Metriken sind

- Gesamtzahl geplanter, spezifizierter (entwickelter), durchgeführter, bestandener/nicht bestandener, blockierter und ausgelassener Tests
- Status der Regressions- und Fehlernachtests, einschließlich der Trends und Gesamtzahlen nicht bestandener Regressions- und Fehlernachtests
- Anzahl für das Testen geplanter gegenüber tatsächlich geleisteter Stunden pro Tag
- Verfügbarkeit der Testumgebung (prozentualer Anteil geplanter Teststunden, in denen die Testumgebung für das Testteam nutzbar ist)

Mögliche Metriken für die Testüberdeckungsbasierte Metriken sind

- Anforderungsüberdeckung und Überdeckung der Entwurfselemente
- Risikoüberdeckung
- Testumgebungs-/Konfigurationsüberdeckung
- Codeüberdeckung

Es ist wichtig, dass Testmanager verstehen, wie Überdeckungsmetriken interpretiert und angewendet werden. In den höheren Teststufen (z.B. Systemtest, Integrationstest und Abnahmetest) basiert das Testen in erster Linie auf Arbeitsergebnissen, wie z.B. Anforderungsspezifikationen, Entwurfsspezifikationen, Anwendungsfälle, User-Stories, Produktrisiken, unterstützte Umgebungen und Konfigurationen. Strukturelle Codeüberdeckungsmetriken betreffen eher die unteren Teststufen, wie z.B. den Modultest (z.B. Anweisungs- oder Zweigüberdeckung) und den Komponentenintegrationstest (z.B. Schnittstellenüberdeckung). Obwohl Testmanager Codeüberdeckungsmetriken verwenden können, um zu messen, inwieweit ihre Tests die Strukturen des zu testenden Systems ausführen, sollten diese Metriken nicht verwendet werden, um über die Testergebnisse der höheren Teststufen zu berichten. Außerdem müssen Testmanager verstehen, dass – auch wenn im Modultest und Modulintegrationstest 100% der strukturellen Überdeckungsziele erreicht wurden – Fehlerzustände und Risiken in den höheren Teststufen noch berücksichtigt werden müssen.

Metriken können auch mit den Aktivitäten des fundamentalen Testprozesses verknüpft sein (der im Foundation Level Lehrplan und im vorliegenden Advanced Level Lehrplan beschrieben ist). Dadurch können Metriken im gesamten Testprozess verwendet werden, um den Testprozess selbst sowie den Fortschritt gegenüber den Projektzielen zu überwachen.

Mögliche Metriken zur Überwachung der Testplanung und -steuerung sind

- Risiko-, Anforderungs- und sonstige Abdeckungen von Testbasiselementen
- Fehlerfindung
- Verhältnis geplanter zu tatsächlich aufgewendeten Stunden für die Entwicklung der Testmittel und die Durchführung der Testfälle

Mögliche Metriken für die Überwachung der Testanalyse sind

- Anzahl der identifizierten Testbedingungen
- Anzahl der Fehlerzustände, die während der Testanalysephase aufgedeckt wurden (z.B. durch Identifikation von Risiken oder Testbedingungen anhand der Testbasis)

Mögliche Metriken für die Überwachung des Testentwurfs sind

- prozentualer Anteil der Anforderungen, die von Testbedingungen abgedeckt werden
- Anzahl der Fehlerzustände, die während der Testentwurfsphase aufgedeckt wurden (z.B. durch Entwurf von Tests gegen die Testbasis)

Mögliche Metriken für die Überwachung der Testrealisierung sind

- prozentualer Anteil der konfigurierten Testumgebungen
- prozentualer Anteil der geladenen Testdatensätze
- prozentualer Anteil der automatisierten Testfälle

Mögliche Metriken für die Überwachung der Testausführung sind

- prozentualer Anteil der geplanten Testfälle, die durchgeführt, bestanden bzw. nicht bestanden wurden
- prozentualer Anteil der Testbedingungen, die durch ausgeführte (und/oder bestandene) Testfälle abgedeckt wurden
- vorausgesagte Fehlerzustände gegenüber tatsächlich berichteten/behobenen Fehlerzuständen
- geplante Überdeckung gegenüber tatsächlich erzielter Überdeckung

Zu den Metriken für die Überwachung von Testfortschritt und -abschluss gehört, dass die in der Testplanungsphase festgelegten (und genehmigten) Meilensteine, Eingangs- und Endekriterien abgebildet sind. Dies kann durch eine oder mehrere der folgenden Metriken erfolgen:

- Vergleich der geplanten mit den tatsächlich durchgeführten Testbedingungen, Testfällen oder Testspezifikationen, jeweils mit der Angabe, ob sie bestanden wurden oder nicht
- Gesamtzahl der aufgedeckten Fehlerzustände, jeweils mit Angabe von Schweregrad, Priorität, Status, betroffenem Teilsystem, oder sonstiger Klassifizierung (siehe Kapitel 4)
- Anzahl der Änderungen (Änderungsanträge), die erzeugt, akzeptiert, implementiert und getestet wurden
- geplante Kosten gegenüber tatsächlichen Kosten
- geplanter Zeitaufwand gegenüber tatsächlich aufgewendeter Zeit
- geplante Termine gegenüber tatsächlichen Terminen von Testmeilensteinen
- geplante Termine gegenüber tatsächlichen Terminen testrelevanter Projektmeilensteine (z.B. Code-Freeze)
- Status von identifizierten Produkt- und Qualitätsrisiken, eingeteilt in reduzierte Risiken, offene Risiken, Hauptrisikobereiche, neue Risiken, die nach der Testanalyse entdeckt wurden, usw.
- prozentualer Anteil des Testaufwands, Kosten oder Testzeit, die durch blockierende Ereignisse oder geplante Änderungen verloren gingen
- Status der Fehlernachtest und Regressionstests

Mögliche Metriken für den Abschluss der Testaktivitäten sind

- prozentualer Anteil der bei der Testdurchführung ausgeführten, bestandenen, nicht bestandenen, blockierten und übersprungenen Testfälle
- prozentualer Anteil der Testfälle, die in das Repository für wiederverwendbare Testfälle aufgenommen wurden
- prozentualer Anteil der automatisierten gegenüber den noch zu automatisierenden Testfällen
- prozentualer Anteil der Testfälle, die in den Regressionstests aufgenommen wurden
- prozentualer Anteil von Fehlermeldungen, die geschlossen bzw. nicht geschlossen wurden
- prozentualer Anteil der identifizierten und archivierten Arbeitsergebnisse

Außerdem werden für die Überwachung des Testprozesses häufig normale Projektmanagement-Methoden verwendet, wie z.B. das Aufteilen in Aufgabenblöcke im Rahmen eines Projektstrukturplans. In agilen Teams ist das Testen Teil des Fortschritts einer User-Story auf einem Burndown-Chart. Wenn Methoden des Lean Managements verwendet werden, wird der Testfortschritt (ebenfalls basierend auf Stories) überwacht, indem die Karte mit der User-Story eine Spalte auf der Kanban-Tafel durchläuft.

Die Messungen zu den definierten Metriken können in Textform, tabellarisch oder graphisch berichtet werden und lassen sich für verschiedene Zwecke nutzen, beispielsweise:

- Analysen, um anhand der Testergebnisse herauszufinden, welche Trends und Ursachen erkennbar sind
- Berichte, um die Testergebnisse an Projektmitglieder und Stakeholder zu kommunizieren
- Teststeuerung, um den Verlauf eines Tests oder auch des Projekts als Ganzes zu ändern und um die Ergebnisse der Kurskorrektur zu überwachen

Wie die Messwerte der Tests in geeigneter Weise gesammelt, analysiert und berichtet werden, hängt vom spezifischen Informationsbedarf ab, sowie von den Zielen und den Fähigkeiten der Adressaten, die diese Messungen nutzen. Außerdem sollte der Inhalt der Testberichte auf die Zielgruppe abgestimmt sein und entsprechend variieren.

Für die Teststeuerung müssen die Metriken über den gesamten Testprozess (nach Abschluss der Testplanung) den Testmanagern die Informationen liefern, die sie benötigen, um einen erfolgreichen Abschluss der Tests hinsichtlich der übergeordneten Zielsetzung, Teststrategie und Testziele herbeizuführen. Daher muss der spezifische Informationsbedarf bei der Planung berücksichtigt werden und die Testüberwachungsaktivitäten müssen das Sammeln der benötigten Metriken einschließen. Der genaue Umfang der Informationen und der Aufwand für das Sammeln sind abhängig von verschiedenen Projektfaktoren, einschließlich Projektgröße, Komplexität und Risiko.

Die Teststeuerung muss auf die Informationen durch das Testen und auf veränderte Rahmenbedingungen eines Projekts oder Vorhabens adäquat reagieren. So müssen Risikoanalyse und Zeitplan revidiert werden, wenn beispielsweise beim dynamischen Testen gehäufte Fehlerzustände in Bereichen entdeckt werden, wo man sie für unwahrscheinlich gehalten hatte, oder wenn die verfügbare Zeit für die Tests wegen einer Verzögerung beim Testbeginn gekürzt wurde. Das kann dazu führen, dass Tests neu priorisiert und verbleibende Ressourcen für die Testausführung neu verteilt werden müssen.

Wenn durch den Testfortschrittsbericht Abweichungen vom Testkonzept festgestellt werden, muss die Teststeuerung eingreifen. Ziel der Steuerungsmaßnahmen ist eine Kurskorrektur, die das Projekt und/oder das Testen in eine erfolgreichere Richtung lenkt.

Wenn der Steuerungsaufwand eines Projekts anhand der Testergebnisse gemessen oder beeinflusst werden soll, bestehen folgende Möglichkeiten:

- Qualitätsrisikoanalyse, Testprioritäten und/oder Testkonzepte überarbeiten
- weitere Ressourcen bereitstellen oder den Testaufwand bzw. Projektaufwand erhöhen
- Freigabetermin verschieben
- Endekriterien abschwächen oder verschärfen
- Änderung des Projektumfangs (funktionale und/oder nicht-funktionale Aspekte)

Die Umsetzung erfordert normalerweise Konsens unter den Projektmitarbeitern und Stakeholdern im Unternehmen sowie die Zustimmung von Projektmanagern oder Betriebsleitern.

Wie der Testbericht strukturiert ist, hängt weitgehend von den Adressaten ab, beispielsweise Projektmanagement oder Geschäftsführung. Projektmanagerin oder Projektmanager interessieren sich wahrscheinlich für ausführliche Informationen zu Fehlerzuständen, während das Hauptinteresse der Geschäftsführung bei Informationen über den Status der Produktrisiken liegen dürfte.

2.7 Mehrwert des Testens

Testmanager müssen sich dafür einsetzen, den Mehrwert des Testens zu optimieren. Wenn zu viel getestet wird, liefert das Testen keinen guten Mehrwert, weil es unzumutbare Verzögerungen mit sich bringt und mehr Kosten verursacht, als es einspart. Wenn zu wenig getestet wird, liefert das Testen keinen guten Mehrwert, weil zu viele Fehlerzustände an die Benutzer weitergegeben werden. Das Optimum liegt zwischen den beiden Extremen. Es ist die Aufgabe des Testmanagers, dafür zu sorgen, dass die Stakeholder dieses Optimum und den Mehrwert des Testens verstehen.

Zwar halten die meisten Unternehmen das Testen in gewisser Weise für wertvoll, aber nur wenige Manager, Testmanager eingeschlossen, können diesen Wert quantifizieren, beschreiben oder in Worte fassen. Viele Testmanager, Testleiter und Tester konzentrieren sich vor allem auf die Ausführungsdetails des Testens (Aspekte, die die konkrete Aufgabe oder Teststufe betreffen), lassen aber die übergeordneten strategischen Gesichtspunkte außer Acht, für die sich andere Projektbeteiligte interessieren, vor allem Manager.

Testen liefert sowohl Organisationen und Projekten als auch dem Betrieb quantitativen und qualitativen Mehrwert:

- Der quantitative Mehrwert des Testens liegt darin, dass es Fehlerzustände vor einer Freigabe vermeidet oder behebt, oder Fehlerzustände vor der Freigabe bekannt macht (nicht behoben, aber dokumentiert und evtl. mit Lösungsalternativen), Risiken durch die Tests vermindert und Informationen über den Projekt-, Prozess- und Produktstatus liefert.
- Der qualitative Mehrwert des Testens liegt in gesteigerter Reputation für Qualität, reibungsloseren und besser berechenbaren Releases, neuem und/oder gesteigertem Vertrauen in die Software, Schutz vor Haftungsansprüchen sowie im Reduzieren des Risikos, ganze Aufträge oder sogar Menschenleben zu verlieren.

Testmanager und Testleiter sollten verstehen, wo der relevante geschäftliche Nutzen für ihr Unternehmen, ihr Projekt und/oder den Betriebsablauf liegt und sie sollten anderen diesen geschäftlichen Nutzen des Testens vermitteln können.

Eine bewährte Methode für die Messung von quantitativem Nutzen und Effizienz des Testens ist in dem Begriff Qualitätskosten zusammengefasst (manchmal auch als Kosten schlechter Qualität bezeichnet). Die Qualitätskosten fassen Projekt- oder Betriebskosten in Bezug auf die Produktfehlerkosten in vier Kategorien zusammen:

- Kosten für die Vorbeugung, z.B. Schulung der Entwickler in der Erstellung wartbaren und sicheren Codes

- Kosten für die Aufdeckung, z.B. Testfälle entwerfen, Testumgebungen konfigurieren und Anforderungen prüfen
- Kosten für interne Fehlerwirkungen, z.B. Defekte, die beim Testen oder bei Reviews gefunden wurden, vor Auslieferung beheben
- Kosten für externe Fehlerwirkungen, z.B. Kosten in Zusammenhang mit fehlerhafter Software, die an den Kunden geliefert wurde

Die Kosten für die Aufdeckung machen einen Teil des Budgets für das Testen aus (d.h. Geld, das für die Entwicklung von Tests ausgegeben würde, selbst wenn die Tester keine Fehlerzustände finden). Der restliche Teil des Budgets entfällt auf Kosten für interne Fehlerwirkungen (d.h. die tatsächlichen Kosten in Zusammenhang mit gefundenen Fehlerzuständen). Die Gesamtkosten für Aufdeckung und interne Fehlerwirkungen liegen normalerweise deutlich unter den Kosten für externe Fehlerwirkungen. Dies macht das Testen deshalb außerordentlich wertvoll. Testmanager und Testleiter haben überzeugende wirtschaftliche Argumente für das Testen, wenn sie die Kosten in diesen vier Kategorien aufzeigen können.

Mehr Informationen über den Geschäftswert des Testens und über die Kosten der Qualität finden Sie in [Black03].

2.8 Verteiltes Testen, Outsourcing und Insourcing

Manchmal wird ein Teil des Testaufwands oder sogar der gesamte Testaufwand von Personen an unterschiedlichen Standorten erbracht, die bei anderen Unternehmen beschäftigt sind, oder nicht am selben Ort wie das Projektteam arbeiten. Beim Testen an mehreren Standorten spricht man von verteiltem Testen. Wenn Personen an einem oder mehreren Standorten testen, die nicht Mitarbeiter des Unternehmens sind und nicht am Standort des Projektteams arbeiten, spricht man von Outsourcing des Testens. Wenn Personen testen, die nicht Mitarbeiter des Projektteams sind, aber am selben Standort arbeiten, dann spricht man von Insourcing des Testens.

Alle drei Arten der Testorganisation brauchen klare Kommunikationswege und gut definierte Erwartungen an Ziel, Aufgaben und Arbeitsergebnisse. Das Projektteam kann sich dabei kaum auf informelle Kommunikationswege verlassen, wie Gespräche unter Kollegen im Flur, oder auf außerbetriebliche soziale Kontakte. Es ist wichtig, dass Kommunikationswege und -mittel klar definiert sind; dabei müssen Themen angesprochen werden, wie das Eskalieren von Problemen, die Art der Informationen, die kommuniziert werden müssen, sowie die vorgesehenen Kommunikationsmethoden. Jeder Beteiligte, auf jeder Seite der Teambeziehungen, muss sowohl die Rollen und Zuständigkeiten der eigenen Seite wie auch der anderen Seite klar verstehen, damit Missverständnisse und unrealistische Erwartungen vermieden werden. Durch unterschiedliche Standorte, Zeitzonen, kulturelle und sprachliche Verschiedenheit wird es wahrscheinlicher, dass es zu Problemen mit Kommunikation und Erwartungen kommt.

Alle drei Arten der Testorganisation müssen auch ihre Methodiken angleichen. Auch wenn es in jedem Projekt vorkommen kann, dass die Methoden nicht oder nicht ausreichend angeglichen werden, so ist dies doch wahrscheinlicher, wenn die Aufgaben verteilt und/oder von externen Einheiten durchgeführt werden. Wenn zwei Testteams verschiedene Methoden anwenden, oder wenn das Testteam eine andere Methode als die Entwicklungsabteilung oder die Projektleitung benutzt, führt dies besonders bei der Testdurchführung zu gravierenden Problemen. Wenn beispielsweise ein Kunde ein agiles Vorgehen einsetzt und der Testdienstleister der eine vordefinierte Testmethodologie verwendet, die einen sequenziellen Lebenszyklus voraussetzt, dann sind Zeitpunkt und Art der Lieferung der Testobjekte an den Testdienstleister mögliche Reibungspunkte.

Beim verteilten Testen an verschiedenen Standorten muss die Aufteilung der Testaufgaben explizit und vorausschauend festgelegt sein. Auch eine kompetente und hoch qualifizierte Gruppe kann ohne solche Vorgaben die Testarbeit nicht erfolgreich durchführen. Wenn nicht jedes Team weiß, wofür es verantwortlich ist, dann wird es möglicherweise nicht das tun, was es tun soll. Ohne sorgfältiges Management wird die Testarbeit Lücken haben (mit steigendem Restrisiko für die Qualität bei Auslieferung) und es kann zu Überlappungen kommen (was die Effizienz verringert).

Letztlich ist für alle drei Arten der Testorganisation entscheidend, dass das ganze Projektteam das Vertrauen entwickelt und behält, dass jedes Testteam seine Rollen richtig ausführt, auch angesichts von organisatorischen, kulturellen, sprachlichen und geographischen Grenzen. Ein Mangel an Vertrauen kann dazu führen, dass durchgeführte Aktivitäten überdurchschnittlich kontrolliert werden, dass es bei Problemen zu gegenseitigen Schuldzuweisungen kommt und dass politische Spielchen betrieben werden. Dabei entstehen Ineffizienzen und mögliche Schwierigkeiten bei der Einhaltung von Plänen.

2.9 Die Anwendung von Industriestandards managen

Sowohl im Foundation Level Lehrplan als auch in den Advanced Level Lehrplänen werden einige Standards genannt. Diese betreffen unterschiedliche Softwarethemen, wie z.B. Softwarelebenszyklus, Softwaretesten, Softwarequalitätsmerkmale, Reviews und Fehlermanagement. Testmanager sollten wissen, welche Standards es gibt, wie das Unternehmen zur Anwendung von Standards steht und ob Standards vorgeschrieben, notwendig oder nützlich sind.

Standards können aus unterschiedlichen Quellen stammen:

- Internationale Standards oder solche mit internationalen Zielsetzungen
- Nationale Standards oder nationale Anwendung internationaler Standards
- Branchenspezifische Standards, wenn also internationale oder nationale Standards für bestimmte Industriezweige angepasst oder speziell für eine Branche entwickelt werden.

Internationale Normungsorganisationen sind ISO und IEEE. ISO ist die Abkürzung für „International Standards Organization“ (auch International Organization for Standardization bzw. ISO genannt). ISO setzt sich zusammen aus Mitgliedern verschiedener nationaler Organisationen, die für die Standardisierung in ihrem Land repräsentativ sind. Die internationale Organisation hat einige hilfreiche Standards und Normen für Softwaretester herausgegeben, wie z.B. ISO 9126 (wird ersetzt durch die ISO 25000 [ISO25000], ISO 12207[ISO 12207] und ISO 15504[ISO 15504].

IEEE ist die Abkürzung für „Institute of Electrical and Electronics Engineers“, einen Berufsverband mit Sitz in den Vereinigten Staaten. Nationale Repräsentanten dieser Organisation gibt es in mehr als hundert Ländern. Das IEEE hat einige hilfreiche Standards und Normen für Softwaretester herausgegeben, wie z.B. IEEE Standard 829 [IEEE 829] und IEEE Standard 1028 [IEEE 1028].

Viele Länder haben eigene spezifische Standards, von denen einige für das Softwaretesten anwendbar und/oder nützlich sind. Ein Beispiel ist der britische Standard BS 7925-2, der Informationen über viele der Testverfahren liefert, die in den Modulen Test Analyst und Technical Test Analyst des Advanced Level Lehrplans behandelt werden.

Standards gibt es auch für verschiedene technische Bereiche und einige davon enthalten relevante Aspekte für Softwaretest, Softwarequalität und -entwicklung. In der Luftfahrtindustrie gilt der branchenspezifische Standard DO-178B (bzw. dessen europäisches Äquivalent ED-12B) für Software in der zivilen Luftfahrt. Der Standard schreibt für Avionik-Software bestimmte strukturelle Abdeckungskriterien vor, je nach Kritikalität der zu testenden Software.

Ein weiteres Beispiel für einen branchenspezifischen Standard betrifft medizinische Systeme. Title 21 CFR Part 820 der U.S. Food and Drug Administration (FDA) [FDA21], also der US-amerikanischen Bundesbehörde zur Überwachung von Nahrungs- und Arzneimitteln, empfiehlt hierin bestimmte strukturelle und funktionale Testverfahren. Darüber hinaus empfiehlt die FDA in diesem Standard Teststrategien und -grundsätze, die mit den ISTQB® Lehrplänen übereinstimmen.

In manchen Fällen wird das Testen von Standards oder weit verbreiteten Methodologien beeinflusst, die sich nicht in erster Linie mit dem Testen befassen, die jedoch großen Einfluss auf den Softwareprozess haben, in dessen Kontext das Testen stattfindet. Ein Beispiel hierfür ist CMMI® (ein Modell zur Softwareprozessverbesserung). Dieses Modell beinhaltet die beiden wichtigen Prozessbereiche Verifizierung und Validierung, die häufig als ein Verweis auf bestimmte Teststufen interpretiert werden (im Speziellen Systemtest bzw. Abnahmetest). Das Modell hat außerdem Auswirkungen bezüglich der Teststrategie, da es oft dahingehend ausgelegt wird, dass analytisches anforderungsbasiertes Testen integraler Bestandteil der Teststrategie sein sollte.

Drei weitere wichtige Beispiele sind PMI's PMBOK, PRINCE2®, sowie ITIL®. PMI und PRINCE2 sind gebräuchliche Projektmanagement-Standards in Nordamerika bzw. in Europa. ITIL ist ein Rahmenwerk, das sicherstellen soll, dass die IT-Abteilung dem Unternehmen, in dem es existiert, wertvolle IT-Dienste liefert. Die Terminologie und Aktivitäten, die in diesen Rahmenwerken spezifiziert werden, unterscheiden sich erheblich von den Lehrplänen und dem Glossar des ISTQB®. Wenn Testmanager in Unternehmen arbeiten, die PMI's PMBOK, PRINCE2 und/oder ITIL verwenden, dann müssen sie die für sie relevanten Rahmenwerke kennen und wissen, wie sie zu implementiert sind. Außerdem müssen sie mit der entsprechenden Terminologie ausreichend vertraut sein, um in diesem Kontext effektiv arbeiten zu können.

Unabhängig davon, welche Standards oder Methodologien angewendet werden, ist in diesem Zusammenhang zu bedenken, dass diese von Expertinnen und Experten erstellt wurden. Als solche spiegeln sie deren kollektives Wissen wider; aber auch deren Schwächen. Testmanager sollten wissen, welche Standards für ihr Testumfeld und ihren Testkontext zutreffen. Das können formale Standards sein (international, national oder branchenspezifisch), oder unternehmensinterne Standards und empfohlene Praktiken.

Wer in Betracht zieht, mehrere Standards gleichzeitig zu verwenden, sollte beachten, dass manche Standards inkonsistent zu anderen sein könnten oder gar gegensätzliche Definitionen mit sich bringen. Der Testmanager muss daher jeweils die Nützlichkeit der verschiedenen Standards für seinen spezifischen Kontext, also dem Testen, bewerten. Die Vorgaben eines Standards können für ein Projekt zwar sehr nützlich, aber auch genaus gut hinderlich sein. Dennoch bieten Standards eine gute Referenz zu nachweislicher Best Practice und eine Basis zur Gestaltung des Testprozesses.

In manchen Fällen ist die Einhaltung von Standards verpflichtend und hat Auswirkungen auf das Testen. Testmanager müssen wissen, welche Standards eingehalten werden müssen und sie müssen auf angemessene Konformität achten.

3. Reviews – [180 Minuten]

Begriffe

Audit, informelles Review, Inspektion, Managementreview, Moderator, Gutachter, Review, Reviewplan, technisches Review, Walkthrough

Lernziele für Reviews

3.2 Managementreviews und Audits

TM-3.2.1 (K2) Sie können die Schlüsselmerkmale von Managementreviews und Audits erläutern

3.3 Management von Reviews

TM-3.3.1 (K4) Sie können ein Projekt analysieren, um die geeignete Review-Art auszuwählen und einen Plan für die Durchführung von Reviews definieren, um die korrekte Durchführung, Nachverfolgung und Nachvollziehbarkeit sicherzustellen

TM-3.3.2 (K2) Sie können erläutern, welche Bedeutung die Qualifikation und der erforderliche Zeitaufwand für die Teilnahme an Reviews haben

3.4 Metriken für Reviews

TM-3.4.1 (K3) Sie können Prozess- und Produktmetriken definieren, die in Reviews verwendet werden

3.5 Management von formalen Reviews

TM-3.5.1 (K2) Sie können anhand von Beispielen die Merkmale eines formalen Reviews erklären

3.1 Einführung

Reviews wurden im ISTQB® Foundation Level Lehrplan als ein statisches Testverfahren für Produkte eingeführt. Audits und Managementreviews sind mehr auf den Softwareprozess fokussiert als auf die Arbeitsergebnisse.

Da Reviews eine Ausprägung des Statischen Tests sind, sind Testmanager nicht selten für den Gesamterfolg von Reviews verantwortlich, zumindest wenn es um Ergebnisse aus dem Test geht. Mit Blick auf alle Softwareprojekte sollte die Festlegung der Zuständigkeit aber durch die allgemeine Strategie der Organisation bzw. des Unternehmens grundsätzlich abgedeckt sein. Bedenkt man aber, dass formelle Reviews in vielen Bereichen eingesetzt werden, also durchaus schon bevor es zu einem Softwareprojekt kommt, so kann diese Zuständigkeit bei einem Testmanager, bei einem Qualitätssicherungsbeauftragten oder sogar bei einem ausgebildeten Review-Koordinator liegen. In diesem Lehrplan wird die verantwortliche Person (wer immer dies auch sein mag) als Review-Leiter bezeichnet.

Der Review-Leiter sollte sicherstellen, dass ein Umfeld geschaffen wird, welches für die im ISTQB® Foundation Level Lehrplan definierten Erfolgsfaktoren förderlich ist. Außerdem sollte der Review-Leiter einen Plan zur Messung von Reviews ausarbeiten, um sicherzustellen, dass die Reviews einen spürbaren Mehrwert liefern.

Da Tester ein tiefes Verständnis für das Systemverhalten und die benötigten Merkmale des Softwaresystems mitbringen, sollten sie am Review-Prozess beteiligt sein.

Teilnehmer an Reviews sollten für Reviews geschult sein, um die jeweiligen Rollen im Review-Prozess besser zu verstehen. Alle Review-Teilnehmer müssen sich engagiert für den Nutzen eines gut geführten Reviews einsetzen.

Wenn sie richtig durchgeführt werden, leisten Reviews nicht nur den größten einzelnen, sondern auch den kosteneffektivsten Beitrag zur gelieferten Qualität. Es ist daher von höchster Wichtigkeit, dass Review-Leiter in den Projekten effiziente Reviews durchführen und deren Nutzen aufzeigen können.

Folgende Reviews sind in einem Projekt möglich:

- Review des Vertrages, durchgeführt zu Projektbeginn und bei wichtigen Projektmeilensteinen
- Review der Anforderungen, durchgeführt, wenn die Anforderungen für ein Review verfügbar sind, idealerweise werden funktionale und nicht-funktionale Anforderungen abgedeckt
- Review des Systementwurfs, durchgeführt wenn der übergreifende Architekturentwurf für ein Review verfügbar ist
- Review des detaillierten Entwurfs, durchgeführt wenn der detaillierte Entwurf für ein Review verfügbar ist
- Code-Reviews, durchgeführt, wenn einzelne Softwaremodule erstellt werden, dies kann die Modultests und deren Ergebnisse sowie den Code selbst einschließen
- Review von Testarbeitsprodukten, dies kann Testkonzept(e), Testbedingungen, Ergebnisse der Qualitätsrisikoanalyse, Tests, Testdaten, Testumgebungen und Testergebnisse einschließen
- Testeingangs-Review (Review der Testbereitschaft) und Testausgangs-Review für jede Teststufe; es werden die Testeingangskriterien vor Beginn der Testausführung beziehungsweise die Endekriterien vor Testabschluss geprüft
- Abnahme-Reviews, werden durchgeführt, um die Genehmigung des Systems durch Kunden oder Stakeholder herbeizuführen

Zusätzlich zur Anwendung mehrerer Review-Arten auf ein Produkt sollten Review-Leiter beachten, dass Reviews – auch wenn sie Fehlerzustände in statischen Dokumenten aufdecken können – mit anderen Arten des statischen Testens (z.B. statische Analyse) und mit dem dynamischen Test des Codes erweitert werden sollten. Mit einer Kombination dieser Verfahren wird die Testüberdeckung erhöht und es werden mehr Fehlerzustände gefunden.

Unterschiedliche Verfahren setzen unterschiedliche Schwerpunkte. Ein Review kann beispielsweise ein Problem in der Anforderungsphase beseitigen, noch bevor das Problem im Code implementiert wird. Die statische Analyse kann helfen, Programmierkonventionen durchzusetzen und Probleme zu prüfen, die für das Team eventuell zu mühsam sind, um sie durch anderweitige Prüfung der Arbeitsergebnisses aufzudecken. Inspektionen können nicht nur zu einer Entdeckung und Beseitigung von Fehlerzuständen führen, sondern können Autoren auch darin schulen, wie Fehlerzustände in den Arbeitsergebnissen vermieden werden können.

Im Foundation Level Lehrplan wurden folgende Review-Arten eingeführt:

- Informelles Review
- Walkthrough
- Technisches Review
- Inspektion

Zusätzlich zu den im Foundation Level Lehrplan eingeführten Review-Arten können Testmanager auch an folgenden Review-Arten beteiligt sein:

- Managementreview
- Audit

3.2 Managementreviews und Audits

Managementreviews werden eingesetzt, um den Fortschritt zu überwachen, den Status zu beurteilen und um Entscheidungen über zukünftige Maßnahmen zu treffen. Diese Reviews unterstützen Entscheidungen über die Zukunft des Projekts, wie z.B. Anpassung der eingesetzten Ressourcen, Ergreifen korrigierender Maßnahmen oder Änderung des Projektumfangs.

Schlüsselmerkmale eines Managementreviews sind:

- Es wird von Managern bzw. für Manager mit direkter Verantwortung für das Projekt oder System durchgeführt.
- Durchgeführt von bzw. für einen Stakeholder oder Entscheidungsträger, beispielsweise mittleres oder gehobenes Management.
- Es wird geprüft, ob Pläne konsistent eingehalten werden oder ob es Abweichungen gibt.
- Es wird festgestellt, ob die eingesetzten Managementverfahren adäquat sind.
- Es enthält die Bewertung von Projektrisiken.
- Es werden die Auswirkungen von Maßnahmen und Möglichkeiten zur Messung dieser Auswirkungen bewertet.
- Es werden Listen mit durchzuführenden Maßnahmen, zu lösenden Problemen und anstehenden Entscheidungen erstellt.

Managementreviews von Prozessen, wie z.B. Bewertungssitzungen (“lessons learned“), sind integraler Bestandteil der Prozessverbesserungsaktivitäten.

Testmanager sollten an Managementreviews zum Testfortschritt teilnehmen und diese möglicherweise überhaupt auch veranlassen.

Audits werden in der Regel dann durchgeführt, wenn Konformität mit bestimmten Kriterien nachgewiesen werden soll, beispielsweise Konformität mit einem geltenden Standard, Vorschriften oder einer vertraglichen Verpflichtung. Der Hauptzweck von Audits ist daher die unabhängige Bewertung der Konformität mit bestimmten Verfahren, Vorschriften, Standards, usw.

Schlüsselmerkmale eines Audits:

- Der Leiter des Audits ist für das Audit verantwortlich und übernimmt die Moderation.
- Nachweise für die Konformität werden durch Interviews, Beobachtungen und die Prüfung von Dokumenten gesammelt.
- Zu den dokumentierten Ergebnissen des Audits gehören Beobachtungen, Empfehlungen, Abhilfemaßnahmen und eine abschließende Beurteilung (bestanden/nicht bestanden).

3.3 Management von Reviews

Reviews sollten für Übergänge oder Meilensteine im Softwareprojekt eingeplant werden. Typischerweise sollten Reviews erfolgen, wenn Anforderungs- und Entwurfsspezifikationen vorliegen. Reviews in diesem Zusammenhang beginnen bei den geschäftlichen Zielen und arbeiten sich vor bis zum detaillierten Entwurf. Management Reviews sollten an den wichtigsten Meilensteinen des Projekts stattfinden, als Teil der Verifizierungsaktivitäten vor, während und nach der Testausführung, sowie in anderen wichtigen Projektphasen. Die Reviewstrategie muss auf die Testrichtlinie und auf die gesamte Teststrategie abgestimmt sein.

Vor Ausarbeitung eines Reviewplans auf Projektebene sollte ein Review-Leiter folgende Faktoren berücksichtigen:

- Was sollte durch Reviews geprüft werden (Produkt und Prozesse)?
- Wer sollte an bestimmten Reviews mitwirken?
- Welche relevanten Risikofaktoren sind abzudecken?

In der frühen Projektplanungsphase sollte der Review-Leiter die Review-Gegenstände identifizieren und die geeignete Review-Art (informelles Review, Walkthrough, technisches Review oder Inspektion, oder eine Mischung aus zwei oder drei Arten) und Grad der Formalität auswählen. An dieser Stelle könnten zusätzliche Review-Schulungen empfohlen werden und es kann dem Reviewprozess ein Budget (Zeit und Ressourcen) zugewiesen werden. In die Budgetierung sollten eine Risikobewertung und eine Rentabilitätsberechnung mit einfließen.

Der Return-on-Investment für Reviews besteht aus der Differenz zwischen den Durchführungskosten eines Reviews und den möglichen Kosten, die später für die Kompensation der gefundenen Fehlerzustände – hätte man sie nicht schon im Review gefunden – (oder dafür dass man sie komplett übersehen hätte) entstanden wären. Die Berechnung der Kosten der Qualität, die in Abschnitt 2.7 genauer erklärt wird, kann bei der Bestimmung dieser Größe verwendet werden.

Die Bestimmung des optimalen Zeitpunkts für die Durchführung von Reviews hängt von folgenden Faktoren ab:

- Verfügbarkeit der Review-Gegenstände in einem hinreichend reviewfähigem Zustand
- Verfügbarkeit des für das Review benötigten Personals
- Termin, zu dem die endgültige Version der Review-Gegenstände verfügbar sein sollte
- Benötigte Zeit für das Review des betreffenden Review-Gegenstandes.

Geeignete Metriken für die Review-Bewertung sollten vom Review-Leiter bei der Testplanung definiert werden. Falls Inspektionen vorgesehen sind, sollte es auf Wunsch des Autors auch möglich sein, Kurzinspektionen auf Teile von Dokumentendurchzuführen, sobald diese fertig sind (z.B. einzelne Anforderungen oder Teile davon).

Die Ziele des Reviewprozesses müssen in der Testplanungsphase definiert werden. Diese beinhalten die Durchführung effektiver und effizienter Reviews und dass in Zusammenhang mit dem Feedback aus den Reviews ein Konsens für Entscheidungen erzielt wird.

Projekt-Reviews für das Gesamtsystem werden häufig abgehalten; sie können auch für Teilsysteme oder sogar für einzelne Softwareelemente erforderlich sein. Anzahl, Art, Organisation und beteiligte Personen der Reviews hängen ab von der Größe und Komplexität des Projekts, sowie auch von den Produktrisiken.

Damit Reviews effizient sein können, müssen die Teilnehmer das entsprechende Wissen mitbringen, sowohl in technischer Hinsicht als auch bezüglich des Verfahrens. Sorgfalt und ein Auge fürs Detail sind nur einige der Fähigkeiten, die Gutachterfür effektive Reviews haben sollten. Klarheit und korrekte Priorisierung sind Eigenschaften, durch die sich gute Review-Bemerkungen auszeichnen. Das benötigte Wissen über Verfahren und Abläufe kann bedeuten, dass Schulungen erforderlich sind, um sicherzustellen, dass die Prüfer ihre Rollen und Verantwortlichkeiten im Review-Prozess verstehen.

In der Reviewplanung sollten Risiken in Zusammenhang mit den technischen, organisatorischen und personenbezogenen Faktoren bei der Durchführung von Reviews berücksichtigt werden. Die Verfügbarkeit von Gutachtern mit ausreichend technischem Wissen ist ein kritischer Erfolgsfaktor eines Reviews. Sämtliche Projektteams sollten in die Planung der Reviews involviert werden, damit sichergestellt ist, dass jedes Team den Erfolg des Reviewprozesses mit Überzeugung und Engagement unterstützt. Bei der Planung muss sichergestellt werden, dass jede einzelne Organisation oder jedes einzelne Unternehmen ausreichend Zeit einplant, damit die benötigten Gutachter die Reviews zu den im Projektplan vorgesehenen Terminen vorbereiten und daran teilnehmen können. Es sollte, sofern erforderlich, auch Zeit für technische oder prozessbezogene Schulungen der Gutachter eingeplant werden. Außerdem sollen Personen identifiziert werden, die einspringen können, falls ein wichtiger Gutachter aus persönlichen oder dienstlichen Gründen verhindert ist.

Bei der Durchführung des formalen Reviews muss der Review-Leiter sicherstellen, dass

- von den Review-Teilnehmern geeignete Messungen geliefert werden, die eine Bewertung der Effizienz des Reviews erlauben
- Checklisten erstellt werden und dass diese für zukünftige Reviews aktuell gehalten werden
- Schwere und Priorität der Fehlerzustände definiert werden, damit diese im Fehlermanagement von Problemen verwendet werden können, die im Review gefunden wurden (siehe Kapitel 4)

Nach jedem Review sollte der Review-Leiter folgende Aufgaben durchführen:

- Die Review-Metriken sammeln und sicherstellen, dass die identifizierten Probleme ausreichend behoben sind, um die spezifischen Testziele des Reviews zu erreichen
- Die Review-Metriken als Grundlage für die Bestimmung der Rentabilität des Reviews verwenden
- Rückmeldungen und Informationen an die relevanten Stakeholder geben
- Rückmeldungen an die Review-Teilnehmer geben.

Um die Effektivität von Reviews zu bewerten, können Testmanager die tatsächlichen Ergebnisse, die im anschließenden Testen gefunden wurden (also nach dem Review) mit den Ergebnissen aus den Review-Berichten vergleichen. Falls ein Arbeitsergebnis aufgrund eines erfolgreichen Reviews genehmigt wurde, sich später dennoch als fehlerhaft erweist, dann sollte der Review-Leiter sich Gedanken darüber machen, wie die Fehlerzustände im Review-Prozess übersehen werden konnten. Zu den möglichen Ursachen gehören Probleme mit dem Review-Prozess (z.B. schlechte Eingangs-/Endekriterien), falsche Zusammensetzung des Review-Teams, ungeeignete Review-Mittel (z.B.

Checklisten, usw.), unzureichende Schulung und Erfahrung der Gutachter, sowie zu wenig Zeit für Review-Vorbereitung und Review-Sitzung(en).

Ergibt sich ein Muster nicht gefundener Fehlerzustände (insbesondere mit hohen Schweregraden), das sich über mehrere Projekte wiederholt, dann ist dies ein Anzeichen dafür, dass es mit der Durchführung von Reviews erhebliche Probleme gibt. In einer solchen Situation muss der Review-Leiter den Prozess überprüfen und entsprechende Maßnahmen ergreifen. Es ist auch möglich, dass Reviews aus verschiedenen Gründen im Laufe der Zeit an Effektivität einbüßen. Solche Wirkungen werden in Bewertungssitzungen in Form einer reduzierten Fehlerfindungsrate offensichtlich. Auch hier muss der Review-Leiter die Ursachen untersuchen und beheben. Auf keinen Fall sollten Review-Metriken dazu dienen, einzelne Gutachter oder Autoren zu belohnen oder abzustrafen, sondern sollten sich vielmehr auf den Reviewprozess an sich fokussiert sein.

3.4 Metriken für Reviews

Review-Leiter (die, wie bereits erwähnt, Testmanager sein können) müssen sicherstellen, dass die Metriken für folgende Zwecke zur Verfügung stehen:

- Bewerten der Qualität des Review-Gegenstands
- Bewerten der Kosten für Durchführung des Reviews
- Bewerten des späteren Nutzens aufgrund der Durchführung des Reviews

Review-Leiter können die Ergebnisse der Messungen dazu verwenden, die Rentabilität und Effizienz der Reviews zu bestimmen. Diese Metriken können außerdem für die Berichterstattung und für Prozessverbesserungsaktivitäten verwendet werden.

Für jedes geprüfte Arbeitsergebnis lassen sich die folgenden Metriken für die Produktbewertung messen und berichten:

- Umfang des Arbeitsergebnisses (Seitenzahl, Anzahl der Codezeilen, usw.)
- Vorbereitungszeit (vor dem Review)
- Zeit für Durchführung des Reviews
- benötigte Zeit zur Fehlerbehebung
- Dauer des Review-Prozesses
- Anzahl der aufgedeckten Fehlerzustände
- Identifizierung von Fehlerhäufungen im Arbeitsergebnis (d.h. Bereiche mit höherer Fehlerdichte)
- Review-Art (informelles Review, Walkthrough, technisches Review oder Inspektion)
- Durchschnittliche Fehlerdichte (d.h. Fehlerzustände pro Seite oder je tausend Codezeilen)
- Geschätzte Anzahl verbleibender Fehlerzustände (oder verbleibende Fehlerdichte)

Für jedes Review lassen sich die folgenden Metriken für die Prozessbewertung messen und berichten:

- Fehlerfindungsrate (unter Berücksichtigung von Fehlerzuständen, die später im Lebenszyklus gefunden wurden)
- Verbesserung des Aufwands für Reviews und der Termintreue
- prozentuale Überdeckung der geplanten Arbeitsergebnisse
- Arten der gefundenen Fehlerzustände und deren Schweregrad
- Teilnehmerbefragungen über Effektivität und Effizienz des Review-Prozesses
- Vergleich der Metriken zu den Kosten von Qualität für gefundene Fehlerzustände im Review, mit Fehlerzuständen, die im dynamischen Test bzw. in der Produktion (Regelbetrieb) gefunden wurden
- Korrelation der Review-Effektivität (Review-Art gegen Fehlerfindungseffektivität)

- Anzahl von Gutachtern
- aufgedeckte Fehlerzustände pro aufgewendete Review-Stunde
- Schätzung zur eingesparten Projektzeit
- durchschnittlicher Fehleraufwand (d.h. Gesamtzeitaufwand für Fehleraufdeckung und Behebung dividiert durch Anzahl der Fehlerzustände)

Außerdem sind die für die Produktbewertung erwähnten Metriken auch bei der Prozessbewertung nützlich.

3.5 Management von formalen Reviews

Im ISTQB® Foundation Level Lehrplan werden die Phasen eines formalen Reviews beschrieben: Planung, Kick-off, individuelle Vorbereitung, Review-Sitzung, Überarbeitung und Nachbereitung. Um formale Reviews korrekt durchzuführen, müssen Review-Leiter sicherstellen, dass alle Schritte des Review-Prozesses befolgt werden.

Zu den Merkmalen formaler Reviews gehören:

- Definierte Eingangs- und Testausgangskriterien/Endekriterien
- Checklisten, die die Gutachter verwenden können
- Arbeitsergebnisse, wie z.B. Berichte, Bewertungsbögen oder sonstige Review-Zusammenfassungen
- Metriken zum Berichten von Effektivität, Effizienz und Fortschritt des Reviews

Vor Einleitung eines formalen Reviews sollte vom Review-Leiter bestätigt werden, dass die Voraussetzungen (definiert in der Vorgehensweise oder in der Liste der Eingangskriterien) für ein Review erfüllt sind.

Wenn die Voraussetzungen für ein formales Review nicht erfüllt sind, kann der Review-Leiter der zuständigen Stelle eine der folgenden Optionen zur Entscheidung vorschlagen:

- Neudefinition des Reviews mit überarbeiteten Zielen
- Korrigierende Maßnahmen, die notwendig sind, damit das Review fortgesetzt werden kann
- Verschiebung des Reviews

Teil der Steuerung eines formalen Reviews ist es, dass diese im Kontext des gesamten (übergeordneten) Programms überwacht werden und mit den Qualitätssicherungsaktivitäten des Projekts in Zusammenhang stehen. Zur Steuerung von formalen Reviews gehören auch die Informationen aus Rückmeldungen unter Verwendung der Produkt- und Prozessmetriken.

4. Fehlermanagement – [150 Minuten]

Begriffe

Anomalie, falsch positives Ergebnis, falsch negatives Ergebnis, Fehlereindämmung innerhalb der Phase, Fehlermanagement-Ausschuss, Fehlerwirkung, Fehlerzustand, Grundursache, Priorität, Schweregrad

Lernziele für das Fehler- und Abweichungsmanagement

4.2 Fehlerlebenszyklus und Softwarelebenszyklus

TM-4.2.1 (K3) Sie können einen Fehlermanagementprozess für eine Testorganisation entwickeln, einschließlich der Abläufe für Fehlerberichte, die zur Überwachung und Steuerung der Fehler in einem Projekt im gesamten Testlebenszyklus verwendet werden können

TM-4.2.2 (K2) Sie können erläutern, welcher Prozess und welche Teilnehmer für ein effektives Fehlermanagement erforderlich sind

4.3 Informationen im Fehlerbericht

TM-4.3.1 (K3) Sie können die Daten und Informationen über Fehlerklassen definieren, die im Fehlermanagementprozess gesammelt werden sollten

4.4 Bewerten der Prozessreife anhand von Fehlerberichten

TM-4.4.1 (K2) Sie können erklären, wie Fehlerberichtsstatistiken dazu verwendet werden können, um die Prozessreife des Test- und des Softwareentwicklungsprozesses zu bewerten

4.1 Einführung

Der Fehlermanagementprozess eines Unternehmens und das verwendete Fehlermanagementwerkzeug sind von entscheidender Bedeutung, nicht nur für das Testteam, sondern für alle Teams, die an der Softwareentwicklung beteiligt sind. Die Daten und Informationen, die durch ein effektives Fehlermanagement gesammelt werden, geben dem Testmanager und den Stakeholdern Aufschluss über den Zustand eines Projektes während des ganzen Entwicklungslebenszyklus. Werden die Daten über einen längeren Zeitraum hinweg gesammelt und analysiert, dann können sie dazu beitragen, die Bereiche zu bestimmen, in denen der Test- und Entwicklungsprozess verbessert werden kann.

Zusätzlich zum Verständnis des kompletten Fehlerlebenszyklusses und dessen, wie er zur Überwachung und Steuerung von Test und vom Softwarelebenszyklus eingesetzt wird, muss der Testmanager wissen, welche Informationen erhoben werden müssen und er muss den richtigen Einsatz des Fehlermanagementprozesses und des verwendeten Fehlermanagementwerkzeuges überzeugend vertreten.

4.2 Fehlerlebenszyklus und Softwarelebenszyklus

Wie im Foundation Level Lehrplan erklärt, entstehen Fehlerzustände, wenn eine Person beim Erstellen eines Arbeitsergebnisses einen Fehler macht. Beim Arbeitsergebnis kann es sich um eine Anforderungsspezifikation, eine User Story, ein technisches Dokument, einen Testfall, den Programmcode, oder um irgendein anderes Arbeitsergebnis handeln, das während der Softwareentwicklungs- oder Wartungsprozesse erstellt wird.

Fehlerzustände können an jeglicher Stelle im Softwarelebenszyklus und in jedem software-bezogenen Arbeitsergebnis entstehen, die mit der Software in Zusammenhang steht. Daher sollte jede Phase des Entwicklungslebenszyklus Aktivitäten zum Erkennen und Beseitigen möglicher Fehlerwirkungen beinhalten. Statische Testverfahren, wie Reviews und die statische Analyse, können beispielsweise für Entwurfsspezifikationen, Anforderungsspezifikationen und den Code durchgeführt werden, bevor diese Arbeitsergebnisse an nachfolgende Aktivitäten weitergegeben werden. Je früher ein Fehlerzustand erkannt und korrigiert wird, desto geringer sind die Kosten der Qualität für das Gesamtsystem; die Kosten der Qualität für die Fehlerzustände eines bestimmten Schweregrades werden minimiert, wenn jeder Fehlerzustand jeweils noch in der Phase korrigiert wird, in der er entstanden ist (d.h.: wenn der Softwareprozess die vollständige Fehlereindämmung innerhalb der Phase erreicht). Des Weiteren, wie im Foundation Level Lehrplan erklärt, findet das statische Testen Fehlerzustände direkt und deckt nicht Fehlerwirkungen auf. Aus diesem Grund sind auch die Kosten für die Fehlerbeseitigung niedriger, da keine Debugging-Aktivitäten anfallen, um die Fehlerzustände zu lokalisieren.

Bei dynamischen Testaktivitäten, wie z.B. Modultests, Integrationstests und Systemtests, wird das Vorhandensein eines Fehlerzustands aufgedeckt, wenn dieser eine Fehlerwirkung verursacht und eine Diskrepanz zwischen den tatsächlichen und den erwarteten Testergebnissen (d.h. eine Anomalie) erkannt wird. Manchmal kommt es zu einem falsch negativen Ergebnis, wenn ein Tester die Anomalie nicht bemerkt. Wenn der Tester die Anomalie jedoch bemerkt, ist eine Situation eingetreten, die genauer untersucht werden muss. Diese Untersuchung beginnt mit der Aufnahme eines Fehlerberichts.

Bei der testgetriebenen Entwicklung werden ausführbare Entwurfsspezifikationen für automatisierte Komponententests verwendet. Der Produktionscode wird entwickelt und dann sofort mit diesen Tests ausgeführt. Solange die Entwicklung des Code noch nicht abgeschlossen ist, werden einige der Tests

nicht erfolgreich bestanden. In einem solchen Fall zählt dies nicht als Fehler und wird normalerweise auch nicht verfolgt.

4.2.1 Fehler-Workflow und Status von Fehlerzuständen

Die meisten Testorganisationen verwenden für das Management der Fehlerberichte im gesamten Fehlerlebenszyklus ein entsprechendes Werkzeug (Fehler-Workflow-Werkzeug). Für Fehlerberichte gibt es einen typischen Ablauf, in dem eine Reihe von Zuständen (Status) im Fehlerlebenszyklus durchlaufen wird. Für die meisten dieser Zustände gibt es einen klar Zuständigen, der für die Durchführung der erforderlichen Maßnahme verantwortlich ist und der – sobald vollendet – den Übergang in den nächsten Status veranlasst (womit der Fehlerbericht in die Verantwortung des nachfolgend Zuständigen übergeht). Im Endstatus hat der Fehlerbericht keinen Eigentümer, da keine weiteren Maßnahmen mehr erforderlich sind. In seinem Endstatus ist der Fehlerbericht:

- geschlossen (dies bedeutet in der Regel, dass der zugrunde liegende Fehlerzustand behoben und durch einen Fehlernachtest verifiziert ist),
- storniert (dies bedeutet in der Regel, dass der Fehlerbericht ungültig ist),
- nicht reproduzierbar (dies bedeutet in der Regel, die Anomalie nicht mehr beobachtet werden kann), oder
- zurückgestellt (dies bedeutet in der Regel, dass die Anomalie einen echten Fehlerzustand betrifft, aber dass dieser nicht während des Projekts behoben wird).

Bei Fehlerzuständen, die Tester im Test gefunden haben, gibt es typischerweise drei Status im Verantwortungsbereich des Testteams:

- Der Anfangsstatus
  - In diesem Status sammeln ein oder mehrere Tester die Informationen, damit die zuständige Person die Fehlewirkung reproduzieren kann (mehr über Informationen, die im Fehlerbericht enthalten sein sollten, siehe Abschnitt 4.3).
  - Dies kann auch als „offener“ oder „neuer“ Status bezeichnet werden.
- Der zurückgewiesene Status
  - In diesem Status hat der Empfänger des Berichts diesen zurückgewiesen, oder die Tester um mehr Informationen gebeten. Dieser Status kann auf Defizite, entweder in Zusammenhang mit der anfänglichen Informationssammlung oder mit dem Testen, hinweisen. Testmanager sollten überwachen, ob es zu übermäßig vielen Rückläufen kommt. Tester müssen entweder die zusätzlichen Informationen liefern oder bestätigen, dass der Bericht tatsächlich zurückgewiesen werden sollte.
  - Dies kann auch als „abgelehnter“ oder „klärungsbedürftiger“ Status bezeichnet werden.
- Der Fehlernachtest-Status
  - In diesem Status führt der Tester einen Fehlernachtest durch (oft hält er sich beim Reproduzieren der Fehlerwirkung an die im Fehlerbericht beschriebenen Schritte) und bestimmt dadurch, ob das Problem tatsächlich gelöst ist. Wenn der Fehlernachtest zeigt, dass der Fehlerzustand behoben ist, sollte der Tester den Bericht schließen. Wenn der Fehlernachtest ergibt, dass der Fehlerzustand nicht behoben ist, sollte der Tester den Bericht wieder öffnen. Dadurch geht der Bericht wieder an den früheren Eigentümer, der die notwendigen Arbeiten abschließen kann, um den Fehlerzustand zu reparieren.
  - Dies kann auch als „gelöster“ oder „Verifizierungs-“ Status bezeichnet werden.

4.2.2 Ungültige und doppelte Fehler managen

Manchmal tritt eine Anomalie nicht als Auswirkung eines Fehlerzustands auf, sondern als Folge eines Problems mit der Testumgebung, den Testdaten, irgendeinem Artefakt oder Testmittel, oder weil der Tester etwas missverstanden hat. Wenn der Tester einen neuen Fehlerbericht erstellt und sich anschließend herausstellt, dass sich der Fehlerbericht nicht auf einen Fehlerzustand in einem zu

testenden Arbeitsergebnis bezieht, dann handelt es sich um ein falsch positives Ergebnis. Solche Berichte werden normalerweise als ungültige Fehlerberichte storniert oder geschlossen. Außerdem kann es manchmal vorkommen, dass ein Fehlerzustand unterschiedliche Wirkungen aufweist, die für die Tester anscheinend in keinem Bezug zueinander stehen. Wenn zwei oder mehr Fehlerberichte erstellt werden und sich anschließend herausstellt, dass diese dieselbe Grundursache betreffen, wird in der Regel einer der Fehlerberichte behalten und die anderen werden als doppelte Fehlerberichte auf den ersten Fehlerbericht referenziert und geschlossen.

Ungültige und doppelte Fehlerberichte sind zwar ein Indikator für ein gewisses Maß an Ineffizienz, sie lassen sich jedoch nicht gänzlich vermeiden und sollten in diesem Sinne auch vom Testmanager akzeptiert werden. Wenn Manager versuchen, ungültige und doppelte Fehlerberichte komplett auszuschließen, erhöht sich dadurch typischerweise die Zahl der falsch negative Ergebnisse, da Tester davon abgehalten werden, Fehlerberichte zu erstellen. Dadurch verringert sich die Effektivität des Unternehmens bei der Fehlerfindung und diese ist wiederum meistens ein Hauptziel der Testorganisation.

4.2.3 Übergreifendes Fehlermanagement

Obwohl Testorganisation und Testmanager normalerweise Eigentümer des gesamten Fehlermanagementprozesses und des Fehlermanagementwerkzeugs sind, ist im Allgemeinen ein übergreifendes Team für das Management der berichteten Fehlerzustände im Projekt zuständig. Teilnehmer im Fehlermanagement-Ausschuss (oder im Fehler-Triage-Ausschuss) sind neben dem Testmanager Stakeholder aus Entwicklung, Projektmanagement, Produktmanagement und sonstige Stakeholder, die ein Interesse an der Software haben, die entwickelt wird.

Der Fehlermanagement-Ausschuss sollte regelmäßig oder in einzelnen kritischen Fehlerfällen sofort zusammentreffen und bestimmen, ob der Fehlerbericht auf Fehlerzustände hinweist und ob diese behoben oder zurückgestellt werden müssen. Bei dieser Entscheidung muss der Fehlermanagement-Ausschuss Nutzen, Risiken und Kosten in Zusammenhang mit der Beseitigung bzw. der Nichtbeseitigung der Fehlerzustände berücksichtigen. Falls ein Fehlerzustand behoben werden soll, dann muss das Team die Priorität für die Beseitigung des Fehlerzustands in Bezug auf andere Projektaufgaben bestimmen. Testmanager und Testteam können zur relativen Wichtigkeit eines Fehlerzustands befragt werden und sollten objektive Informationen zur Verfügung stellen.

Ein Fehlerverfolgungswerkzeug sollte nicht als ein Ersatz für gute Kommunikation eingesetzt werden und umgekehrt sollten auch die Besprechungen des Fehlermanagement-Ausschusses nicht ein Ersatz sein für die effektive Verwendung eines guten Fehlerverfolgungswerkzeugs. Kommunikation, geeignete Werkzeugunterstützung und ein engagierter Fehlermanagement-Ausschuss sind allesamt für ein effektives und effizientes Fehlermanagement unerlässlich.

4.3 Informationen im Fehlerbericht bzw. Fehlermeldung

Wenn (im statischen Test) ein Fehlerzustand entdeckt wird, oder wenn (im dynamischen Test) eine Fehlerwirkung beobachtet wird, sollten durch den/die Zuständigen Daten eingeholt und in den Fehlerbericht aufgenommen werden. Diese Informationen dienen drei Zwecken:

- Management des Fehlerberichts im gesamten Fehlerlebenszyklus
- Bewertung des Projektstatus, besonders hinsichtlich Produktqualität und Testfortschritt
- Bewertung der Prozessreife (wie in Abschnitt 4.4 beschrieben).

Die für das Management des Fehlerberichts und Bewertung des Projektstatus benötigten Daten können variieren, je nachdem wann der Fehlerzustand im Lebenszyklus aufgedeckt wurde. Normalerweise wird in frühen Phasen (z.B. Anforderungs-Review und Modultest) weniger Information benötigt. Allerdings sollten die wesentlichen Informationen, die im Laufe des Lebenszyklus gesammelt

werden, einheitlich sein. Idealerweise sollten sie sogar projektübergreifend konsistent sein, um einen aussagekräftigen Vergleich von Fehlerdaten aller Projekte zu ermöglichen.

Das Sammeln von Daten kann für Testfortschrittsüberwachung, Teststeuerung und Bewertung der Endekriterien hilfreich sein. Beispielsweise sollten die Fehlerinformationen die Analyse der Fehlerdichte, die Trendanalyse der gefundenen und behobenen Fehlerzustände, den durchschnittlichen Zeitaufwand von Fehlerfindung bis Fehlerbehebung, sowie die Zeitspanne zwischen Ausfällen (z.B. für die Berechnung der „Mean Time Between Failures“) unterstützen.

Zu den gesammelten Fehlerdaten gehören:

- Name der Person, die den Fehlerzustand aufgedeckt hat
- Rolle dieser Person (z.B. Endnutzer, Fachanalytiker, Entwickler, Technischer Support)
- Ausgeführte Testart (z.B. Benutzbarkeitstest, Regressionstest)
- Zusammenfassung des Problems
- eine detaillierte Beschreibung des Problems
- Schritte zum Reproduzieren der Wirkung eines Fehlerzustands, zusammen mit den tatsächlichen und erwarteten Ergebnissen, die im Fehlerbericht hervorgehoben sind, sowie, falls zutreffend, Screenshots, Datenbank-Dumps und Protokolle
- die Lebenszyklusphase, in der der Fehlerzustand entstanden, aufgedeckt und beseitigt wurde, ggf. einschließlich der Teststufe
- das Arbeitsergebnis, in dem der Fehlerzustand entstanden ist
- Schweregrad der Auswirkungen auf System und/oder Produkt-Stakeholder (wird in der Regel durch das technische Systemverhalten bestimmt)
- Priorität der Fehlerbeseitigung (wird in der Regel durch die geschäftlichen Auswirkungen der Fehlerwirkung bestimmt)
- Teilsystem oder Komponente, in der sich der Fehlerzustand befindet (für eine Fehlerhäufungsanalyse)
- die Projektaktivität zum Zeitpunkt der Fehleraufdeckung
- die Identifizierungsmethode, die zum Aufdecken des Fehlerzustands geführt hat (z.B. Review, statische Analyse, dynamischer Test, Einsatz in der Produktion bzw. im Regelbetrieb)
- Art des Fehlerzustands (die Klassifizierung entspricht in der Regel der Fehlertaxonomie, falls eine solche verwendet wird)
- das vom Fehlerzustand betroffene Qualitätsmerkmal
- die Testumgebung, in der die Fehlerwirkung beobachtet wurde (beim dynamischen Testen)
- Projekt und Produkt, in denen das Problem vorliegt
- derzeitiger Eigentümer, d.h. die Person, die für die Bearbeitung des Problems zuständig ist, falls der Fehlerbericht nicht geschlossen ist
- derzeitiger Status des Fehlerberichts (dies erfolgt in der Regel durch das Fehlerverfolgungswerkzeug innerhalb des Lebenszyklus)
- die spezifischen Arbeitsergebnisse (z.B. Testobjekte mit Versionsnummern), in denen das Problem beobachtet wurde, sowie die spezifischen Arbeitsergebnisse, in denen das Problem endgültig behoben wurde
- Auswirkung auf die Interessen von Projekt- und Produkt-Stakeholdern
- Schlussfolgerungen, Empfehlungen und Genehmigungen für die zur Problembehebung durchgeführten bzw. nicht durchgeführten Maßnahmen
- Risiken, Kosten, Chancen und Nutzen, die sich ergeben, wenn der Fehlerzustand behoben, bzw. wenn er nicht behoben wird
- die Termine, an denen die verschiedenen (Status-)Übergänge im Fehlerlebenszyklus stattfanden, die jeweiligen Eigentümer des Berichts zum Zeitpunkt der jeweiligen Übergänge, sowie die von den Projektmitgliedern ergriffenen Maßnahmen zum Eingrenzen und Reparieren des Fehlerzustands und Verifizieren der Fehlerbehebung

- eine Beschreibung, wie der Fehlerzustand letztendlich behoben wurde und Empfehlungen zum Testen der Lösung (falls der Fehlerzustand durch eine Softwareänderung behoben wurde)
- sonstige Verweise und Angaben, wie z.B. der Test, der den Fehlerzustand aufgedeckt hat, sowie Risiko, Anforderung oder sonstige Elemente der Testbasis, die einen Bezug zum Fehlerzustand haben (beim dynamischen Testen)

Es gibt mehrere Standards und Dokumente, wie z.B. ISO 9126 (wird ersetzt durch ISO 25000 [ISO25000], IEEE Standard 829 [IEEE 829], IEEE Standard 1044 [IEEE 1044] und "Orthogonal Defect Classification" (orthogonale Fehlerklassifizierung), die für den Testmanager hilfreich sind bei der Bestimmung der für den Fehlerbericht zu sammelnden Informationen. Egal welche Informationen letztlich in den Fehlerbericht aufgenommen werden, es ist in jedem Fall äußerst wichtig, dass die von den Testern eingegebenen Informationen vollständig, kurz und prägnant, richtig und präzise, objektiv und relevant sind und dass sie rechtzeitig erfolgen. Selbst wenn Probleme mit den Informationen des Fehlerberichts durch manuelle Intervention und direkte Kommunikation bei der Behebung einzelner Fehlerzustand überwunden werden können, so können diese Probleme in Zusammenhang mit einer korrekten Bewertung von Projektstatus, Testfortschritt und Prozessreife zu unüberwindbaren Hindernissen werden.

4.4 Bewerten der Prozessreife anhand der Fehlerberichtsinformationen

Wie in Kapitel 2 beschrieben, können Fehlerberichte für die Überwachung und das Berichten des Projektstatus nützlich sein. Die Auswirkungen der Metriken auf den Prozess werden hauptsächlich im Testmanagement-Modul des Expert-Level-Lehrplans behandelt. Ein Advanced Level Testmanagement [ISTQB ETM SYL] sollte wissen, welche Bedeutung Fehlerberichte für die Bewertung der Prozessreife der Softwaretest- und Softwareentwicklungsprozesse haben.

Zusätzlich zur Unterstützung der Testfortschrittsüberwachung (wie in Kapitel 2 und in Abschnitt 4.3 erwähnt) unterstützen die Fehlerberichtsinformationen auch Prozessverbesserungsmaßnahmen, z.B. durch:

- Verwendung der Information über die Phasen in denen Entstehung, Aufdeckung und Behebung der Fehlerzustände erfolgte, jeweils getrennt nach Phase, um die Fehlereindämmung innerhalb der Phasen zu bewerten und Wege zu einer besseren Fehlerfindungseffektivität in den einzelnen Phasen aufzuzeigen.
- Verwendung der Informationen über die Phase der Entstehung für eine Pareto-Analyse derjenigen Phasen, in denen die meisten Fehlerzustände entstehen, um zielgerichtete Verbesserungsmaßnahmen zur Fehlerreduzierung zu ermöglichen.
- Verwendung der Informationen über Grundursachen, um die darunterliegenden Gründe für die Entstehung von Fehlerzuständen herauszufinden, um Prozessverbesserungsmaßnahmen zur Reduktion der Gesamtzahl der Fehlerzustände einzuleiten.
- Verwendung der Informationen der Phasen Einführung, Aufdeckung und Behebung, um die Kosten der Qualität zu analysieren und die Kosten im Zusammenhang mit Fehlerzuständen zu minimieren.
- Verwendung der Informationen über Komponenten mit Fehlerzuständen, um Fehleranhäufungen zu analysieren, um technische Risiken (für das risikoorientierte Testen) besser zu verstehen und um fehlerhafte Komponenten neu zu überarbeiten.

Die Verwendung von Metriken zur Bewertung der Effektivität und Effizienz des Testprozesses werden im Expert Level Lehrplan Testmanagement behandelt [ISTQB ETM SYL].

Manchmal entscheiden sich Teams, keine Fehleverfolgung in einem Teil oder über den gesamten Softwarelebenszyklus durchzuführen. Dies geschieht häufig im Namen der Effizienz und Reduzierung

des Prozessaufwands. In Wirklichkeit verringert sich dadurch jedoch die notwendige Transparenz, die eine Bewertung der Prozessreife von Softwaretests- und -entwicklung ermöglicht. Das erschwert die oben angeregten Verbesserungen, weil es an zuverlässigen Daten fehlt.

5. Verbesserung des Testprozesses – [135 Minuten]

Begriffe

Capability Maturity Model Integration (CMMI), Critical Testing Processes (CTP), Systematic Test and Evaluation Process (STEP), Test Maturity Model integration (TMMi), TPI Next

Lernziele für die Verbesserung des Testprozesses

5.2 Testverbesserungsprozess

TM-5.2.1 (K2) Sie können anhand von Beispielen erklären, warum es wichtig ist, den Testprozess zu verbessern

5.3 Testprozess verbessern

TM-5.3.1 (K3) Sie können einen Testprozessverbesserungsplan mit dem IDEAL-Modell erstellen

5.4 Testprozess mit TMMi verbessern

TM-5.4.1 (K2) Sie können Hintergrund, Umfang und Ziele des Testprozessverbesserungsmodells TMMi zusammenfassen

5.5 Testprozess mit TPI Next verbessern

TM-5.5.1 (K2) Sie können Hintergrund, Umfang und Ziele des Testprozessverbesserungsmodells TPI Next zusammenfassen

5.6 Testprozess mit CTP verbessern

TM-5.6.1 (K2) Sie können Hintergrund, Umfang und Ziele des Testprozessverbesserungsmodells CTP zusammenfassen

5.7 Testprozess mit STEP verbessern

TM-5.7.1 (K2) Sie können Hintergrund, Umfang und Ziele des Testprozessverbesserungsmodells STEP zusammenfassen

5.1 Einführung

Ist ein firmenweiter Testprozess eingeführt, sollte er kontinuierlich verbessert werden. Dieses Kapitel beleuchtet zunächst allgemeine Themen zur Testprozessverbesserung, gefolgt von einer Einführung in einige spezifische Testprozessverbesserungsmodelle. Testmanager sollten davon ausgehen, dass sie die treibende Kraft für Änderungen und Verbesserungen des Testprozesses sind. Sie sollten daher die in der Industrie anerkannten Verfahren kennen, welche in diesem Kapitel behandelt werden. Weitere Informationen zur Testprozessverbesserung werden im Expert Level Lehrplan „Testprozessverbesserung“ behandelt.

5.2 Testverbesserungsprozess

So wie das Testen dazu dient, die Software zu verbessern, werden Techniken zur Prozessbewertung und -verbesserung ausgewählt und eingesetzt, um den Softwareentwicklungsprozess (und die daraus resultierenden Arbeitsergebnisse) zu verbessern. Prozessverbesserung ist auch für Testprozesse möglich. Es gibt unterschiedliche Möglichkeiten und Methoden, durch die das Testen von Software und Softwaresystemen verbessert werden kann. Diese Methoden haben das Ziel, den Prozess (und damit die Arbeitsergebnisse) zu verbessern, indem sie Richtlinien und Einsatzbereiche für die Verbesserungen zur Verfügung stellen.

Das Testen macht oft einen wesentlichen Teil der Gesamtkosten eines Projektes aus. Dennoch findet der Testprozess in den diversen Softwareprozessverbesserungsmodellen wie CMMI® (siehe unten) nur begrenzte Beachtung.

Testprozessverbesserungsmodelle, wie Test Maturity Model Integration (TMMi®), Systematic Test and Evaluation Process (STEP), Critical Testing Processes (CTP) und TPI Next® wurden entwickelt, um die mangelnde Aufmerksamkeit für das Testen in den meisten Softwareprozessverbesserungsmodellen zu thematisieren. Wenn sie richtig angewendet werden, dann liefern diese Modelle ein gewisses Maß an organisationsübergreifenden Metriken für benchmarkartige Vergleiche.

Die Modelle in diesem Lehrplan sind nicht als Anwendungsempfehlungen zu verstehen, sondern sollen vielmehr eine repräsentative Darstellung der Funktionsweise und der Inhalte von Modellen sein.

5.2.1 Einführung in die Prozessverbesserung

Prozessverbesserungen sind sowohl für den Softwareentwicklungs- als auch für den Testprozess relevant. Eine Organisation lernt aus den eigenen Fehlern und kann dann die Prozesse verbessern, die sie für die Entwicklung und das Testen von Software einsetzt. Der PDCA-Zyklus nach Deming (Plan/Do/Check/Act = Planen/Ausführen/Prüfen/Handeln) ist seit Jahrzehnten im Einsatz und wird auch noch heute zur Prozessverbesserung eingesetzt.

Eine Voraussetzung der Prozessverbesserung ist die Vorstellung, dass die Qualität eines Systems stark von der Qualität des Prozesses abhängt, der zur Entwicklung der Software verwendet wird. Qualitätsverbesserungen in der Softwareindustrie verringern die für die Wartung der Software benötigten Ressourcen und stellen dadurch einen größeren zeitlichen Freiraum für das Ausarbeiten von mehreren und besseren Lösungen für die Zukunft zur Verfügung. Prozessmodelle bieten einen Ansatz für Verbesserungen, indem sie die Prozessfähigkeit einer Organisation gegen das Modell messen. Das Modell dient außerdem als Rahmenwerk für die Verbesserung der Prozesse in einer Organisation anhand der Bewertungsergebnisse.

Eine Prozessbewertung führt zu einer Bestimmung der Prozessreife, die wiederum eine Prozessverbesserung anregt. Später lässt sich in einer weiteren Prozessbewertung die Wirkung der Verbesserung messen.

5.2.2 Arten der Prozessverbesserung

Bewertungsmodelle sind eine übliche Methode, die einen standardisierten Ansatz für die Verbesserung der Testprozesse nach erprobten und bewährten Verfahren gewährleisten.

Prozessverbesserungsmodelle werden in zwei Kategorien eingeteilt:

1. Das Prozessreferenzmodell liefert eine Messung der Prozessreife. Hierfür wird der Reifegrad einer Organisation mit dem Modell verglichen, sie innerhalb des Rahmenwerks des Modells bewertet und es wird ein Weg für die Prozessverbesserung aufgezeigt.
2. Das Inhaltsreferenzmodell liefert geschäftsgetriebene Bewertungen und Verbesserungsmöglichkeiten für die Organisation. Dies beinhaltet in manchen Fällen objektive Messungen für benchmarkartige Vergleiche mit dem Branchendurchschnitt. Diese Bewertung kann dazu dienen, einen Weg für die Verbesserung des Prozesses aufzuzeigen.

Eine Testprozessverbesserung ist aber auch ohne Modell möglich, beispielsweise durch analytische Ansätze und Reviewsitzungen.

5.3 Testprozess verbessern

Die IT-Branche verwendet häufig Testprozessverbesserungsmodelle, um mehr Reife und Professionalität zu erreichen. Dabei dienen Standardmodelle dazu, organisationsübergreifende Metriken und Maße zu entwickeln, die einen Vergleich ermöglichen. Da es in der Testbranche einen klaren Bedarf für Prozessverbesserungen gibt, haben sich mehrere empfohlene Modelle herausgebildet, beispielsweise STEP, TMMi, TPI Next und CTP. Stufenmodelle, wie TMMi und CMMI, liefern Standards für Vergleiche zwischen verschiedenen Unternehmen und Organisationen. Kontinuierliche Modelle, wie CTP, STEP und TPI Next, erlauben es den Organisationen, die Themen mit der höchsten Priorität freier in der Reihenfolge der Implementierung handzuhaben. Die erwähnten Modelle werden in den folgenden Abschnitten behandelt.

Mit jedem dieser Modelle kann eine Organisation herausfinden, wo sie gegenwärtig mit ihrem derzeitigen Testprozess steht. Nach der Bewertung liefern TMMi und TPI Next einen verbindlichen Weg für die Verbesserung des Testprozesses. Die Modelle STEP und CTP hingegen helfen einer Organisation herauszufinden, wo sich Investitionen in die Testprozessverbesserung am meisten rentieren und überlassen ihr die Wahl des geeigneten Verbesserungswegs.

Wenn Einigkeit herrscht, dass die Testprozesse geprüft und verbessert werden sollen, lassen sich die für die Prozessverbesserung notwendigen Schritte anhand des IDEAL[SM] Modells [IDEAL96] definieren:

- Initiierung des Verbesserungsprozesses (engl.: **i**nitiating)
- Diagnose der bestehenden Situation (engl.: **d**iagnosing)
- Etablieren eines Testprozessverbesserungsplans (engl.: **e**stablishing)
- Agieren, um Verbesserungen zu implementieren (engl.: **a**cting)
- Lernen aus dem Verbesserungsprogramm (engl. **l**earning)

Initiierung des Verbesserungsprozesses
Bevor die Aktivitäten zur Prozessverbesserung beginnen, werden Zielsetzungen, Umfang und Überdeckung der vorgesehenen Prozessverbesserungen mit den Stakeholdern vereinbart. Zu diesem Zeitpunkt wird auch das Prozessverbesserungsmodell ausgewählt. Zur Wahl stehen zum einen die veröffentlichten Modelle (wie CTP, STEP, TMMi und TPI Next) oder zum anderen ein individuell definiertes Modell. Außerdem sollten Erfolgskriterien definiert sowie die Methode implementiert werden, die während der Prozessverbesserungsaktivitäten zur Messung dieser Erfolgskriterien anzuwenden ist.

Diagnose der bestehenden Situation
Der vereinbarte Bewertungsansatz wird durchgeführt und ein Testbewertungsbericht erstellt, der als Ergebnis eine Bewertung der gegenwärtigen Testpraktiken und eine Liste möglicher Prozessverbesserungen liefert.

Etablieren eines Testprozessverbesserungsplans
Die Liste der möglichen Prozessverbesserungen wird nach Priorität geordnet. Die Priorität kann sich orientieren an: Rentabilität, Risiken, Angleichung an die Gesamtstrategie der Organisation und/oder an messbarem quantitativen oder qualitativen Nutzen. Nach Erstellung der priorisierten Liste wird ein Plan für die Durchführung der Verbesserungen entwickelt.

Agieren, um Verbesserungen zu implementieren
Der Testprozessverbesserungsplan für die Durchführung der Verbesserungen wird umgesetzt. Dazu können Schulungen oder Coachings anfallen und es kann die Pilotierung von Prozessen einschließen. Der Prozess führt schließlich zur vollständigen Implementierung der Verbesserungen.

Lernen aus dem Verbesserungsprogramm
Wenn die Prozessverbesserungen im Einsatz sind, ist unbedingt zu prüfen, ob sich die vorab vereinbarten Vorteile verwirklicht haben (gemeint sind die geplanten sowie eventuell unerwarteten Vorteile). Außerdem ist zu prüfen, welche der Erfolgskriterien für die Prozessverbesserungsmaßnahmen erfüllt wurden.

Je nach verwendetem Prozessmodell ist dies der Schritt im Verbesserungsprozess, mit dem die nächste Reifestufe anvisiert wird und die Entscheidung fällt, ob der Verbesserungszyklus von vorne beginnen oder vorerst an diesem Punkt enden soll.

5.4 Testprozess mit TMMi verbessern

Das Test Maturity Model Integration (TMMi) besteht aus fünf Reifestufen und soll CMMI ergänzen. Jede der fünf Stufen enthält definierte Prozessbereiche, die durch die Erreichung spezifischer und allgemeiner Ziele zu 85% erfüllt sein müssen, bevor die Organisation mit der nächsten Stufe fortfahren kann.

Die TMMi-Reifestufen sind:

- Stufe 1: Initial
  Auf der ersten Stufe gibt es keinen formal dokumentierten oder strukturierten Testprozess. Tests werden normalerweise nach dem Programmieren ad hoc entwickelt und das Testen als Debugging verstanden. Ziel des Testens ist es, nachzuweisen, dass die Software funktioniert.
- Stufe 2: Managed
  Die zweite Stufe wird erreicht, wenn die Testprozesse klar vom Debugging getrennt sind. Diese Stufe kann durch das Formulieren von Testrichtlinien und Testzielen, Einführen eines fundamentalen Testprozesses (z.B. Testplanung), sowie Implementieren grundlegender Testverfahren und -methoden erreicht werden.

- Stufe 3: Defined
  Die dritte Stufe wird erreicht, wenn ein Testprozess in den Softwarelebenszyklus integriert und in formalen Standards, Verfahren und Methoden dokumentiert wird. Es finden Reviews statt und es existiert eine klar definierte Softwaretestfunktion, die gesteuert und überwacht werden kann.
- Stufe 4: Measured
  Die vierte Stufe wird erreicht, wenn sich der Testprozess auf Unternehmensebene sowohl effektiv messen und steuern lässt, als auch wenn spezifische Projekte davon profitieren.
- Stufe 5: Optimized
  In der höchsten Stufe wird eine Testprozessreife erreicht, bei der sich Daten aus dem Testprozess nutzen lassen, um Fehlerzuständen vorzubeugen und bei der die weitere Optimierung des etablierten Prozesses in den Fokus rückt.

Weitere Informationen über TMMi finden Sie in [vanVeenendaal11] und unter [www.tmmi.org].

5.5 Testprozess mit TPI Next verbessern

Das TPI Next-Modell definiert 16 Kernbereiche, von denen jeder einen bestimmten Aspekt des Testprozesses wie z.B. Teststrategie, Metriken, Testwerkzeuge und Testumgebung abdeckt.

Im Modell sind vier Reifestufen vorgesehen:

- Initial
- Kontrolliert
- Effizient
- Optimierend

Es sind spezifische Kontrollpunkte definiert, um jeden der Kernbereiche in jeder Reifestufe zu bewerten. Die Ergebnisse werden zusammengefasst und in Form einer Testreifematrix visualisiert, die alle Kernbereiche abdeckt. Die Definition von Verbesserungszielen und deren Implementierung kann auf die Erfordernisse und Leistungsfähigkeit der Testorganisation zugeschnitten werden.

Der allgemeine Ansatz macht das TPI Next Modell unabhängig von den Softwareprozessverbesserungsmodellen. Mit TPI Next werden sowohl Aspekte des Testens als auch Unterstützung von Managemententscheidungen abgedeckt [deVries09].

Für weitere Informationen über TPI Next, siehe [www.tpinext.com].

5.6 Testprozess mit CTP verbessern

Der zentrale Grundsatz beim CTP-Bewertungsmodell (Critical Testing Process) besagt, dass bestimmte Testprozesse als kritisch gelten. Wenn diese kritischen Prozesse gut ausgeführt werden, dann sind sie eine Unterstützung für erfolgreiche Testteams. Das gilt auch umgekehrt: Wenn diese Aktivitäten schlecht ausgeführt werden, bleiben selbst begabte Tester und Testmanager wahrscheinlich erfolglos. Das CTP-Modell bestimmt zwölf kritische Testprozesse. Das Modell der kritischen Testprozesse (CTP) ist in erster Linie ein Inhaltsreferenzmodell.

Das CTP-Modell ist ein kontextabhängiger Ansatz, der sich anpassen lässt, beispielsweise durch:

- Identifizieren von spezifischen Herausforderungen
- Erkennen von Merkmalen guter Prozesse
- Auswahl der Reihenfolge und der Wichtigkeit für die Implementierung von Prozessverbesserungen

Das CTP-Modell lässt sich für alle Lebenszyklusmodelle der Softwareentwicklung anpassen.

Zusätzlich zu Interviews mit Teilnehmern, beinhaltet das CTP-Modell die Verwendung von Metriken, um Organisation mit Branchendurchschnitten und Best Practices zu vergleichen.

Für weitere Informationen über CTP, siehe [Black03].

5.7 Testprozess mit STEP verbessern

STEP (Systematic Test and Evaluation Process) setzt wie CTP (und anders als TMMi und TPI Next) nicht voraus, dass Verbesserungen in einer bestimmten Reihenfolge erfolgen.

STEP ist in erster Linie ein Inhaltsreferenzmodell, das auf der Vorstellung basiert, dass das Testen eine Aktivität über den gesamten Softwarelebenszyklus ist, welche mit der Formulierung der Anforderungen beginnt und fortgeführt wird, bis das System außer Betrieb genommen wird. Die STEP-Methode verfährt nach dem Grundsatz „erst testen, dann programmieren“. Ihre anforderungsbasierte Teststrategie soll erreichen, dass die frühe Erstellung der Testfälle die Anforderungsspezifikation noch vor Systemdesign und -programmierung validieren.

Grundvoraussetzungen der Methode sind u.a.:

- Eine anforderungsbasierte Teststrategie;
- das Testen beginnt am Anfang des Softwarelebenszyklus;
- Tests werden als Anforderungs- und Anwendungsmodelle eingesetzt;
- Design der Testmittel führt das Softwaredesign;
- Fehlerzustände werden früher gefunden oder ganz vermieden;
- Fehlerzustände werden systematisch analysiert;
- Tester und Entwickler arbeiten zusammen.

Das STEP-Bewertungsmodell wird manchmal mit dem TPI Next-Reifemodell kombiniert.

Für weitere Informationen über STEP, siehe [Craig02].

6. Testwerkzeuge und Automatisierung – [135 Minuten]

Begriffe

Open-Source Werkzeug, maßgeschneidertes Werkzeug

Lernziele für Testwerkzeuge und Automatisierung

6.2 Auswahl von Werkzeugen

TM-6.2.1 (K2) Sie können die für das Management relevanten Kriterien bei der Auswahl von Open-Source-Werkzeugen beschreiben

TM-6.2.2 (K2) Sie können die für das Management relevanten Kriterien bei der Auswahl von maßgeschneiderten Werkzeugen beschreiben

TM-6.2.3 (K4) Sie können eine vorgegebene Situation bewerten und einen entsprechenden Plan für die Auswahl von Werkzeugen erstellen, in dem Risiken, Kosten und Nutzen berücksichtigt sind

6.3 Werkzeuglebenszyklus

TM-6.3.1 (K2) Sie können die verschiedenen Phasen im Werkzeuglebenszyklus erklären

6.4 Werkzeugmetriken

TM-6.4.1 (K2) Sie können beschreiben, wie das Sammeln und Bewerten von Metriken durch den Einsatz von Werkzeugen verbessert werden kann

6.1 Einführung

Dieser Abschnitt vertieft den Foundation Level Lehrplan mit einigen allgemeinen Konzepten, die Testmanager in Zusammenhang mit Werkzeugen und Automatisierung berücksichtigen müssen.

6.2 Auswahl von Werkzeugen

Es gibt unterschiedliche Aspekte, die Testmanager bei der Auswahl von Werkzeugen berücksichtigen müssen.

Meist wurden Werkzeuge in der Vergangenheit von einem kommerziellen Anbieter gekauft. In manchen Fällen ist das auch die einzige Option, die in Frage kommt. Es gibt jedoch auch Alternativen, wie Open-Source-Werkzeuge und maßgeschneiderte Werkzeuge; auch diese können praktikable Optionen sein.

Unabhängig von der Art des Werkzeuges, müssen Testmanager die Gesamtbetriebskosten (Total Cost of Ownership) untersuchen, die über die erwartete Lebensdauer des Werkzeugs (zusätzlich zu den Anschaffungskosten) anfallen. Hierfür ist eine Kosten-Nutzen-Analyse durchzuführen. Diese Thematik wird im Abschnitt Rentabilität behandelt.

6.2.1 Open-Source-Werkzeuge

Open-Source-Werkzeuge sind für nahezu jede Facette des Testprozesses erhältlich, vom Testfallmanagement bis hin zur Fehlerverfolgung und Testfallautomatisierung, um nur ein paar wenige zu nennen. Open-Source-Werkzeuge unterscheiden sich aber von kommerziellen Werkzeugen vor allem dadurch, dass sie in der Regel zwar keine hohen Anschaffungskosten haben, es aber möglicherweise keinen geregelten Support für das Werkzeug gibt. Viele Open-Source-Werkzeuge haben jedoch eine gut organisierte Community, die den Benutzern in unkonventioneller und informeller Weise gerne Support bietet.

Eine weitere typische Besonderheit von Open-Source-Werkzeugen ist, dass viele Open-Source-Werkzeuge ursprünglich zur Lösung eines spezifischen Problems oder mit einem ganz speziellen Ziel erstellt wurden, weshalb sie vielleicht nicht alle Funktionen abdecken, die ein vergleichbares kommerzielles Werkzeug bieten würde. Vor der Auswahl eines Open-Source-Werkzeuges sollte daher eine sorgfältige Analyse der tatsächlichen Erfordernisse des Testteams durchgeführt werden.
Ein Vorteil von Open-Source-Werkzeugen ist, dass sie in der Regel vergleichsweise leicht von den Benutzern modifiziert oder erweitert werden können. Wenn eine Testorganisation über die Kernkompetenzen verfügt, dann kann sie das Werkzeug so anpassen, dass es zusammen mit anderen Werkzeugen funktioniert oder besser den Bedürfnissen des Testteams entspricht. Mehrere Werkzeuge lassen sich auch kombinieren und können so Probleme lösen, die ein kommerzielles Werkzeug nicht bearbeiten kann. Selbstverständlich steigen Komplexität und Kosten, je mehr Werkzeuge eingesetzt und je mehr Modifikationen gemacht werden. Ein Testmanager muss sicherstellen, dass der Einsatz eines Open-Source-Werkzeugs nicht zum Selbstzweck wird. Wie auch bei anderen Werkzeugen, muss das Ziel des Aufwands immer eine positive Rentabilität sein.

Testmanager müssen die Lizenzbestimmungen des ausgewählten Werkzeugs verstehen. Für viele Open-Source-Werkzeuge gilt eine abgeänderte Version der GNU General Public License, die festlegt, dass die Weitergabe der Software immer zu denselben Bedingungen erfolgen muss wie das Beschaffen der Software. Falls das Testteam also das Werkzeug zur Unterstützung des eigenen Testens modifiziert, dann müssen diese Änderungen nach den Bestimmungen der Werkzeuglizenz unter Umständen allen externen Benutzern unentgeltlich zur Verfügung gestellt werden. Testmanager sollten daher die rechtlichen Konsequenzen für ihre Organisation beachten, die im Zusammenhang

mit der Verwendung von Open-Source-Werkzeugen für das eigene Softwareprodukt entstehen könnten.

Organisationen, die sicherheitskritische oder geschäftskritische Software entwickeln, oder Software, die strenge Vorschriften einhalten muss, haben möglicherweise Vorbehalte in Zusammenhang mit dem Einsatz von Open-Source-Werkzeugen. Obwohl viele Open-Source-Werkzeuge qualitativ hochwertig sind, ist die Genauigkeit der einzelnen Werkzeuge wahrscheinlich nicht zertifiziert. Kommerzielle Werkzeuge dagegen sind häufig hinsichtlich ihrer Genauigkeit und Verwendung für bestimmte Aufgaben zertifiziert (z.B. DO-178B). Obwohl Open-Source-Werkzeuge möglicherweise genauso gut sind, müsste man dann aber als Benutzer sich selbst um die Zertifizierung kümmern, was zusätzliche Kosten verursacht.

6.2.2 Maßgeschneiderte Werkzeuge

Die Testorganisation hat möglicherweise einen spezifischen Bedarf, für den weder ein kommerzielles noch ein Open-Source-Werkzeug auf dem Markt erhältlich ist. Gründe könnten eine proprietäre Hardware sein, eine angepasste Testumgebung oder ein Prozess, der auf eine ungewöhnliche Weise modifiziert wurde. In solchen Fällen könnten Testmanager, falls im Team die notwendige Kernkompetenz vorhanden ist, die Entwicklung eines maßgeschneiderten Werkzeuges in Betracht ziehen.

Die Vorteile der Entwicklung eines maßgeschneiderten Werkzeugs sind, dass es die Erfordernisse des Teams präzise erfüllen und effizient im gegebenen Kontext betrieben werden kann. Das Werkzeug kann für eine Integration mit anderen verwendeten Werkzeugen ausgelegt werden; außerdem können die vom Team benötigten Daten genau in der benötigten Form generiert werden. Darüber hinaus können sich, abgesehen vom aktuellen Projekt, weitere Nutzungsmöglichkeiten des Werkzeuges innerhalb der Organisation anbieten. Allerdings sollten vor der Freigabe des Werkzeuges für andere Projekte unbedingt der Zweck, die Ziele, sowie dir Vor- und Nachteile geprüft werden.

Testmanager, die eine Entwicklung von maßgeschneiderten Werkzeugen erwägen, müssen sich auch mit den möglichen negativen Aspekten auseinandersetzen. Maßgeschneiderte Werkzeuge hängen oft sehr von der Person ab, die sie entwickelt hat. Maßgeschneiderte Werkzeuge müssen daher so dokumentiert werden, dass sie auch von anderen Personen gewartet und weiterentwickelt werden können.
Ansonsten würden sie verwaisen oder gar nicht mehr verwendet werden, sollte die Person, die sie entwickelt hat, das Projekt einmal verlassen. Im Laufe der Zeit kann es passieren, dass die maßgeschneiderten Werkzeuge über den ursprünglich beabsichtigten Umfang hinaus erweitert werden. Das kann wiederum Qualitätsprobleme nach sich ziehen, z.B. falsch positive Fehlerberichte oder die Generierung inkorrekter Daten. Testmanager müssen bedenken, dass ein maßgeschneidertes Werkzeug im Grunde ein weiteres Softwareprodukt ist, für das dieselben Entwicklungsprinzipien gelten, wie für jedes andere Softwareprodukt. Maßgeschneiderte Werkzeuge sollten so entwickelt und getestet werden, dass sie den Erwartungen entsprechend funktionieren.

6.2.3 Rentabilität

Testmanager müssen sicherstellen, dass alle in die Testorganisation eingeführten Werkzeuge einen Mehrwert für die Arbeit des Teams darstellen und sie der Organisation gegenüber eine positive Rentabilität nachweisen können. Durch eine Kosten-Nutzen-Analyse lässt sich sicherstellen, dass ein Werkzeug tatsächlich und dauerhaft Nutzen bringt; daher sollte eine Kosten-Nutzen-Analyse vor der Anschaffung bzw. der Entwicklung eines Werkzeuges durchgeführt werden. Zu berücksichtigen sind dabei sowohl die wiederkehrenden als auch die einmaligen Kosten sowie die Risiken, die den Wert des Werkzeugs mindern könnten. Einige der Kosten sind monetäre Kosten, andere hingegen Kosten für Ressourcen oder Zeitaufwand.

Zu den einmaligen Kosten gehören:
- Spezifizieren der Anforderungen an das Werkzeug zur Erreichung der Ziele
- Evaluieren und Auswählen des richtigen Werkzeugs und Werkzeuganbieters
- Kauf, Anpassung oder Eigenentwicklung des Werkzeugs
- Einführungsschulung für das Werkzeug
- Integrieren des Werkzeugs mit anderen Werkzeugen
- Beschaffen der erforderlichen Hardware/Software, um das Werkzeug in Betrieb nehmen zu können.

Zu den wiederkehrenden Kosten gehören:
- Werkzeugbesitz
  - Lizenz- und Supportgebühren
  - Wartung des Werkzeugs
  - Wartung von Artefakten, die vom Werkzeug erstellt wurden
  - Laufende Schulungs- und Betreuungskosten
- Übertragen des Werkzeugs in verschiedene Umgebungen
- Anpassen des Werkzeugs an zukünftige Erfordernisse
- Prozess- und Qualitätsverbesserung, um die optimale Nutzung der gewählten Werkzeuge sicherzustellen.

Testmanager sollten außerdem die Opportunitätskosten, die jedes Werkzeug mit sich bringt, in die Überlegungen miteinbeziehen. Zeit, die für die Beschaffung, Verwaltung, Schulung und Verwendung des Werkzeugs aufgewendet wird, fehlt dann für die eigentlichen Testaufgaben, weshalb bis zum produktiven Einsatz des Werkzeugs zusätzliche Ressourcen für das Testen erforderlich sein könnten.

In Zusammenhang mit dem Einsatz von Werkzeugen gibt es viele Risiken und nicht alle Werkzeuge bringen ausreichend Nutzen, um die Risiken auszugleichen. Die Risiken einer Werkzeugunterstützung wurden bereits im Foundation Level Lehrplan behandelt. Testmanager sollten bezüglich der Rentabilität auch noch folgende Risiken berücksichtigen:
- Mangelnde Reife der Organisation (der Einsatz des Werkzeugs kommt noch zu früh)
- Arbeitsergebnisse, die mit dem Werkzeug erstellt worden sind, könnten schwer wartbar sein und bei Änderung der getesteten Software erhebliche Überarbeitungen erfordern
- Test Analysten werden durch die Einführung des Werkzeugs von ihren Testaufgaben abgehalten, worunter der Erfolg des Testens leiden kann (z.B. kann sich die Effektivität der Fehlerfindungsrate reduzieren, weil ausschliesslich automatisierte Testskripte durchgeführt werden).

Schließlich müssen Testmanager den zusätzlichen Nutzen erkennen, der sich durch die Werkzeugunterstützung erzielen lässt. Zum Nutzen des Werzeugeinsatzes können gehören:
- Weniger repetitive Arbeiten
- Kürzere Testzykluszeiten (z.B. durch automatisierte Regressionstests)
- Geringere Testausführungskosten
- Intensivierung bestimmter Testarten (z.B. Regressionstests)
- Weniger menschliches Fehlverhalten in verschiedenen Testphasen. Beispiele dafür sind u.a.:
  - Testdaten können bei Werkzeugunterstützung durch Testdatengeneratoren einen höheren Wert haben
  - Vergleiche von Testergebnissen können bei Werkzeugunterstützung durch Testkomparatoren präziser werden
  - Die korrekte Eingabe der Testdaten kann mit Hilfe eines Skriptwerkzeugs zuverlässiger werden
- Weniger Aufwand in Zusammenhang mit dem Zugriff auf Testinformationen. Beispiele dafür sind u.a.:
  - Berichte und Metriken werden vom Werkzeug generiert

    - Testmittel (z.B. Testfälle, Testskripte und Testdaten) werden wiederverwendet
- Mehr Tests, die ohne Werkzeugunterstützung nicht möglich waren (z.B. Performanztests, Lasttests)
- Verbesserte Reputation der Tester und letztendlich auch der Testorganisation, die die Automatisierung durchgeführt haben, weil sie bewiesen haben, dass sie etwas von anspruchsvollen Werkzeugen verstehen und damit umgehen können

Generell gilt, dass Testteams nur selten ein einziges Werkzeug verwenden; dementsprechend ist die Gesamtrentabilität, die das Testteam erzielt, in der Regel auch eine Funktion aller eingesetzten Werkzeuge zusammen. Werkzeuge müssen Informationen austauschen und eng miteinander zusammenarbeiten. Es empfiehlt sich daher, eine langfristige, umfassende Testwerkzeugstrategie zu verfolgen.

6.2.4 Auswahlverfahren

Testwerkzeuge sind eine langfristige Investition, die meistens mehrere Iterationen eines Projekts und/oder mehrere Projekte betrifft. Testmanager müssen ein potenzielles Werkzeug unter mehreren Gesichtspunkten betrachten:

- Aus Sicht der Organisation muss das Werkzeug rentabel sein. Damit ein hoher Gegenwert der Investitionen erreicht wird, sollte die Organisation sicherstellen, dass die Werkzeuge mit anderen Testwerkzeugen oder sonstigen Werkzeugen zusammenarbeiten. Manchmal müssen Prozesse und Konnektivität für den Einsatz der Werkzeuge verbessert werden, damit die Interoperabilität erreicht wird; dafür ist die entsprechende Zeit einzuplanen.
- Aus Sicht des Projekts muss das Werkzeug effektiv sein (z.B. um Fehler beim manuellen Testen zu vermeiden, wie Tippfehler bei der Dateneingabe). Es kann beträchtliche Zeit dauern, bevor ein Werkzeug sich bezahlt macht und einen positiven ROI (Return on Investment) erreicht. Häufig wird dies erst nach dem zweiten Release oder in der Wartungsphase erreicht und nicht in der Anfangsphase, wenn die Automatisierung implementiert wurde. Testmanager sollten den gesamten Lebenszyklus der Anwendung berücksichtigen.
- Aus dem Blickwinkel der Person, die das Werkzeug benutzt, muss das Werkzeug alle Projektmitglieder bei der Durchführung ihrer Aufgaben effizient und effektiv unterstützen. Die Lernkurve ist mit einzuplanen, um sicherzustellen, dass die Benutzer das Werkzeug schnell und mit minimalem Stress erlernen können. Wenn ein Werkzeug eingeführt wird, ist eine Schulung und Anleitung der Anwender erforderlich.

Damit alle Gesichtspunkte in die Überlegungen einbezogen werden, ist es wichtig, einen Plan für die Einführung des Testwerkzeugs auszuarbeiten.

Folgende Gesichtspunkte bei der Auswahl eines Werkzeugs wurden bereits im Foundation Level Lehrplan behandelt:

- Bewerten der Reife einer Organisation
- Identifizieren der Anforderungen für die Werkzeuge
- Evaluation der Werkzeuge
- Evaluation der Anbieter oder des Servicesupports (Open-Source-Werkzeuge, maßgeschneiderte Werkzeuge)
- Identifikation der internen Anforderungen für Coaching und Anleitung bei der Anwendung der Werkzeuge
- Evaluation des Schulungsbedarfs, unter Berücksichtigung der vorhandenen Testautomatisierungskenntnisse des Testteams
- Einschätzen des Kosten/Nutzen-Verhältnisses (siehe Abschnitt 6.2.3 Rentabilität)

Für jede Werkzeugart sollten Testmanager, unabhängig von der Testphase, in der sie eingesetzt werden soll, die nachfolgend aufgelisteten Fähigkeiten berücksichtigen:

- Analyse
  - Wird man mit dem Werkzeug die Eingaben korrekt abbilden können?
  - Ist das Werkzeug für den vorgesehenen Zweck geeignet?
- Entwurf
  - Wird dieses Werkzeug beim Entwurf von Testmitteln auf Basis der vorhandenen Informationen behilflich sein (z.B. Testentwurfswerkzeuge, die Testfälle aus den Anforderungen entwerfen)?
  - Kann der Entwurf automatisch erstellt werden?
  - Können die tatsächlichen Skripte ganz oder teilweise in einem wartbaren und benutzbaren Format generiert werden?
  - Können die benötigten Testdaten automatisch generiert werden (z.B. aus der Codeanalyse)?
- Daten- und Testauswahl
  - Wie wählt das Werkzeug die benötigten Daten aus (z.B. welcher Testfall mit welchen Daten ausgeführt wird)?
  - Kann das Werkzeug die manuell oder automatisch eingegebenen Auswahlkriterien verarbeiten?
  - Kann das Werkzeug auf Basis gewählter Eingaben bestimmen, wie Produktionsdaten anonymisiert werden?
  - Kann das Werkzeug basierend auf vorgegebenen Überdeckungskriterien bestimmen, welche Tests benötigt werden (z.B. kann das Werkzeug anhand einer vorgegebenen Menge an Anforderungen diese zurückverfolgen und bestimmen, welche Testfälle ausgeführt werden müssen?
- Ausführung
  - Läuft das Werkzeug automatisch oder sind manuelle Eingriffe notwendig?
  - Wie kann das Werkzeug angehalten werden und wie läuft es wieder an?
  - Sollte das Werkzeug relevante Ereignisse überwachen können (z.B. sollte ein Testmanagementwerkzeug den Testfallstatus automatisch aktualisieren, wenn ein Fehlerbericht, der zum Testfall gehört, geschlossen wird)?
- Auswertung
  - Wie bestimmt das Werkzeug, ob das aufgenommene Ergebnis korrekt ist? Verwendet es beispielsweise ein Testorakel, um die Richtigkeit der Antwort zu bestimmen?
  - Welche Möglichkeiten des Wiederanlaufes bietet das Werkzeug?
  - Liefert das Werkzeug geeignete Bericht- und Aufzeichnungsfunktionen?

6.3 Werkzeuglebenszyklus

Folgende vier verschiedene Phasen im Lebenszyklus der Nutzung eines Werkzeuges müssen Testmanager koordinieren können:

1. Beschaffung. Das Werkzeug muss, wie oben beschrieben, beschafft werden (siehe Abschnitt 6.2 Auswahl von Werkzeugen). Nachdem die Entscheidung für die Einführung des Werkzeuges getroffen wurde, sollte der Testmanager einen Werkzeug-Administrator bestimmen (in der Regel ein Test Analyst oder ein Technical Test Analyst). Diese Person entscheidet, wie und wann das Werkzeug verwendet wird, wo die erstellten Artefakte gespeichert werden, legt Namenskonventionen fest, usw. Werden diese Entscheidungen vorab anstatt ad-hoc getroffen, kann dies insgesamt einen deutlichen Unterschied in der Rentabilität des Werkzeugs bewirken. Die Anwender des Werkzeugs werden wahrscheinlich Schulungen benötigen.
2. Support und Wartung. Es werden laufende Aufwände für Support und Wartung des Werkzeuges erforderlich sein. Die Verantwortung für die Wartung kann an den Werkzeug-

Administrator gehen oder an eine eigens dafür zuständige Gruppe. Falls das Werkzeug mit anderen Werkzeugen zusammenarbeiten muss, dann müssen der Datenaustausch und die Prozessabläufe für die Zusammenarbeit geregelt werden. Auch Entscheidungen über die Datensicherung und Wiederherstellung sowie über die Ausgaben des Werkzeugs müssen getroffen werden.

3. Weiterentwicklung. Laufende Anpassungen des Werkzeuges müssen berücksichtigt werden. Im Laufe der Zeit können die Umgebung, die geschäftlichen Erfordernisse oder Anforderungen des Anbieters größere Änderungen am Werkzeug oder bei dessen Verwendung bewirken. Beispielsweise können sich aufgrund einer verlangten Aktualisierung des Werkzeuges durch den Anbieter Änderungen im Zusammenspiel mit anderen Werkzeugen ergeben. Eine Änderung an der Umgebung, die aus geschäftlichen Gründen benötigt wird, kann Probleme mit dem Werkzeug verursachen. Je komplexer die Betriebsumgebung eines Werkzeuges ist, desto mehr können schrittweise Veränderungen seine Verwendung stören. Je nachdem, wie wichtig die Rolle des Werkzeugs im Testen ist, müssen Testmanager für die Organisation den laufenden Betrieb des Werkzeuges sicherstellen.
4. Außerbetriebnahme. Irgendwann ist der Zeitpunkt erreicht, wenn die Nutzungsdauer des Werkzeugs abgelaufen ist und es in einem geregelten Ablauf außer Betrieb genommen werden muss. Die vom Werkzeug erbrachte Funktionalität muss ersetzt und die Daten aufbewahrt und archiviert werden. Dies kann erforderlich werden, weil das Werkzeug das Ende des Lebenszyklus erreicht hat, oder einfach weil ein Zeitpunkt erreicht ist, wo die Nutzen und Chancen einer Umstellung auf ein neues Werkzeug die Kosten und Risiken übersteigen.

Im gesamten Werkzeuglebenszyklus muss der Testmanager dafür sorgen, dass das Werkzeug reibungslos, zuverlässig und ununterbrochen das Testteam unterstützt.

6.4 Werkzeugmetriken

Testmanager können objektive Metriken der Werkzeuge, die von technischen Test Analysten und Test Analysten verwendet werden, aufsetzen und sammeln. Verschiedene Werkzeuge können wertvolle Echtzeitdaten aufzeichnen und den Aufwand für das Sammeln von Daten reduzieren. Diese Daten können vom Testmanager zur Einschätzung des Testaufwandes genutzt werden.

Unterschiedliche Werkzeuge sind auf die Sammlung unterschiedlicher Datentypen spezialisiert. Beispiele dafür sind u.a.:

- Testmanagementwerkzeuge können viele unterschiedliche Metriken liefern. Die Rückverfolgbarkeit von den Anforderungen zu Testfällen und automatisierten Skripten ermöglicht die Aufnahme von Überdeckungsmetriken. Jederzeit kann man sich ein Bild über die verfügbaren Tests, geplanten Tests sowie über den aktuellen Status (bestanden, nicht bestanden, nicht ausgeführt, geblockt, in der Warteschlange) machen.
- Fehlermanagementwerkzeuge liefern eine Fülle von Informationen über Fehlerzustände, einschließlich aktuellen Status, Schweregrad und Priorität, Zuordnung zum System, usw. Andere aufschlussreiche Daten, wie die Phasen, in denen der Fehlerzustand entstanden ist und gefunden bzw. nicht gefunden wurde, sind für die Prozessverbesserung hilfreich.
- Statische Analysatoren können dabei helfen, Probleme mit der Wartbarkeit aufzudecken und zu berichten.
- Performanzwerkzeuge können wertvolle Informationen über die Skalierbarkeit des Systems liefern.
- Überdeckungsanalysatoren unterstützen den Testmanager bei der Information, wieviel Prozent des Systems tatsächlich getestet wurde.

Die Berichtsanforderungen der Werkzeuge sollten während deren Auswahl und Konfiguration verstanden werden, um sicherzustellen, dass die durch die Werkzeuge aufgezeichneten Informationen so zur Verfügung gestellt werden können, dass sie für die Stakeholder verständlich sind.

7. Soziale Kompetenz und Teamzusammensetzung – [210 Minuten]

Begriffe

Unabhängigkeit des Testens

Lernziele für soziale Kompetenz und Teamzusammensetzung

7.2 Individuelle Fähigkeiten

TM-7.2.1 (K4) Sie können durch das Verwenden eines vorgegebenen Skill Assessments Fragebogens analysieren, wo die Stärken und Schwächen von Teammitgliedern liegen (bezogen auf das Anwenden von Softwaresystemen, auf die Kenntnisse über den Geschäftsbereich bzw. die Branche, auf die Systementwicklung, auf das Testen von Software und die zwischenmenschlichen Fähigkeiten)

TM-7.2.2 (K4) Sie können eine vorgegebene Kompetenzbewertung für ein Team analysieren, um Schulungen und die Entwicklung von Fähigkeiten zu definieren und zu planen

7.3 Dynamik im Testteam

TM-7.3.1 (K2) Sie können für eine vorgegebene Situation erläutern, welche Hard- und Softskills für die Leitung eines Testteams benötigt werden.

7.4 Testen in der Organisationsstruktur etablieren

TM-7.4.1 (K2) Sie können verschiedene Optionen für eine Unabhängigkeit des Testens erklären

7.5 Motivieren

TM-7.5.1 (K2) Sie können Beispiele für motivierende und demotivierende Faktoren für Tester anführen

7.6 Kommunizieren

TM-7.6.1 (K2) Sie können die Faktoren erklären, die eine effektive Kommunikation innerhalb eines Testteams, sowie zwischen einem Testteam und Stakeholdern beeinflussen

7.1 Einführung

Testmanager sind zuständig für das Rekrutieren von Mitarbeitern sowie für das Erhalten und Betreuen von Testteams, in denen die richtigen Fähigkeiten vertreten sind. Die benötigten Fähigkeiten können sich im Laufe der Zeit ändern. Es ist daher wichtig, nicht nur zu Beginn die richtigen Mitarbeiter zu rekrutieren, sondern auch für geeignete Fortbildungs- und Entwicklungsmöglichkeiten zu sorgen, um das Testteam zu erhalten und die besten Leistungen zu fördern. Abgesehen von den Fähigkeiten des Testteams, müssen Testmanager auch für die Erhaltung jener Fähigkeiten sorgen, die es ermöglichen, dass das Testteam in einem stressigen, hektischen Umfeld effektiv funktioniert. Dieses Kapitel beschreibt, wie man Fähigkeiten bewertet, Lücken füllt und dadurch ein synergetisches Testteam formt, das als solches zusammenhält und im Unternehmenskontext effektiv ist. Es wird beschrieben, wie man dieses Testteam motiviert und wie man effektiv kommuniziert.

7.2 Individuelle Fähigkeiten

Eine Person kann entweder durch Erfahrung oder durch Schulung in verschiedenen Arbeitsbereichen zum Testen von Software befähigt sein. Folgende Erfahrungen können als Wissensbasis dienen:

- Anwenden von Softwaresystemen
- Kenntnisse über den Geschäfts- oder Fachbereich
- Tätigkeiten in den verschiedenen Phasen des Softwareentwicklungsprozesses (einschließlich der Analyse, der Entwicklung und dem technischen Support)
- Tätigkeiten im Bereich des Softwaretestens

Anwender von Softwaresystemen sind vertraut mit der Nutzerseite der Systeme und haben gute Kenntnisse darüber, wie die Systeme angewendet werden, in welchen Bereichen Fehlerwirkungen die schwerwiegendsten Auswirkungen haben würden und welches Systemverhalten in bestimmten Situationen erwartet werden kann.

Fachexperten wissen, welche Bereiche für das Geschäft von größter Wichtigkeit sind und kennen den Einfluss dieser Bereiche auf die Fähigkeit des Unternehmens, die geschäftlichen Anforderungen zu erfüllen. Dieses Wissen lässt sich für die Priorisierung der Testaktivitäten, den Entwurf realistischer Testdaten bzw. Testfälle und die Verifizierung oder Erstellung von Anwendungsfällen nutzen.

Kenntnisse über den Softwareentwicklungsprozess (wie Anforderungsanalyse, Systemarchitektur, Entwurf und Programmierung) helfen zu verstehen, wie Fehlhandlungen zur Entstehung von Fehlerzuständen führen, wo Fehlerzustände zu finden sind und wie man ihnen vorbeugen kann. Erfahrung im technischen Support liefert zudem Wissen über die Praxis der Nutzer sowie über ihre Erwartungen und Anforderungen an die Benutzbarkeit. Erfahrung im Bereich der Softwareentwicklung ist zudem wichtig für den Einsatz von anspruchsvollen Testunterstützungswerkzeugen (die Programmier- und Entwurfsspezialisten erfordern) sowie für das Mitarbeiten bei statischen Analysen, Code-Reviews, Unit-Tests und technischen Aspekten des Integrationstests.

Zu den spezifischen Fähigkeiten beim Softwaretesten gehören die, die im Foundation Level Lehrplan sowie in den Advanced Level Test Analyst und Technical Test Analyst Lehrplänen behandelt werden. Hierzu gehören z.B. die Fähigkeit zur Analyse von Spezifikationen, die Teilnahme an Risikoanalysen, der Entwurf von Testfällen sowie Fleiß und Sorgfalt beim Durchführen der Tests und beim Aufzeichnen der Testergebnisse.

Besonders für Testmanager sind Wissen, Fähigkeiten und Erfahrung im Projektmanagement wichtig, da das Testmanagement viele Projektmanagementtätigkeiten beinhaltet. Typische Fähigkeiten sind z.B. das Erstellen von Plänen, die Überwachung und Kontrolle des Fortschritts, sowie das Berichten an die Projektbeteiligten. Wenn es keinen Projektmanager gibt, nimmt der Testmanager zusätzlich

zum Testmanagement auch die Rolle des Projektmanagers wahr (besonders in den späteren Projektphasen). Diese Fähigkeiten gehen jedoch über die im Foundation Level Lehrplan und im vorliegenden Lehrplan beschriebenen Fähigkeiten hinaus.

Neben den technischen Fähigkeiten sind zwischenmenschliche Fähigkeiten, wie der Umgang mit konstruktiver Kritik, Einflussnahme und Verhandlungsgeschick, wichtig für die Rolle des Testers. Technisch kompetente Tester können ihrer Rolle als Tester kaum gerecht werden, wenn sie die nötigen zwischenmenschlichen Fähigkeiten nicht haben und einsetzen. Erfolgreiche Tester müssen jedoch nicht nur mit anderen effektiv zusammenarbeiten können, sondern müssen auch ihre Arbeit gut organisieren können, ein Auge fürs Detail haben und ausgeprägte kommunikative Fähigkeiten besitzen (sowohl schriftlich als auch mündlich).

Im idealen Testteam gibt es eine gute Mischung von unterschiedlichen Fähigkeiten und Erfahrungen. Die Testteammitglieder sollten bereit und fähig sein, sich gegenseitig zu schulen und voneinander zu lernen. In manchen Situationen sind einige Fähigkeiten wichtiger und anerkannter als andere. In einem technischen Testumfeld, in der API-Test und Programmierfähigkeiten benötigt werden, werden technische Fähigkeiten mehr wertgeschätzt als z.B. Kenntnisse über den Fachbereich. Beim Black-Box-Test hingegen wird die Erfahrung des Fachbereichs für wertvoller erachtet. Es ist wichtig daran zu denken, dass sich das Umfeld und die Projekte ändern können.

Beim Erstellen eines "Skills Assessment" Fragebogens sollten Testmanager alle Fähigkeiten auflisten, die für die Arbeit wichtig sind und der Position entsprechen. Wenn alle Fähigkeiten Punkt für Punkt aufgelistet sind, können die einzelnen Testteammitglieder anhand eines Punktesystems bewertet werden (z.B. auf einer Bewertungsskala von 1 bis 5, wobei 5 die höchste für den betroffenen Bereich zu erwartende Fähigkeitsstufe ist). So ist es möglich, einzelne Personen zu bewerten und deren Stärken und Schwächen zu bestimmen; basierend auf diesen Informationen können dann Einzel- oder Gruppenschulungspläne erstellt werden. Testmanager können Leistungsziele für einzelne Personen setzen, damit diese ihre Fähigkeiten in bestimmten Bereichen verbessern und sie sollten spezifische Kriterien definieren, die zur Bewertung der Fähigkeiten Einzelner angewendet werden.

Mitarbeiter sollten für den langfristigen Einsatz eingestellt werden und nicht allein für den Beitrag, den sie bei einem einzigen Projekt leisten können. Wenn Testmanager in die Tester investieren und ein Umfeld für kontinuierliches Lernen schaffen, dann werden die Testteammitglieder dadurch motiviert, ihre Fähigkeiten und ihr Wissen zu steigern, um für die nächste Aufgabe vorbereitet zu sein.

Es wird kaum vorkommen, dass Testmanager die „perfekten“ Testteammitglieder einstellen können. Und selbst wenn die Testteammitglieder für das aktuelle Projekt genau passen, dann ist nicht gesagt, dass diese Mischung für das nächste Projekt genauso passend sein wird. Testmanager sollten Mitarbeiter einstellen, die intelligent, neugierig, anpassungsfähig und leistungsbereit sind, die effektiv im Team arbeiten können und die lernbereit und lernfähig sind. Auch wenn die Zusammenstellung eines perfekter Testteams wahrscheinlich nicht möglich ist, lässt sich ein starkes Testteam auch bilden, indem man die Stärken und Schwächen der einzelnen Testteammitglieder ausgewogen miteinander kombiniert.

Mit Hilfe von "Skills Assessment" Fragebögen können Testmanager herausfinden, wo die Stärken und Schwächen des Testteams sind. Auf Basis dieser Informationen lässt sich ein Plan für Schulung und Kompetenzentwicklung erstellen. Es sollte mit den Schwächen begonnen werden, die sich auf die Effektivität und Effizienz des Testteams am stärksten auswirken; Testmanager sollten entscheiden, wie sie in diesen Bereichen vorgehen möchten. Eine Möglichkeit zur Kompetenzentwicklung ist das Schulen, wie z.B. die Teilnahme der Mitarbeiter an öffentlichen Schulungen, das Durchführen von internen Schulungen, die Entwicklung eigener Schulungen und der Einsatz von E-Learning Kursen. Eine andere Möglichkeit ist das Selbststudium, wie z.B. das Studieren von Büchern, die Teilnahme an Webinaren und die Internet-Recherche. Der Wissenstransfer innerhalb des Testteams ist ein weiterer Ansatz; z.B. indem eine Person, die eine Fähigkeit erlernen möchte, einer Person zugeordnet wird,

die diese Fähigkeit bereits besitzt, um gemeinsam eine Aufgabe zu bearbeiten, für die diese Fähigkeit benötigt wird, oder indem lokale Fachkräfte kurze Präsentationen über ihr jeweiliges Fachgebiet geben, usw. Das Mentoring ist ein ähnlicher Ansatz; hier werden Personen, die eine neue Rolle übernehmen, von einer erfahrenen Person betreut, die diese Rolle bisher ausgeführt hat. Diese Person fungiert dann als eine ständige Anlaufstelle für Ratschläge und Unterstützung. Abgesehen von den Schwächen sollten Testmanager aber auch daran denken, die in der Bewertung der Fähigkeiten identifizierten Stärken im Plan für Schulung und Kompetenzentwicklung zu nutzen. Mehr Informationen über Testteam-Entwicklungspläne, siehe [McKay07].

7.3 Dynamik im Testteam

Eine der wichtigsten Funktionen des Testmanagers in einem Unternehmen ist die Auswahl der Mitarbeiter, um daraus das bestmögliche Testteam zu bilden. Neben einer individuellen Befähigung für die Aufgabe sind viele Aspekte zu berücksichtigen. Wenn eine Person für das Testteam ausgewählt wird, ist auch die Dynamik im Testteam zu beachten. Wird diese Person die im Testteam vorhandenen Fähigkeiten und verschiedenen Persönlichkeiten sinnvoll ergänzen? Es kann Vorteile haben, wenn in einem Testteam sowohl unterschiedliche Persönlichkeiten als auch unterschiedliche technische Fähigkeiten zusammenkommen. Ein starkes Testteam kann mit mehreren Projekten unterschiedlicher Komplexität umgehen und zugleich die zwischenmenschlichen Beziehungen zu den anderen Mitgliedern im Projektteam erfolgreich gestalten.

Softwaretester stehen häufig unter Zeit- und Leistungsdruck. Die Terminpläne für die Softwareentwicklung werden oft, sogar über ein realistisches Maß hinaus, komprimiert. Stakeholder haben hohe, manchmal auch unangemessen hohe, Erwartungen an das Testteam. Testmanager müssen Mitarbeiter auswählen, die mit diesen stressigen Situationen und mit Frustrationen gut umgehen können und sich dabei auf die Arbeit konzentrieren können, auch wenn die Einhaltung der Terminvorgaben scheinbar unmöglich ist. Es ist die Aufgabe der Testmanager, mit den Terminplänen und Erwartungen umzugehen; trotzdem sollten sich Testmanager bewusst sein, dass der Druck auch für die Testteammitglieder spürbar ist. Wenn Testmanager Personal für das Testteam auswählen, ist dabei die Arbeitsumgebung ein wichtiger Gesichtspunkt und dass die ausgewählten Personen zu diesem Umfeld passen. Für ein Umfeld, in dem es viel Arbeit und wenig Zeit gibt, sollten Testmanager gezielt nach Mitarbeitern suchen, die ihre Aufgaben zuverlässig abschließen und sich selbständig nach weiteren Aufgaben erkundigen, sobald die aktuelle Aufgabe erledigt ist.

Personen arbeiten intensiver und sorgfältiger, wenn sie das Gefühl haben, dass sie respektiert und gebraucht werden. Die einzelnen Personen müssen verstehen, dass jede/jeder Einzelne ein wichtiges Testteammitglied ist und dass sie einen wichtigen Beitrag zum Erfolg des gesamten Testteams leisten. Wenn diese Dynamik im Testteam erreicht ist, dann erfolgt der Wissenstransfer im Testteam informell, die Testteammitglieder gleichen unterschiedliche Arbeitsbelastung selbst untereinander aus und dem Testmanager bleibt mehr Zeit für andere Themen.

Neue Testteammitglieder müssen schnell vom Testteam integriert werden und eine angemessene Betreuung und Unterstützung erhalten. Jede Person im Testteam sollte ihre definierte Rolle haben, die sich aus einer individuellen Beurteilung ergeben kann. Ziel ist es, dass jedes Mitglied als einzelne Person erfolgreich ist und gleichzeitig zum Erfolg des gesamten Testteams beiträgt. Das lässt sich einerseits durch passende Testteamrollen für die jeweiligen Persönlichkeitstypen erreichen und andererseits, indem man sowohl auf die vorhandenen Fähigkeiten des Einzelnen baut, als auch sein/ihr Portfolio an Fähigkeiten erweitert.

Bei der Entscheidung, welcher Mitarbeiter eingestellt wird bzw. wer ins Testteam kommt, kann eine objektive Bewertung der Fähigkeiten hilfreich sein. Dies kann durch Interviews, Tests des Kandidaten, Evaluierung von Arbeitsproben sowie durch Verifizieren von Referenzen erfolgen. Bei der Bewertung sollten folgende Fähigkeiten berücksichtigt werden:

Technische Fähigkeiten (Fachkompetenz) lassen sich nachweisen durch:

- Ableiten von Testfällen aus einem Anforderungsdokument
- Review von technischer Dokumentation (Anforderungen, Programmcode usw.)
- Erstellen von Reviewbemerkungen, die klar, verständlich und objektiv sind
- Anwenden unterschiedlicher Testverfahren, die für vorgegebene Testziele geeignet sind (z.B. Entscheidungstabellen zum Testen einer Menge von Geschäftsregeln)
- Bewerten einer Fehlerwirkung und deren genaue Dokumentation
- Verständnis für die Klassifizierung von Fehlerinformationen nachweisen
- Verständnis für die Grundursachen von Fehlerzuständen nachweisen
- Anwenden eines Werkzeugs zum Testen einer vorgegebenen API (einschließlich positiver und negativer Tests)
- Mit Hilfe von SQL-Abfragen Datenbankinformationen finden und ändern, um ein vorgegebenes Szenario zu testen
- Entwurf eines Testautomatisierungsframeworks, der mehrere Tests ausführt und die Testergebnisse sammelt
- Ausführen automatisierter Tests und Analyse von Fehlerwirkungen
- Erstellen von Testkonzepten/-spezifikationen
- Erstellen eines Testberichts, der eine Bewertung der Testergebnisse beinhaltet

Zwischenmenschliche Fähigkeiten (Sozialkompetenz) lassen sich nachweisen durch:

- Präsentation von Informationen über ein Testprojekt, das in Verzug ist
- Erklären eines Fehlerberichts gegenüber einem Entwickler, der der Meinung ist, dass kein Fehlerzustand vorliegt
- Schulung eines Kollegen
- Präsentation eines Problems gegenüber dem Management bezüglich eines Prozesses, der nicht effektiv ist
- Review eines von einem Kollegen entworfenen Testfalls und Präsentieren der Bemerkungen gegenüber dem Kollegen
- Interview eines Kollegen
- Positive Rückmeldung an einen Entwickler

Obwohl diese Liste nicht vollständig und die spezifischen Fähigkeiten für unterschiedliche Umgebungen und Unternehmen variieren, so handelt es sich bei den aufgelisteten Fähigkeiten um solche, die in der Regel benötigt werden. Durch stellen effektiver Interviewfragen und das Demonstrierenlassen von Fähigkeiten des Kandidaten kann der Testmanger die Fähigkeiten des Kandidaten bewerten sowie dessen Stärken und Schwächen identifizieren. Sobald gute Bewertungsvorlagen erstellt sind, sollten diese auch für das bestehende Personal eingesetzt werden, um die Themengebiete für Weiterentwicklungs- und Schulungsmaßnahmen zu festzulegen.

Neben den Fähigkeiten, die von den einzelnen Testteammitgliedern verlangt werden, müssen Testmanager über exzellente Kommunikationsfähigkeiten und diplomatisches Geschick verfügen. Testmanager müssen in der Lage sein, kontroverse Situationen zu entschärfen, sie müssen die richtigen Kommunikationsmittel situationsgerecht einsetzen und sie müssen sich darauf konzentrieren, die Beziehungen innerhalb des Unternehmens aufzubauen und zu erhalten.

7.4 Testen in der Organisationsstruktur etablieren

Unternehmen ordnen das Testen sehr unterschiedlich in ihre Organisationsstruktur ein. Qualität gehört zwar während des gesamten Softwarelebenszyklus in die gemeinsame Verantwortung aller, ein unabhängiges Testteam kann aber entscheidend zu einem hochwertigen Produkt beitragen. In der

Praxis ist das Testen in unterschiedlichem Maße unabhängig, wie die folgende Auflistung zeigt, in der Reihenfolge von der geringsten zur höchsten Unabhängigkeit:

- Keine unabhängigen Tester
  - Keine Unabhängigkeit der Tester, der Entwickler testet seinen eigenen Code
  - Falls ihm Zeit für das Testen zur Verfügung steht, wird der Entwickler ermitteln, ob das Programm wie von ihm beabsichtigt funktioniert. Ob damit die tatsächlichen Anforderungen erfüllt sind oder nicht, bleibt offen.
  - Der Entwickler kann aufgedeckte Fehlerzustände schnell korrigieren.
- Ein Entwickler, der den Code nicht geschrieben hat, testet
  - Es gibt nur wenig Unabhängigkeit zwischen Entwickler und Tester.
  - Ein Entwickler, der den Programmcode eines anderen Entwicklers testet, wird Fehlerzustände möglicherweise ungern berichten.
  - Die Denkweise eines Entwicklers beim Testen führt meist zu einem Fokus auf positive Testfälle.
- Ein Tester (oder Testteam) aus dem Entwicklungsteam testet.
  - Tester (oder Testteam) können an das Projektmanagement oder den Entwicklungsmanager berichten
  - Die Denkweise eines Testers führt meist zu einem Fokus darauf, die Einhaltung von Anforderungen zu verifizieren.
  - Weil der Tester Mitglied des Entwicklungsteams ist, kann er zusätzlich Entwicklungsaufgaben haben.
  - Der Manager des Testers ist möglicherweise mehr auf seinen Terminkalender fokussiert als auf die zu erreichenden Qualitätsziele.
  - Im Testteam hat Softwaretesten möglicherweise einen niedrigeren Status als die Softwareentwicklung.
  - Der Tester hat möglicherweise keine Befugnisse, die Qualitätsziele beziehungsweise deren Einhaltung zu beeinflussen.
- Tester kommen aus den Fachabteilungen des Unternehmens, aus dem Kreis der Nutzer oder aus einer anderen technischen Organisation, die nicht mit Softwareentwicklung befasst ist.
  - Testergebnisse werden objektiv an die Stakeholder berichtet.
  - Qualität ist das Hauptziel dieses Testteams.
  - Die Kompetenzentwicklung und die Ausbildung der Mitarbeiter konzentrieren sich auf das Testen.
  - Softwaretesten wird als Karriereweg betrachtet
  - Es gibt dedizierte Führungskräfte, deren Fokus auf Qualitätsthemen liegt und die für das Management der Testgruppe zuständig sind.
- Externe Testexperten führen spezielle Testarten durch.
  - Für spezielle Testarten werden Experten eingesetzt, da allgemeines Testen (oder Testwissen) oft nicht ausreicht.
  - Mögliche Testarten sind Benutzbarkeit, Sicherheit, Performanz oder sonstige Bereiche, in denen eine Spezialisierung notwendig ist.
  - Qualität sollte im Vordergrund dieser Experten stehen, allerdings ist die Sichtweise durch ihre Spezialisierung auf die betroffenen Bereiche eingeschränkt. Ein Produkt, dessen Performanz gut ist, erfüllt nicht unbedingt die funktionalen Produktanforderungen; dies wird aber von einem Performanz-Testexperten möglicherweise nicht aufgedeckt.
- Ein externes Unternehmen testet.
  - Dieses Modell bietet maximale Unabhängigkeit zwischen Tester und Entwickler.
  - Der Wissenstransfer könnte nicht ausreichend sein, damit der Tester kompetent testen kann.
  - Es sind eindeutige Anforderungen und eine gut definierte Kommunikationsstruktur erforderlich.
  - Die Qualität der externen Unternehmenen muss regelmäßig in einem Audit bewertet werden.

Die Liste basiert auf dem typischen Fokus der einzelnen Personen; dies trifft jedoch möglicherweise nicht auf eine bestimmte Organisation zu. Position und Titel legen nicht unbedingt den Fokus einer Person fest. So können Entwicklungsmanager ebenso wie gute Testmanager qualitätsorientiert sein. Unabhängige Testmanager könnten an ein Management berichten, das an Terminplänen orientiert ist und sich daher selbst wiederum mehr auf Terminpläne als auf die Produktqualität konzentrieren. Um die beste Einordnung eines Testteams innerhalb der Organisationsstruktur festlegen zu können, müssen Testmanager die Unternehmensziele verstehen.

Der Grad der Unabhängigkeit zwischen der Entwicklungs- und Testorganisationen variiert stark. Es ist wichtig zu verstehen, dass ein Mehr an Unabhängigkeit mit mehr Isolation und weniger Wissenstransfer einhergehen kann. Ein geringeres Maß an Unabhängigkeit kann das Wissen über das System verbessern, es kann aber auch zu Interessenskonflikten durch widersprüchliche Zielsetzungen führen. Der Grad der Unabhängigkeit wird auch vom verwendeten Softwareentwicklungsmodell bestimmt: Bei einer agilen Entwicklungsmethode sind Tester beispielsweise meist Teil des Entwicklungsteams.

Die oben erwähnten Alternativen können in einem Unternehmen gemischt vorkommen. Es kann innerhalb der Entwicklungsabteilung getestet werden und zusätzlich auch durch eine unabhängige Testabteilung; abschließend wäre aucheine Zertifizierung durch ein externes Unternehmen möglich. Es ist wichtig, Zuständigkeiten und Erwartungen für die einzelnen Phasen des Testens zu verstehen und die Anforderungen an sie so zu stellen, dass sich die bestmögliche Qualität des Endprodukts im Rahmen der zeitlichen und finanziellen Grenzen erzielen lässt.

7.5 Motivieren

Es gibt viele Möglichkeiten, die einzelnen Testmitarbeiter zu motivieren, darunter

- Anerkennung für die bewältigten Aufgaben
- positive Rückmeldung durch das Management
- Respekt im Projektteam und in der Gruppe
- Übertragung von mehr Verantwortung und Selbstständigkeit
- angemessene Anerkennung der geleisteten Arbeit (einschließlich Gehalt, mehr Verantwortung und Respekt)

Bestimmte Projekteinflüsse können die Anwendung dieser Anreize erschweren. So arbeitet ein Tester möglicherweise sehr hart an einem Projekt mit einem unmöglichen Endtermin. Er oder sie kann dann alles Menschenmögliche unternehmen (Überstunden, Mehrarbeit), um die Qualitätsziele im Testteam voranzutreiben, aber das Produkt wird aufgrund externer Einflüsse vorzeitig ausgeliefert und kann trotz aller Bemühungen des Testers eine schlechte Qualität haben. Solche Vorkommnisse können demotivieren, vor allem, wenn der Einsatz des Testers nicht verstanden und anerkannt wird, unabhängig davon, ob das Endprodukt erfolgreich ist.

Das Testteam muss geeignete Metriken erheben, um nachzuweisen, dass bei Durchführung des Testens gute Arbeit geleistet, Risiken minimiert und Ergebnisse richtig protokolliert wurden. Solange diese Informationen nicht erfasst und mitgeteilt werden, kann ein Testteam bei fehlender Anerkennung für gute Arbeit leicht demotiviert werden.

Anerkennung zeigt sich nicht nur durch immaterielle Werte, wie Respekt und Zustimmung, sie wird auch durch die Chance auf eine Beförderung, größere Verantwortung oder Gehaltserhöhung aufgrund besonderer Leistungen wahrnehmbar. Wird das Testteam nicht wertgeschätzt, können solche Chancen fehlen.

Anerkennung und Respekt gewinnen die Tester, wenn es offensichtlich wird, dass sie den Mehrwert eines Projekts steigern. Bei einem einzelnen Projekt lässt sich das am besten erreichen, indem die Tester bereits von Projektbeginn an beteiligt werden und dies über den gesamten Softwarelebenszyklus so bleibt. Mit der Zeit werden die Tester Anerkennung und Respekt der anderen Projekt-Stakeholder für ihren Beitrag zur positiven Entwicklung des Projekts erhalten; ihr Beitrag sollte aber auch in Anbetracht der geringeren Kosten für die Qualität und der Risikobeherrschung quantifiziert werden.

Testmanager spielen eine bedeutende Rolle beim Motivieren einzelner Mitglieder des Testteams, sowie als Interessenvertreter des Testteams innerhalb einer größeren Organisation. Testmanager sollten die Leistungen einzelner Tester anerkennen, müssen jedoch auch Fehlerzustände fair und angemessen bewerten. Faire und konsequente Testmanager werden das Testteam dadurch motivieren, dass sie mit gutem Beispiel vorangehen.

7.6 Kommunikation

Die Kommunikation im Testteam findet vorwiegend in folgender Form statt:

- Dokumentation der Testmittel: Teststrategie, Testkonzept, Testfälle, Testberichte, Fehlerberichte usw.
- Review-Feedback zu geprüften Dokumenten: Anforderungen, Funktionsspezifikationen, Anwendungsfälle, Komponententestdokumentation usw.
- Sammeln und Verteilen von Informationen: Interaktion mit Entwicklern, anderen Testteammitgliedern, Management usw.

Damit das Testteam von anderen nachhaltig respektiert wird, muss die Kommunikation zwischen Testmitarbeitern und anderen Stakeholdern immer professionell, objektiv und effektiv sein. Wenn Tester anderen Beteiligten Rückmeldungen, vor allem konstruktive Kritik, zu deren Arbeitsergebnissen geben, brauchen sie Fingerspitzengefühl und Objektivität. Des Weiteren sollte Kommunikation immer auf das Erreichen der Testziele fokussiert sein und darauf, die Qualität (sowohl die Qualität des Produkts als auch die Qualität der Prozesse für die Herstellung der Softwaresysteme) zu verbessern.

Testmanager kommunizieren mit einem breiten Personenkreis, wie Nutzern, Projektteammitgliedern, Management, externen Testgruppen und Kunden. Die Kommunikation muss für die jeweiligen Adressaten effektiv sein. So kann beispielweise ein Bericht für die Entwickler, der die Trends und Tendenzen in Menge und Schweregraden der gefundenen Fehlerzustände aufzeigt, für eine Managementpräsentation auf Vorstandsebene zu detailliert und damit ungeeignet sein. Bei jeder Kommunikation, aber ganz besonders bei Präsentationen, ist es wichtig, die eigentliche Botschaft zu verstehen und wie diese Botschaft aufgenommen und verstanden werden kann, sowie den Erklärungsbedarf, der notwendig sein wird, um die Voraussetzung dafür zu schaffen, dass die Botschaft akzeptiert wird. Da Testmanager häufig Informationen zum Projektstatus präsentieren, ist es wichtig, dass diese Informationen den richtigen Detaillierungsgrad haben und in einer klaren und verständlichen Form präsentiert werden (z.B. einfache Diagramme und farbige Grafiken). Beispielsweise wollen Manager lieber über Fehlertendenzen als über einzelne Fehlerzustände informiert werden. Durch effiziente Kommunikation lässt sich die Aufmerksamkeit der Adressaten aufrechterhalten und trotzdem die richtige Botschaft übermitteln. Testmanager sollten jede Präsentation als eine Gelegenheit verstehen, Qualität und Qualitätsprozesse zu fördern.

Testmanager kommunizieren nicht nur mit Personen außerhalb der Abteilung (Kommunikation nach außen). Ein wichtiger Teil der Kommunikation der Testmanager ist es, effektiv innerhalb der Testgruppe zu kommunizieren (Kommunikation nach innen), um Neuigkeiten, Anweisungen, veränderte Prioritäten und sonstige, in Zusammenhang mit dem normalen Testprozess übliche Informationen, weiterzugeben. Testmanager kommunizieren außerdem mit bestimmten Personen

sowohl nach oben (Kommunikation nach oben) als auch nach unten (Kommunikation nach unten) in der Unternehmenshierarchie. Unabhängig von der Kommunikationsrichtung gelten jedoch dieselben Regeln; die Kommunikation sollte für die jeweiligen Adressaten angemessen sein, die Botschaft sollte effektiv transportiert werden und es sollte sichergegangen werden, dass die Botschaft verstanden wurde.

Testmanager müssen sich mit den unterschiedlichen Kommunikationsmitteln gut auskennen. Viele Informationen werden per E-Mail, mündlich, in formellen oder informellen Besprechungen und sogar durch den Einsatz von Testmanagementwerkzeugen, wie Fehlermanagementwerkzeugen, ausgetauscht. Kommunikation muss immer professionell und objektiv sein. Korrekturlesen, sowohl für Qualität als auch für Inhalt, ist ein notwendiger Arbeitsschritt; auch für die dringendste Kommunikation. Schriftliche Kommunikation existiert häufig noch lange nach dem eigentlichen Projekt. Es ist daher wichtig, dass Testmanager Dokumentationen in professioneller Qualität erstellen, die für ein Unternehmen stehen, welches Qualität fördert.

8. Referenzen

8.1 Standards

- [BS-7925-2] BS 7925-2 Software Component Testing Standard, Chapter 2
- [FDA21] FDA Title 21 CFR Part 820, Food and Drug Administration, USA
- [IEEE829] IEEE Standard for Software and System Test Documentation
- [IEEE1028] IEEE Standard for Software Reviews and Audits, Chapter 2
- [IEEE1044] IEEE Standard Classification for Software Anomalies, Chapter 4
- [ISO9126] ISO/IEC 9126-1:2001, Software Engineering – Software Product Quality, Chapters 2 and 4
- [ISO12207] ISO 12207, Systems and Software Engineering, Software Live Cycle Processes, Chapter 2
- [ISO15504] ISO/IEC, Information Technology – Process Assessment, Chapter 2
- [ISO25000] ISO/IEC 25000:2005, Software Engeneering – Software Product Quality, Requirements and Evaluation (SQualRE), Chapters 2 and 4;
- [RTCA DO-178B/ED-12B] Software Considerations in Airborne Systems and Equipment Certification, RTCA/EUROCAE ED12B, 1992; Chapter 2.

8.2 ISTQB Documents

- [ISTQB_AL_OVIEW] ISTQB Advanced Level Overview, Version 1.0.
- [ISTQB_ALTM_SYL] ISTQB Advanced Level Test Manager Syllabus, Version 1.0.
- [ISTQB ALTTA SYL] ISTQB Advanced Level Technical Test Analyst Syllabus, Version 1.0.
- [ISTQB ETM SYL] ISTQB Expert Test Management Syllabus, Version 1.0
- [ISTQB_FL_SYL] ISTQB Foundation Level Syllabus 2011
- [ISTQB_GLOSSARY] Standard glossary of terms used in Software Testing, Version 2.2, 2012.
- [ISTQB ITP SYL9 ISTQB Expert Improving the Test Process Syllabus; Version 1.0

8.3 Eingetragene Marken

Die folgenden eingetragenen Marken und Dienstleistungsmarken werden im vorliegenden Dokument verwendet:

- CMMI®, IDEAL[SM], ISTQB®, ITIL®, PRINCE2®, TMMi®, TPI Next®

- CMMI ist eingetragen beim U.S. Patent and Trademark Office von der Carnegie Mellon University.
- IDEAL ist eine Dienstleistungsmarke des Software Engineering Institute (SEI), Carnegie Mellon University.
- ISTQB ist ein eingetragenes Warenzeichen des International Software Testing Qualifications Board.
- ITIL ist ein eingetragenes Warenzeichen, sowie eine eingetragene Gemeinschaftsmarke des Office of Government Commerce und ist außerdem eingetragen beim U.S. Patent and Trademark Office.
- PRINCE2 ist ein eingetragenes Warenzeichen des britischen „Cabinet Office“.
- TMMi ist ein eingetragenes Warenzeichen der TMMi Foundation.
- TPI Next ist ein eingetragenes Warenzeichen der Sogeti Nederland B.V.

8.4 Literatur

[Black03]: Rex Black, “Critical Testing Processes,” Addison-Wesley, 2003, ISBN 0-201-74868-1

[Black09]: Rex Black, “Managing the Testing Process, third edition,” John Wiley & Sons: New York, 2009, ISBN 0-471-22398-0

[Black11]: Rex Black, Erik van Veenendaal, Dorothy Graham, “Foundations of Software Testing,” Thomson Learning, 2011, ISBN 978-1-84480-355-2

[Craig02]: Rick Craig, Stefan Jaskiel, “Systematic Software Testing,” Artech House, 2002, ISBN 1-580-53508-9

[Crispin09]: Lisa Crispin, Janet Gregory, “Agile Testing: A Practical Guide for Testers and Agile Teams”, Addison-Wesley, 2009, ISBN 0-321-53446-8

[de Vries09]: Gerrit de Vries, et al., “TPI Next”, UTN Publishers, 2009, ISBN 90-72194-97-7

[Goucher09]: Adam Goucher, Tim Riley (editors), “Beautiful Testing,” O’Reilly, 2009, ISBN 978-0596159818

[IDEAL96]: Bob McFeeley, “IDEAL: A User's Guide for Software Process Improvement,” Software Engineering Institute (SEI), 1996, CMU/SEI-96-HB-001

[Jones07] Capers, Jones:”Estimating Software Costs, 2nd edition”, McGraw-Hill, 2007, ISBN 978-0071483001

[Jones11]: Capers Jones and Olivier Bonsignour, “Economics of Software Quality,” Pearson, 2011, ISBN 978-0132582209

[McKay07]: Judy McKay, “Managing the Test People,” Rocky Nook, 2007, ISBN 978-1933952123

[Musa04]: John Musa, “Software Reliability Engineering, second edition,” Author House, 2004, ISBN 978-1418493882

[Stamatis03]: D.H. Stamatis, “Failure Mode and Effect Analysis,” ASQC Quality Press, 2003, ISBN 0-873-89300

[vanVeenendaal11] Erik van Veenendaal, “The Little TMMi,” UTN Publishers, 2011, ISBN 9-490-986038

[vanVeenendaal12] Erik van Veenendaal, “Practical Risiko-based Testing,” UTN Publishers, 2012, ISBN 978-9490986070

[Whittaker09]: James Whittaker, “Exploratory Testing,” Addison-Wesley, 2009, ISBN 978-0321636416

[Wiegers03]: Karl Wiegers, “Software Requirements 2,” Microsoft Press, 2003, ISBN 978-0735618794

8.5 Sonstige Referenzen

Die folgenden Referenzen verweisen auf Informationen im Internet. Diese Referenzen wurden zum Zeitpunkt der Veröffentlichung dieses Advanced Level Lehrplans geprüft.

http://www.istqb.org
http://www.sei.cmu.edu/cmmi/
http://www.tmmi.org/
http://www.tpinext.com/

8.6 Ergänzende deutschsprachige Literatur

[Spillner11] Andreas Spillner / Thomas Roßner / Mario Winter / Tilo Linz, „Praxiswissen Softwaretest – Testmanagement“, 3. überarbeitete u. erweiterte Auflage, Seiten, April 2011, dpunkt.verlag. ISBN 978-3-89864-746-5 [Kommentar: basiert auf CTAL Syllabus 2007]

[Spillner13] Andreas Spillner / Thomas Roßner / Mario Winter / Tilo Linz, „Praxiswissen Softwaretest – Testmanagement“, 4. überarbeitete u. erweiterte Auflage, Seiten, April 2013, dpunkt.verlag. ISBN 978-3-89864-746-5 [Kommentar: basiert auf CTAL Syllabus 2012]

[ISTQB_GLOSSARY_DE] "ISTQB®/GTB Standardglossar der Testbegriffe - Deutsch/Englisch", Version 2.2, 2013, German Testing Board e.V. (basiert auf [ISTQB_GLOSSARY])

9. Index

Abschluss der Testaktivitäten 8, 16
aggregierte Risikoprioriätszahl 29
agile 40, 42
Anomalie 60
Arten der Prozessverbesserung 69
Audit 53
Berichterstattung 24
Bewertung von Testendekriterien und Bericht 16
Capability Maturity Model Integration 67
CMMI 51
Critical Testing Processes 67
CTP 68, 69, 71
Dynamik im Testteam 84
Fachkompetenz 85
falsch negatives Ergebnis 60
falsch positives Ergebnis 60
Fehlerbaumanalyse (FTA) 32
Fehlereindämmung innerhalb der Phase 60
Fehlermanagementausschuss 60
Fehlertriage 63
Fehlerwirkung 60
Fehlerzustand 60
Gefährdungsanalyse 32
Grundursache 60
Hardware-Software-Integrationtest 24
IDEAL 69
IEEE 50
IEEE 1028 50
IEEE 829 50
Individuelle Fähigkeiten 82
informelles Review 53
Inspektion 53
Interaktionstest der Funktionen 24
ISO 50
ITIL 51
Kommunikation 88
Kosten einer Gefährdung 32
Kosten für die Aufdeckung 49
Kosten für externe Fehlerwirkungen 49
Kosten für interne Fehlerwirkungen 49
Lebenszyklus
  agile 23
  iterativ 22
Leitender Testmanager 18
Management von formalen Reviews 59
Managementreview 53
Mastertestkonzept 18, 39
Metriken für Reviews 58
Moderator 53
Motivieren 87
Personenbezogene Metriken 43
Phasentestplan 36
Planungsrisiko 26
PMI 51
PRINCE2 51
Priorität 60
Produktintegrationstest beim Kunden 24
Produktrisiko 18
Projektrisiko 18
Projekttestkonzept 36
Prozessmetriken 43
Qualitätsrisiko 18
Quality Function Deployment (QFD) 32
Rentabilität (ROI) 75
Review 53
Review auf Zweideutigkeit 34
Reviewer 53
Reviewplan 53
Reviews 54
Risiko 18
Risikoanalyse 18, 29
Risikoassessment 18
Risikobeherschung 29
Risikoidentifikation 18
Risikoidentifizierung 27
Risikomanagement 18, 27
Risikoorientiertes Testen 18, 26, 27, 31, 34
Risikoprioritätszahl 29
Risikostufe 18
Riskobeherrschung 18
Schweregrad 60
service level agreements (SLAs) 10
Sozialkompetenz 85
Standards
  CTP 68
  DO-178B 51
  ED-12B 51
  IEEE 829 8
  STEP 68
  TMMi 68
STEP 68, 69, 72
Strategien
  beratend 38
  Prozess- oder standardkonform 38
  Regressionstest 38
Stufentestkonzept 18, 40
Systematischer Test- und Bewertungsprozess 67
Systemintegrationstest 24
Teamzusammensetzung 81
technisches Review 53

Technisches Risiko 28
Test Maturity Model Integration 67
Test Maturity Model Integration (TMMi) 70
Testanalyse 12
Testaufwandsschätzung 41
Testbedingungen 8, 11, 12, 18
Testbericht 8
Testdurchführung 8, 15
Testen in der Organisationsstruktur etablieren 86
Testendekriterien 8
TestendekriterienEndekriterien 8
Testen-in-der-Breite 30
Testentwurf 8, 13
Testfall 8
Testkonzept 18
Testleiter 18
Testmanagement 18, 20
Testplanung 8
Testplanung, Testüberwachung und Teststeuerung 9
Testplanung, zeitliche 18
Testprotokoll 8
Testprozess mit CTP verbessern 71
Testprozess mit STEP verbessern 72
Testprozess mit TMMI verbessern 70
Testprozess mit TPI Next verbessern 71
Testprozess verbessern 69
Testrealisierung 8, 14
Testrichtlinie 18, 36
Testschätzung 18
Testskript 8
Teststeuerung 8, 18
Teststrategie 18, 37
Teststufe 18, 40
Testüberwachung 18
Testverbesserungsprozess 68
Testverfahren 24
Testvorgehensweise 18, 36
TMM 68
TMMi 69
TPI Next 67, 69
Ursache-Wirkungs-Graph-Analyse 34
Walkthrough 53
Werkzeuglebenszyklus 78
Werkzeugmetriken 79
Wideband Delphi 18