ISTQB® Certified Tester

Lehrplan

Advanced Level Test Management

Version 3.0

Deutschsprachige Ausgabe. Herausgegeben durch

Austrian Testing Board, German Testing Board e. V. & Swiss Testing Board

Übersetzung des englischsprachigen Lehrplans des International Software Testing Qualifications Board (ISTQB®), Originaltitel: Certified Tester Advanced Level Test Management Syllabus, Version 3.0

Urheberschutzvermerk

Zur besseren Lesbarkeit wird im gesamten Dokument auf die gleichzeitige Verwendung männlicher und weiblicher Sprachformen verzichtet. Es wird das generische Maskulinum verwendet, wobei unterschiedliche Geschlechter gleichermaßen gemeint sind.

Änderungsübersicht der deutschsprachigen Ausgabe

Version	Datum	Bemerkungen
ISEB v1.1	2001/09/04	ISEB Practitioner Syllabus
ISTQB® 1.2E	2003/09	ISTQB® Advanced Level Syllabus von EOQ-SG
V2007	2007/10/12	Certified Tester Advanced Level Syllabus Version 2007
D100626	2010/06/10	Einarbeitung der 2009 angenommenen Änderungen, Aufteilung der einzelnen Kapitel auf die einzelnen Module
D101227	2010/12/10	Akzeptieren der Änderungen am Format und Korrekturen, die sich nicht auf die Bedeutung der Sätze auswirken.
D2011	2011/10/31	Änderung des geteilten Lehrplans, überarbeitete LOs und Textänderungen zur Anpassung an die LOs. Hinzufügung von BOs.
Alpha 2012	2012/02/09	Einarbeitung aller Kommentare der NBs, die seit der Veröffentlichung im Oktober eingegangen sind.
Beta 2012	2012/03/26	Einarbeitung der fristgerecht eingegangenen Kommentare der NBs aus der Alpha-Version
Beta 2012	2012/04/07	Beta-Version bei GA eingereicht
Beta 2012	2012/06/08	Überarbeitete Version an die NBs freigegeben
Beta 2012	2012/06/27	Kommentare der EWG und des Glossars eingearbeitet
RC 2012	2012/08/15	Release Candidate Version – einschließlich endgültiger NB-Bearbeitungen
Beta v3.0	2023/10/31	Einarbeitung aller Kommentare von Member Boards, die für alle Sections (BOIncs) aus dem Alpha-Review eingegangen sind.
POST Beta v3.0	2024/01/31	Einarbeitung aller Kommentare von Member Boards, die für alle Sections (BOIncs) aus dem Beta-Review eingegangen sind.

POST Beta v3.0	2024/02/29	Geringfügige Änderungen vom Korrekturlesen
RC v3.0	2024/03/28	Release Candidate Version – letzte formale Template-Änderungen, wie von der PWG vorgeschlagen
v3.0	2024/05/03	Überarbeitung nach der Veröffentlichung; nur Typos und Inkonsistenzen wurden beseitigt.
BETA v3.0 GERM	2024/09/09	Lokalisierung der englischen Version
POST BETA v3.0 GERM	2024/12/20	Lokalisierung der englischsprachigen Version nach Einarbeitung der BETA-Befunde
V3.0 GERM	2025/01/18	Finalisierung nach Integration Feedback der BETA Reviewer

Inhaltsverzeichnis

Danksagung

Das englischsprachige Dokument wurde von der Generalversammlung des ISTQB® am 03. Mai 2024 formell freigegeben.

Es wurde von einem Team des International Software Testing Qualifications Board erstellt: Horst Pohlmann (Product Owner, Vice Chair AELWG), Tauhida Parveen, Francis Fenner, Laura Albert, Matthias Hamburg, Maud Schlich, Tanja Tremmel, Ralf Bongard, Erik van Veenendaal, Jan Giessen, Bernd Freimut, Andreas Neumeister, Georg Sehl, Rabih Arabi, Therese Kuhfuß, Ecaterina Irina Manole, Veronica Belcher, Kenji Onishi, Pushparajan Balasubramanian, Meile Postuma und Miroslav Renda.

Das Team dankt Gary Mogyorodi für sein Technisches Review (in Beta), Julia Sabatine für ihr Korrekturlesen, dem Review-Team und den nationalen Mitgliedboards für ihre Anregungen und Beiträge.

Die folgenden Personen waren am Review, der Kommentierung und der Abstimmung zu diesem Lehrplan beteiligt:

Alpha-Reviews: Benjamin Timmermans, Mattijs Kemmink, Rik Marselis, Jean-Francois Riverin, Gary Mogyorodi, Ralf Bongard, Ingvar Nordström, Yaron Tsubery, Imre Mészáros, Ádám Bíró, Ramit M Kaul, Chinthaka Indikadahena, Darvay Tamás Béla, Beata Karpinska, Young jae Choi, Stuart Reid, Tal Pe'er, Meile Posthuma, Daniel van der Zwan, Klaudia Dussa-Zieger, Jörn Münzel, Petr Neugebauer, Derk-Jan de Grood, Rik Kochuyt, Andreas Hetz, Laura Albert, Eszter Sebestyeni, Tamás Szőke, Henriett Braunné Bokor, Ágota Horváth, Péter Sótér, Ferenc Hamori, Paul Weymouth, Lloyd Roden, Kevin Chen, Huang qin, Pushparajan Balasubramanian, Szilard Szell, Tamas Stöckert, Lucjan Stapp, Adam Roman, Anna Miazek, Márton Siska, Erhardt Wunderlich, László Kvintovics, Murian Song, Mette Bruhn-Pedersen, Petra Schneider, Michael Stahl, Dilhan Jayakody, Francisca Cano Ortiz, Johan Klintin, Liang Ren, Ole Chr. Hansen, Zsolt Hargitai, Tamás Rakamazi, Kenji Onishi, Arnika Hryzszko, Rabih Arabi, Veronica Belcher und Vignesh Balasubramanian.

Beta-Reviews: Maria-Therese Teichmann, Dominik Weber, Thomas Puffler, Peter Kunit, Martin Klonk, Michaël Pilaeten, Wim Decoutere, Arda Ender Torçuk, Piet de Roo, Rik Marselis, Jakub Platek, Ding Guofu, Zheng Dandan, Liang Ren, Yifan Chen, Hallur Helmsdal, Ole Chr. Hansen, Klaus Skafte, Gitte Ottosen, Tanzeela Gulzar, Arne Becher, Klaudia Dussa-Zieger, Jan Giesen, Florian Fieber, Carsten Weise, Arnd Pehl, Matthias Hamburg, Stephanie Ulrich, Jürgen Beniermann, Márton Siska, Ádám Sterbinszky, Ágnes Srancsik, Marton Matyas, Tamas Stöckert, Csilla Varga, Zsolt Hargitai, Bíró Ádám, Ágota Horváth, Eszter Sebestyéni, Szilárd Széll, Péter Sótér, Giancarlo Tomasig, Nicola de Rosa, Kaiwalya Katyarmal, Pradeep Tiwari, Sreeja Padmakumari, Seunghee Choi, Stuart Reid, Dmitrij Nikolajev, Mantas Aniulis, Monika Stoecklein-Olsen, Adam Roman, Mahmoud Khalaili, Ingvar Nordström, Beata Karpinska, Armin Born, Ferdinand Gramsamer, Mergole Kuaté, Thomas Letzkus, Nishan Portoyan, Ainsley Rood, Lloyd Roden, Sarah Ireton.

Der Lehrplan Advanced Level Test Management 2010-2012 wurde von einem Kernteam der ISTQB®- Unterarbeitsgruppe – Advanced Test Manager – erstellt: Rex Black (Chair), Judy McKay (Vice Chair), Graham Bath, Debra Friedenberg, Bernard Homès, Paul Jorgensen, Kenji Onishi, Mike Smith, Geoff Thompson, Erik van Veenendaal und Tsuyoshi Yumoto.

Das Kernteam dankt dem Review-Team und den nationalen Boards für ihre Anregungen und Beiträge.

Zum Zeitpunkt der Fertigstellung des Advanced-Level-Lehrplans bestand die Advanced-Level-Arbeitsgruppe aus den folgenden Mitgliedern (in alphabetischer Reihenfolge):

Graham Bath, Rex Black, Maria Clara Choucair, Debra Friedenberg, Bernard Homès (stellvertretender 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 und Tsuyoshi Yumoto.

Die folgenden Personen haben am Review, der Kommentierung und der Abstimmung über diesen Lehrplan teilgenommen:

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

0 Einführung in diesen Lehrplan

0.1 Zweck dieses Dokuments

Dieser Lehrplan bildet die Grundlage der internationalen Qualifikation für Softwaretester Advanced Level für Test Management. Das German Testing Board e. V. (im Folgenden GTB genannt) hat diesen Lehrplan in Zusammenarbeit mit dem Austrian Testing Board (ATB) und dem Swiss Testing Board (STB) in die deutsche Sprache übersetzt. Das ISTQB® stellt den Lehrplan folgenden Adressaten zur Verfügung:

1. Nationalen Mitgliedboards, die den Lehrplan in ihre Sprache(n) übersetzen und Schulungsanbieter akkreditieren dürfen. Die nationalen Mitgliedboards dürfen den Lehrplan an die Anforderungen ihrer nationalen Sprache anpassen und Referenzen hinsichtlich lokaler Veröffentlichungen berücksichtigen.
2. Zertifizierungsstellen zur Ableitung von Prüfungsfragen in ihrer nationalen Sprache, die an die Lernziele dieses Lehrplans angepasst sind
3. Schulungsanbieter zur Erstellung von Lehrmaterialien und zur Bestimmung angemessener Lehrmethoden
4. Zertifizierungskandidaten zur Vorbereitung auf die Zertifizierungsprüfung (entweder als Teil einer Schulung oder unabhängig davon)
5. Der internationalen Software- und Systementwicklungs-Community zur Förderung des Berufsbildes des Software- und Systemtesters und als Grundlage für Bücher und Fachartikel

ATB, GTB, STB und ISTQB® können die Nutzung dieses Lehrplans auch anderen Personenkreisen oder Institutionen für andere Zwecke genehmigen, sofern diese vorab eine entsprechende schriftliche Genehmigung einholen.

0.2 Der Certified Tester Advanced Level Test Management

Diese Advanced-Level-Qualifikation richtet sich an alle, die mit dem Management von Softwaretests befasst sind. Dazu gehören Personen in Rollen wie Tester, Testberater, Testmanager, Benutzerakzeptanztester, Scrum Master, Projektmanager oder Product Owner. Die Qualifikation Advanced Level Test Management eignet sich auch für alle, die ein fortgeschrittenes Verständnis für das Testen von Software haben möchten, wie z. B. Projektmanager, Qualitätsmanager, Softwareentwicklungsmanager, Systemanalytiker (Businessanalysten), IT-Leiter und Unternehmensberater. Inhaber des Zertifikats Advanced Level Test Management können ISTQB®-Expert- Level-Qualifikationen im Bereich Softwaretest erwerben. Das ISTQB®-Zertifikat Certified Tester Advanced Level – Test Manager oder Test Management ist lebenslang gültig und muss nicht erneuert werden. Die Zertifizierung ist international anerkannt und weist die berufliche Kompetenz und Glaubwürdigkeit der Kandidaten im Testmanagement nach.

0.3 Karriereweg für Tester

Das ISTQB®-Schema unterstützt Testexperten in allen Stufen ihrer Karriere und bietet ihnen sowohl eine breite als auch eine tiefe Wissensbasis. Personen, die die ISTQB®-Zertifizierung Certified Tester Advanced Level Test Management erlangt haben, sind möglicherweise auch an anderen Core-Advanced-Level-Zertifizierungen (d. h. Test Analyst und Technical Test Analyst) und danach an ISTQB®-Expert-Level-Zertifizierungen (d. h. Test Management oder

Improving the Test Process) interessiert. Wer Fähigkeiten in der Testtätigkeit in einer agilen Umgebung entwickeln möchte, könnte die Zertifizierung Agile Test Leadership at Scale (ATLaS) in Betracht ziehen. Der Spezialistenstrang bietet einen tiefen Einblick in Bereiche, die spezifische Testansätze und Testaktivitäten beinhalten (z. B. KI-Tests oder Testen mobiler Apps) oder die das Test-Know-how für bestimmte Branchendomänen bündeln (z. B. Automotive oder Gaming).

Aktuelle Informationen über das ISTQB®-Certified-Tester-Schema finden Sie unter www.istqb.org oder auf den Seiten der nationalen Boards, wie z. B. www.gtb.de (Deutschland), www.austriantestingboard.at (Österreich) oder www.swisstestingboard.org (Schweiz).

0.4 Geschäftlicher Nutzen

In diesem Abschnitt werden 11 geschäftliche Nutzen (Business Outcomes, BO) aufgeführt, die von einer Person erwartet werden, die die Zertifizierung Advanced Level Test Management bestanden hat.

Ein im Advanced Level Test Management zertifizierter Tester kann ...

TM_01	… Tests in verschiedenen Softwareentwicklungsprojekten durch Anwendung von Testmanagementprozessen managen, die für das Projektteam oder die Testorganisation festgelegt wurden.
TM_02	… Stakeholder und Softwareentwicklungslebenszyklus-Modelle identifizieren, die in einem bestimmten Kontext relevant sind.
TM_03	… eine Risikoidentifizierung und Risikobewertung innerhalb eines Softwareentwicklungslebenszyklus organisieren und die Ergebnisse nutzen, um das Testen so zu steuern, dass die Testziele erreicht werden.
TM_04	… eine Teststrategie für das Projekt im Einklang mit der organisationsweiten Teststrategie und dem Projektkontext definieren.
TM_05	… Tests zur Erreichung der Projektziele kontinuierlich überwachen und steuern.
TM_06	… den Testfortschritt bewerten und an die Projektbeteiligten berichten.
TM_07	… die notwendigen Kompetenzen identifizieren und diese Kompetenzen im Team entwickeln.
TM_08	… einen Business Case für das Testen in verschiedenen Kontexten erstellen und präsentieren, der die Kosten und den erwarteten Nutzen darlegt.
TM_09	… Testprozessverbesserungsaktivitäten in Projekten oder Softwareentwicklungsprodukt-Streams leiten und zu organisatorischen Initiativen der Testprozessverbesserung beitragen.
TM_10	… die Testaktivitäten einschließlich der erforderlichen Testinfrastruktur planen und den für das Testen erforderlichen Aufwand schätzen.

TM_11	… Fehlerberichte erstellen und einen an den Softwareentwicklungslebenszyklus angepassten Fehlerworkflow definieren.

0.5 Prüfbare Lernziele und kognitive Stufen des Wissens

Die Lernziele (Learning Objectives, LO) unterstützen den geschäftlichen Nutzen und dienen zur Ausarbeitung der Prüfungen für die Zertifizierung Certified Tester Advanced Level Test Management.

Im Allgemeinen sind alle Inhalte der Kapitel 1-3 dieses Lehrplans auf K1-Niveau prüfbar, mit Ausnahme der Einleitung, der Referenzen, des Epilogs und der Anhänge.

Das heißt, vom Prüfling kann gefordert werden, einen Schlüsselbegriff oder ein Konzept aus einem der drei Kapitel wiederzuerkennen, sich daran zu erinnern oder wiederzugeben. Die Stufen der spezifischen Lernziele werden am Anfang jedes Kapitels genannt und wie folgt klassifiziert:

- K2: Verstehen
- K3: Anwenden
- K4: Analysieren

Weitere Einzelheiten und Beispiele für Lernziele finden Sie in Anhang A.

Alle Begriffe, die als Schlüsselbegriff direkt unter den Kapitelüberschriften aufgeführt sind, müssen bekannt sein (K1), auch wenn sie nicht ausdrücklich in den Lernzielen erwähnt werden.

0.6 Die Zertifikatsprüfung für das Advanced Level Test Management

Die Zertifikatsprüfung für das Advanced Level Test Management basiert auf diesem Lehrplan. Die Beantwortung der Prüfungsfragen kann die Nutzung von Inhalten aus mehr als einem Abschnitt dieses Lehrplans erfordern. Alle Abschnitte des Lehrplans sind prüfungsrelevant, mit Ausnahme der Einführung, der Anhänge und der Referenzen. Standards und Bücher sind als Referenzen genannt (Kapitel 7), ihr Inhalt ist jedoch nicht prüfungsrelevant, abgesehen von dem, was im Lehrplan selbst aus diesen Standards und Büchern zusammengefasst ist.

Weitere Einzelheiten entnehmen Sie bitte dem Dokument “Exam Structures and Rules Compatible with Syllabus Foundation and Advanced Levels and Specialists Modules”.

Das Einstiegskriterium für die Teilnahme an der Prüfung für das Zertifikat Advanced Level Test Management ist: Die Kandidaten müssen das ISTQB®-Foundation-Level-Zertifikat besitzen, bevor sie die Zertifikatsprüfung für das Advanced Level Test Management ablegen können. Es wird jedoch dringend empfohlen, dass der Kandidat zumindest ein Minimum an Erfahrung in der Softwareentwicklung oder im Softwaretesten mitbringt, z. B. sechs Monate Erfahrung als Tester oder als Softwareentwickler.

0.7 Akkreditierung

Ein nationales ISTQB®-Mitgliedboard kann Schulungsanbieter akkreditieren, deren Lehrmaterial diesem Lehrplan entspricht. Die Akkreditierungsrichtlinien können bei diesem nationalen Board (in Deutschland: German Testing Board e. V.; in der Schweiz: Swiss Testing Board; in Österreich: Austrian Testing Board) oder bei einer der Organisationen bezogen werden, die die Akkreditierung im Auftrag des nationalen Boards durchführt. Eine akkreditierte

Schulung ist als konform mit diesem Lehrplan anerkannt und darf eine ISTQB®-Prüfung als Teil der Schulung enthalten. Die Akkreditierungsrichtlinien für diesen Lehrplan folgen den allgemeinen Akkreditierungsrichtlinien, die von der ISTQB®-Arbeitsgruppe "Processes Management and Compliance" veröffentlicht wurden.

0.8 Umgang mit Normen und Standards

Im Lehrplan für Advanced Level Test Management wird auf einige Normen und Standards verwiesen (z. B. ISO, IEC und IEEE). Diese Verweise dienen als Rahmen oder als Quelle für zusätzliche Informationen, falls der Leser dies wünscht. Bitte beachten Sie, dass der Lehrplan diese Normen und Standards als Referenzen verwendet. Die Inhalte der Normen und Standards sind nicht prüfungsrelevant. Weitere Informationen über Normen und Standards sind in Kapitel 4 nachlesbar.

0.9 Auf dem Laufenden bleiben

Die Softwarebranche verändert sich schnell. Um diesen Veränderungen Rechnung zu tragen und den Beteiligten Zugang zu relevanten und aktuellen Informationen zu verschaffen, haben die ISTQB®-Arbeitsgruppen auf der Website www.istqb.org Links angelegt, die auf unterstützende Dokumentation und Änderungen von Standards verweisen. Diese Informationen sind im Rahmen des Lehrplans für Advanced Level Test Management nicht prüfungsrelevant.

0.10 Detaillierungsgrad

Der Detaillierungsgrad dieses Lehrplans erlaubt international einheitliche Schulungen und Prüfungen. Um dieses Ziel zu erreichen, enthält der Lehrplan Folgendes:

- Allgemeine Lehrziele, die die Intention des Lehrplans für Advanced Level Test Management beschreiben
- Eine Liste von Begriffen (Schlüsselbegriffe), die die Lernenden abrufen können müssen
- Lernziele für jeden Wissensbereich, die die zu erreichenden kognitiven Lernergebnisse beschreiben
- Eine Beschreibung der wichtigsten Konzepte, einschließlich Verweisen auf Quellen wie empfohlene Literatur oder Standards

Der Inhalt des Lehrplans ist keine Beschreibung des gesamten Wissensgebiets „Softwaretesten“. Er spiegelt den Detaillierungsgrad wider, der in Schulungen zum Advanced Level Test Management abgedeckt wird. Der Schwerpunkt liegt auf Konzepten und Verfahren des Testens, die auf alle Softwareprojekte angewendet werden können.

0.11 Wie dieser Lehrplan organisiert ist

Es gibt drei Kapitel mit prüfbarem Inhalt. Die oberste Überschrift jedes Kapitels gibt die Mindestzeit an, die für akkreditierte Schulungen erforderlich ist, um den Inhalt des Kapitels abzudecken; unterhalb der Kapitelebene werden keine Zeitangaben gemacht. Für akkreditierte Ausbildungskurse verlangt der Lehrplan mindestens 22,75 Unterrichtsstunden, die sich wie folgt auf die drei Kapitel verteilen:

- Kapitel 1: Die Testaktivitäten managen (750 Minuten)
  - Der Teilnehmer lernt, die Aktivitäten des Testmanagements zu erklären (d. h. Testplanung, Testüberwachung, Teststeuerung und Testabschluss).

- Der Teilnehmer lernt, eine Teststrategie für das Projekt zu definieren, einschließlich der Testziele und der Auswahl des richtigen Testansatzes in Übereinstimmung mit der organisationsweiten Teststrategie und dem Projektkontext.
- Der Teilnehmer lernt, Projekte in verschiedenen Kontexten zu managen.
- Der Teilnehmer lernt, risikobasierte Tests anzuwenden, um das Testen auf die identifizierten Risiken zu konzentrieren.
- Der Teilnehmer lernt, Aktivitäten der Testprozessverbesserung zu leiten, indem er eine Retrospektive für ein Projekt oder eine Iteration durchführt.
- Der Teilnehmer lernt, wie er die Werkzeugunterstützung für das Testen verbessern kann, indem er die Risiken, Kosten und Vorteile der Werkzeugunterstützung berücksichtigt.

- Kapitel 2: Das Produkt managen (390 Minuten)
  - Der Teilnehmer lernt, das Testen mit Hilfe von Testmetriken zu überwachen und zu steuern, um die Testziele zu erreichen sowie den Testfortschritt zu bewerten und zu berichten.
  - Der Teilnehmer lernt die Auswahl geeigneter Testschätzungsverfahren bei verschiedenen Organisationen nach unterschiedlichen Softwareentwicklungsmodellen.
  - Der Teilnehmer lernt, wie man einen Fehlerworkflow innerhalb des Fehlermanagements für sequenzielle, agile und hybride Softwareentwicklungs-Modelle definiert.
- Kapitel 3: Das Team managen (225 Minuten)
  - Der Teilnehmer lernt, wie man einen gegebenen Projektkontext analysiert, um die vom Testteam benötigten Fähigkeiten zu identifizieren.
  - Der Teilnehmer lernt, ein Team nach dem Whole-Team-Ansatz zu führen.
  - Der Teilnehmer lernt, wie man einen Business Case für die Testaktivitäten im Projekt definiert.

Hinweis: Für jedes Lernziel existiert im aktuellen Lehrplan ein entsprechender Unterabschnitt mit Inhalt (z. B. für LO-1.2.3 existiert ein Unterabschnitt 1.2.3).

0.12 Was sind die grundlegenden Annahmen dieses Lehrplans?

Dieser Lehrplan richtet sich an alle, die ein fortgeschrittenes Kompetenzniveau im Testmanagement erreichen wollen, wie Testmanager, Testanalytiker, Testingenieure, Testberater, Testkoordinatoren, Testleiter und Projektmanager. Er ist abgestimmt auf den ins Deutsche übersetzten ISTQB Lehrplan Certified Tester Foundation Level V4.0, der die grundlegenden Kenntnisse und das Verständnis des Softwaretestens vermittelt.

In dem vorliegenden Lehrplan werden zwei Hauptrollen beim Testen behandelt: die Rolle des Testmanagements und die Rolle des Testers.

Die Rolle des Testmanagements wird im Kontext des sequenziellen Entwicklungsmodells auch als Testmanager bezeichnet, wobei der Testmanager in der Regel eine vom Projektmanager oder dem Product Owner getrennte Rolle ist. Die Testmanagement-Rolle ist für den gesamten Testprozess, das Testteam und das Testmanagement verantwortlich. Dazu

gehören die Festlegung der Teststrategie, die Planung der Testaktivitäten, die Überwachung und Steuerung des Testfortschritts, die Berichterstattung über die Testergebnisse und das Management der Testrisiken und -probleme. Das Testmanagement stellt außerdem sicher, dass die Testziele mit den Anforderungen des Unternehmens und der Stakeholder übereinstimmen und dass die Testaktivitäten mit anderen Projektbeteiligten koordiniert werden.

Die Rolle des Testers führt auch Aktivitäten zur Testauswertung, zum Fehlermanagement und zum Abschluss der Testaktivitäten durch. Die Tester setzen verschiedene Testverfahren ein, um die Qualität und Zuverlässigkeit der Testartefakte und des Systems unter Test sicherzustellen. Die Rolle des Testers verwendet auch Testwerkzeuge und -automatisierung, um den Testprozess zu unterstützen und die Effizienz und Effektivität der Tests zu verbessern. Die Aktivitäten und Aufgaben, die diesen Rollen zugewiesen werden, können je nach Kontext, wie Projekt, Produkt, Fähigkeiten und Organisation, variieren (siehe auch ISTQB®-Lehrplan Foundation Level V4.0).

Der Begriff *Testteammitglied* wird in diesem Lehrplan für jede Person in einer Testmanagement- oder Testrolle verwendet, die Tests durchführt, unabhängig vom organisatorischen Kontext und anderen Rollen. Testteams setzen sich aus Personen mit unterschiedlichen Fähigkeiten und Kompetenzen zusammen. Die Teammitglieder können auch über unterschiedliche Erfahrungsstufen und Zertifizierungen verfügen, z. B. Grundkenntnisse, fortgeschrittene Kenntnisse oder Expertenkenntnisse. Die Mitglieder von Testteams können je nach Testansatz und Testprozessmodell, wie z. B. agiles Testen, modellbasiertes Testen, risikobasiertes Testen, auch unterschiedliche Rollen und Verantwortlichkeiten haben.

Ein wichtiger Punkt in Bezug auf die Perspektive, die der Lehrplan bietet, ist die Tatsache, dass er sich auf das Testmanagement auf Projektebene und nicht auf das Testmanagement auf Organisationsebene konzentriert. Daher entspricht dieser Lehrplan den Anforderungen und enthält Informationen, die für das Testmanagement auf Projektebene verwendet werden können, aber weniger für das Testmanagement auf Organisationsebene.

Mit hybrider Softwareentwicklung wird in diesem Lehrplan ein Softwareentwicklungsansatz bezeichnet, der Elemente verschiedener Softwarelebenszyklusmodelle wie des V-Modells, des iterativen, des inkrementellen und des agilen Modells kombiniert. Die hybride Softwareentwicklung zielt darauf ab, die Stärken der einzelnen Modelle zu nutzen und ihre Schwächen abzumildern, je nach Kontext und Anforderungen des Projekts. So kann ein hybrides Softwareentwicklungsmodell beispielsweise ein V-Modell für die anfängliche Planungs- und Anforderungsanalysephase verwenden, gefolgt von einem agilen Modell für die Design-, Entwicklungs- und Testphase. Alternativ dazu kann ein hybrides Softwareentwicklungsmodell ein iteratives Modell für das gesamte Projektmanagement nutzen, während für jede Iteration ein inkrementelles Modell und für jedes Inkrement ein agiles Modell angewendet wird. Die hybride Softwareentwicklung erfordert ein hohes Maß an Flexibilität, Kommunikation und Zusammenarbeit zwischen den Beteiligten sowie ein klares Verständnis der Ziele, Risiken und Einschränkungen der einzelnen Phasen und Modelle.

Laut diesem Lehrplan und dem Glossar ist eine Teststrategie eine Beschreibung, wie das Testen durchgeführt wird, um die Testziele unter den gegebenen Umständen zu erreichen. Eine Teststrategie legt den Geltungsbereich, den Testansatz und die Ressourcen für das Testen eines Systems oder eines Produkts fest. Sie wird in der Regel in einem Testkonzept oder als Teil anderer Dokumente dokumentiert, je nach dem Kontext des Testens. Eine Teststrategie wird von der organisationsweiten Teststrategie –einer übergeordneten Teststrategie – beeinflusst, die beschreibt, wie das Testen in einer Organisation durchgeführt wird. Eine Teststrategie kann auch für eine einzelne Teststufe oder eine Testart existieren.

Dabei handelt es sich um spezifische Aspekte des Testens, die sich auf unterschiedliche Ziele, Vorgaben und Kriterien konzentrieren. Der allgemeine Begriff "Teststrategie" kann in jedem Kontext (Projekt, Organisation, Produkt) verwendet werden. Ein Testansatz ist die Art und Weise, in der Testaufgaben umgesetzt werden, insbesondere die Auswahl und Kombination von Teststufen, Testarten und Testverfahren für statische und dynamische Tests sowie andere Testverfahren wie Testskripte, manuelle Tests, vergleichende Testverfahren usw. Der vom Testmanagement gewählte Testansatz ist eine wichtige Entscheidung bei der Formulierung einer geeigneten Teststrategie für einen bestimmten Kontext.

1 Die Testaktivitäten managen – 750 Minuten

Schlüsselbegriffe

Eintrittswahrscheinlichkeit des Risikos, erfahrungsbasiertes Testen, funktionaler Test, hybrides Softwareentwicklungsmodell, inkrementelles Entwicklungsmodell, iteratives Entwicklungsmodell, nicht-funktionaler Test, Produktrisiko, Qualitätsrisiko, Retrospektive, Risikoanalyse, risikobasiertes Testen, Risikobewertung, Risikoidentifizierung, Risikomanagement, Risikominderung, Risikostufe, Risikoüberwachung, S.M.A.R.T.-Zieldefinitionsmethode, Schadensausmaß des Risikos, sequenzielles Entwicklungsmodell, Softwareentwicklungslebenszyklus, Test Maturity Model Integration, Testabschluss, Testart, Testkonzept, Testplanung, Testprozessverbesserung, Teststeuerung, Teststrategie, Teststufe, Testüberwachung, Testziel, TPI NEXT

Domänenspezifische Schlüsselbegriffe

IDEAL, Indikator, Messung, Metrik, Ziel-Frage-Metrik (Goal Question Metric, GQM)

Lernziele für Kapitel 1: Der Lernende kann …

1.1 Der Testprozess

TM-1.1.1 (K2) … die Aktivitäten der Testplanung zusammenfassen.

TM-1.1.2 (K2) … die Aktivitäten der Testüberwachung und Teststeuerung zusammenfassen.

TM-1.1.3 (K2) … die Aktivitäten des Testabschlusses zusammenfassen.

1.2 Der Kontext des Testens

TM-1.2.1 (K2) … vergleichen, warum verschiedene Stakeholder am Testen interessiert sind.

TM-1.2.2 (K2) … erklären, warum das Wissen der Stakeholder im Testmanagement relevant ist.

TM-1.2.3 (K2) … das Testen in einem hybriden Softwareentwicklungsmodell erklären.

TM-1.2.4 (K2) … die Aktivitäten des Testmanagements für verschiedene Softwareentwicklungslebenszyklen zusammenfassen.

TM-1.2.5 (K2) … die Aktivitäten des Testmanagements auf verschiedenen Teststufen miteinander vergleichen.

TM-1.2.6 (K2) … die Aktivitäten des Testmanagements für verschiedene Testarten miteinander vergleichen.

TM-1.2.7 (K4) … ein vorgegebenes Projekt analysieren und dazu Testmanagementaktivitäten, insbesondere Testplanung, Testüberwachung und Teststeuerung, festlegen.

1.3 Risikobasiertes Testen

TM-1.3.1 (K2) … die verschiedenen Maßnahmen erläutern, die bei risikobasiertem Testen ergriffen werden müssen, um auf Risiken zu reagieren.

TM-1.3.2 (K2) … Beispiele für verschiedene Verfahren nennen, die ein Testmanager zur Identifizierung von Risiken im Zusammenhang mit der Produktqualität verwenden kann.

TM-1.3.3 (K2) … die Faktoren zusammenfassen, die die Risikostufen in Bezug auf die Produktqualität beeinflussen.

TM-1.3.4 (K4) … geeignete Testaktivitäten zur Minderung von Risiken gemäß ihrer Risikostufe in einem bestimmten Kontext auswählen.

TM-1.3.5 (K2) … Beispiele von schwergewichtigen und leichtgewichtigen risikobasierten Testverfahren unterscheiden.

TM-1.3.6 (K2) … Beispiele für Erfolgsmetriken und Schwierigkeiten im Zusammenhang mit risikobasierten Tests nennen.

1.4 Die Teststrategie für das Projekt

TM-1.4.1 (K2) … die typische Auswahl an Testansätzen erläutern.

TM-1.4.2 (K4) … eine organisationsweite Teststrategie und den Projektkontext analysieren, um einen geeigneten Testansatz auszuwählen.

TM-1.4.3 (K3) … die S.M.A.R.T.-Zieldefinitionsmethode verwenden, um messbare Testziele und Endekriterien zu definieren.

1.5 Verbesserung des Testprozesses

TM-1.5.1 (K2) … erklären, wie das IDEAL-Modell zur Testprozessverbesserung bei einem bestimmten Projekt einzusetzen ist.

TM-1.5.2 (K2) … den modellbasierten Ansatz zur Verbesserung des Testprozesses zusammenfassen und verstehen, wie man ihn im Projektkontext anwendet.

TM-1.5.3 (K2) … den analytisch basierten Ansatz zur Verbesserung des Testprozesses zusammenfassen und verstehen, wie man ihn im Projektkontext anwendet.

TM-1.5.4 (K3) … eine Projekt- oder Iterationsretrospektive implementieren, um Testprozesse zu bewerten und verbesserungswürdige Testbereiche zu ermitteln.

1.6 Testwerkzeuge

TM-1.6.1 (K2) … die Best Practices für die Einführung von Werkzeugen zusammenfassen.

TM-1.6.2 (K2) … die Auswirkungen der verschiedenen technischen und fachlichen Aspekte bei der Entscheidung für eine Werkzeugart erklären.

TM-1.6.3 (K4) … eine gegebene Situation analysieren und damit einen Plan für die Auswahl von Werkzeugen erstellen, der Risiken, Kosten und Nutzen berücksichtigt.

TM-1.6.4 (K2) … die Phasen des Lebenszyklus eines Werkzeugs voneinander unterscheiden.

TM-1.6.5 (K2) … Beispiele für die Erfassung und Bewertung von Metriken mithilfe von Testwerkzeugen nennen.

1.1 Der Testprozess

Einführung

Der ISTQB®-Lehrplan Certified Tester Foundation Level V4.0 beschreibt einen Testprozess, der die folgenden Aktivitäten umfasst: Testplanung, Testüberwachung und Teststeuerung, Testanalyse, Testentwurf, Testrealisierung, Testdurchführung und Testabschluss.

Im ISTQB®-Lehrplan Foundation Level V4.0 heißt es, dass diese Aktivitäten im Testprozess oft iterativ oder parallel durchgeführt werden, je nach Softwareentwicklungslebenszyklus (Software Development Lifecycle, SDLC) und Projektkontext. In der Regel ist es notwendig, diese Aktivitäten auf das Produkt und das Projekt abzustimmen.

In diesem Lehrplan liegt der Schwerpunkt auf den folgenden Aktivitäten des Testmanagements:

- **Testplanung:** Definition der Testziele, des Testansatzes, des Testumfangs, der Testressourcen, des Testzeitplans, der Testliefergegenstände und der Testteilnehmer (Test-Stakeholder)
- **Testüberwachung und Teststeuerung:** Verfolgung des Testfortschritts und der Testergebnisse; Einleitung von Korrekturmaßnahmen, wenn nötig; Berichten des Teststatus und der Ergebnisse an die relevanten Stakeholder
- **Testabschluss:** Fertigstellung und Archivierung der Testartefakte, Bewertung des Testprozesses und des Testarbeitsergebnisses, Identifikation von Maßnahmen zur Testprozessverbesserung und Kommunikation des Testabschlusses an die relevanten Stakeholder

Die Norm ISO/IEC/IEEE 29119-2 definiert die Testmanagementprozesse, die diese Aktivitäten des Testmanagements abdecken. Diese Testmanagementprozesse können auf verschiedenen Ebenen des Testens angewandt werden, z. B. auf Projekt-, Programm- oder Portfolioebene. Jede Ebene kann über ein eigenes Testkonzept verfügen, das mit dem Testkonzept der höheren Ebene abgestimmt ist.

1.1.1 Testplanungsaktivitäten

Dieser Abschnitt konzentriert sich auf die Aktivitäten zur Planung von Tests für verschiedene Umfänge, wie das gesamte Projekt, eine Teststufe, eine Testart oder ein Release/eine Iteration/einen Sprint im Agilen. Je nach Umfang kann die Testplanung an verschiedenen Punkten im Entwicklungsprozess beginnen und enden. Bei der Testplanung geht es darum, die Aktivitäten und Ressourcen zu ermitteln, die erforderlich sind, um die in der Testrichtlinie festgelegten Testziele zu erreichen. Mit der Testplanung sollte so früh wie möglich im Entwicklungsprozess begonnen werden, vorzugsweise bevor die Anforderungen ermittelt werden, und sie sollte mit dem Projektfortschritt aktualisiert werden. Die Testplanung ist häufig ein iterativer Prozess, der während des Projekts eine Neuplanung erfordert, um Änderungen und Rückmeldungen zu berücksichtigen.

Die folgenden Aufgaben sind Teil der Testplanung (ähnlich denen in der ISO/IEC/IEEE 29119-2):

- **Verständnis des Kontexts und Organisation der Testplanung:** Für die Testplanung ist es entscheidend, den Kontext der Organisation (z. B. die Testrichtlinie und die organisationsweite Teststrategie), den Testumfang und das Testelement (d. h. das zu testende Arbeitsergebnis) zu verstehen (siehe Abschnitt 1.2 „Der Kontext des Testens“). Dazu gehören auch alle Aktivitäten, die für die Gestaltung der Testkonzeptentwicklung

und der Genehmigung dieser Aktivitäten und des Zeitplans durch die Stakeholder (z. B. den Product Owner, den Projektmanager oder den Leiter des Entwicklungsteams) benötigt werden.

- **Identifizierung und Analyse von Produktrisiken:** Die Risikoanalyse umfasst die Identifizierung und Bewertung der potenziellen Schadensausmaße und der Eintrittswahrscheinlichkeit von Produktrisiken im Rahmen der Testplanung. In Abschnitt 1.3 „Risikobasiertes Testen“ finden Sie weitere Einzelheiten zu Produktrisiken.
- **Identifizierung von Ansätzen zur Risikobehandlung:** Auf der Grundlage der Risikoanalyse werden die geeigneten Ansätze zur Risikobehandlung ausgewählt und im Testkonzept dokumentiert. Diese können vorbeugende, korrigierende oder minimierende Maßnahmen zur Bewältigung der identifizierten Risiken umfassen.
- **Definition des Testansatzes, Schätzung und Zuweisung der Testressourcen:** Auf der Grundlage der organisationsweiten Teststrategie, der regulatorischen Standards, der durch das Projekt gegebenen Einschränkungen und der Ansätze zur Risikobehandlung wird der Testansatz für den aktuellen Testumfang definiert (siehe Abschnitt 1.4 „Die Teststrategie für das Projekt“). Bei der Festlegung eines Testansatzes ist es wichtig, die erforderlichen Testressourcen wie Testpersonal, Testwerkzeuge, Testumgebungen und Testdaten abzuschätzen und diese Ressourcen den Testaktivitäten zuzuweisen.
- **Erstellung des Testkonzepts**: Das Testkonzept muss von allen Stakeholdern akzeptiert werden, und daher sollten Differenzen zwischen ihnen frühzeitig ausgeräumt werden.

1.1.2 Testüberwachungs- und Teststeuerungsaktivitäten

Damit das Testmanagement eine effiziente Teststeuerung gewährleisten kann, muss ein Testzeitplan und ein Testüberwachungskonzept geschaffen werden, die es ermöglichen, den Status des Testens und den Testfortschritt zu verfolgen. Dieser Rahmen sollte die detaillierten Messgrößen und Ziele enthalten, die erforderlich sind, um den Status der Testergebnisse und Ressourcen mit dem Plan und den strategischen Zielen in Beziehung zu setzen.

Bei kleinen und weniger komplexen Projekten kann es relativ einfach sein, die Testergebnisse und -aktivitäten mit dem Plan und den strategischen Zielen in Verbindung zu bringen, aber im Allgemeinen sollten dazu detailliertere Ziele definiert werden. Dies kann die Definition von Maßnahmen und Zielvorgaben umfassen, die erforderlich sind, um die Testziele und die Überdeckung der Testbasis zu erreichen.

Besonders wichtig ist es, den Status von Testarbeitsergebnissen und -aktivitäten in einer Weise darzustellen, die für die Projekt- und Business-Stakeholder verständlich und relevant ist.

Testüberwachung und Teststeuerung sind fortlaufende Aktivitäten.

Die Teststeuerung vergleicht den tatsächlichen Testfortschritt mit den im Testkonzept festgelegten Endekriterien und führt bei Bedarf Korrekturmaßnahmen durch. Sie leitet das Testen, um die Teststrategien zu erfüllen und die Testziele zu erreichen (siehe Abschnitt 1.4 „Die Teststrategie für das Projekt“) und überdenkt die Testplanungsaktivitäten bei Bedarf. Um angemessen reagieren zu können, ist es notwendig, dass die Teststeuerung auf detaillierte Informationen der Testplanung zurückgreifen kann. Diese Aktivität umfasst:

- Umsetzung des Testkonzepts und der Teststeuerungsrichtlinien
- Management von Abweichungen von der Testplanung (Testzeitplan)
- Behandlung von neu identifizierten und veränderten Risiken

- Feststellung der Bereitschaft, um mit dem Testen zu beginnen
- Erteilung und Einholung der Genehmigung für den Testabschluss auf der Grundlage der Endekriterien.

Die Testüberwachung umfasst das Sammeln und Aufzeichnen von Testergebnissen, das Erkennen von Abweichungen vom geplanten Testen (Testzeitplan), das Erkennen und Analysieren neuer Risiken, die ein Testen erfordern, und das Überwachen von Änderungen bei identifizierten Risiken.

1.1.3 Testabschlussaktivitäten

Der Testabschluss erfolgt in der Regel an Projektmeilensteinen (z. B. bei einem Release, am Ende einer Iteration oder beim Abschluss der Teststufen). Für alle nicht behobenen Fehlerzustände werden Änderungsanträge (Change Requests) oder Product-Backlog-Einträge erstellt (siehe hierzu ISTQB®-Lehrplan Foundation Level V4.0). Sobald die Endekriterien erfüllt sind, sollten die wichtigsten Ergebnisse erfasst, archiviert und den entsprechenden Stakeholdern zur Verfügung gestellt werden. Der Testabschluss erfordert die folgenden Aufgaben:

- **Erstellung und Genehmigung des Testabschlussberichts:** Diese Aufgabe stellt sicher, dass alle Tests ausgeführt und alle Testziele erreicht wurden. Sie umfasst das Sammeln relevanter Informationen aus verschiedenen Testmitteln, wie Testkonzepten, Testergebnissen, Testfortschrittsberichten, Testabschlussberichten und Fehlerberichten. Die gesammelten Informationen werden ausgewertet und im Testabschlussbericht zusammengefasst. Der Testabschlussbericht wird genehmigt und an die relevanten Stakeholder weitergeleitet.
- **Archivierung der Testmittel:** Diese Aufgabe identifiziert Testmittel, die in der Zukunft nützlich sein können oder voraussichtlich wiederverwendet werden, in der Regel die Testfälle. Die Archivierung macht diese Testmittel für die zukünftige Wiederverwendung zugänglich und leicht verständlich. Darüber hinaus sollten Testergebnisse, Testprotokolle, Testberichte und andere Testmittel vorübergehend im Konfigurationsmanagementsystem archiviert werden.
- **Übergabe von Testmitteln:** Diese Aufgabe liefert wertvolle Arbeitsergebnisse an diejenigen, die sie benötigen. So sollten beispielsweise bekannte Fehlerzustände, die zurückgestellt oder akzeptiert wurden, denjenigen mitgeteilt werden, die die Testmittel verwenden oder deren Verwendung unterstützen werden.
- **Bereinigen und Rücksetzen der Testumgebung:** Diese Aufgabe stellt sicher, dass die Testumgebung für den nächsten Testzyklus oder das nächste Projekt bereit ist. Sie umfasst das Entfernen aller Testdaten, Testwerkzeuge, Testtreiber, Platzhalter und Testskripte aus der Testumgebung. Außerdem muss die Testumgebung in ihren ursprünglichen oder gewünschten Zustand zurückgesetzt werden.
- **Durchführung/Sammlung/Dokumentation von Lessons Learned:** Diese Aufgabe wird in Retrospektiven durchgeführt, in denen wichtige Erkenntnisse aus dem Testprozess diskutiert und dokumentiert werden. Dies kann auch Befunde für den gesamten Softwareentwicklungslebenszyklus (Software Development Life Cycle, SDLC) beinhalten. Die gewonnenen Erkenntnisse können für die Testprozessverbesserung genutzt werden, wie in Abschnitt 1.5 „Verbesserung des Testprozesses“ beschrieben sind.

1.2 Der Kontext des Testens

Einführung

Der Kontext des Testens umfasst die besonderen Bedingungen und Einschränkungen, die den Testprozess beeinflussen und die Entscheidungen und Strategien für die Planung, den Entwurf und die Ausführung von Tests prägen. Für Testmanager ist es von entscheidender Bedeutung, diesen Kontext zu verstehen, um das Testen auf die spezifischen Bedürfnisse und Ziele des Softwareentwicklungsprojekts abzustimmen. Dieser Kontext kann sich je nach Produkt, Branche, gesetzlichen Anforderungen und vom spezifischen Softwareentwicklungslebenszyklus (SDLC) unterscheiden.

Testmanager haben eher die Aufgabe, etablierte Teststrategien anzuwenden und Testverfahren auszuwählen, und weniger, solche zu entwickeln. Sie spielen eine Schlüsselrolle bei der Formulierung von Testkonzepten, die auf den Kontext des Projekts zugeschnitten sind. Indem sie diese verschiedenen Faktoren verstehen und berücksichtigen, können Testmanager sicherstellen, dass die Tests zweckdienlich, effektiv und effizient sind, um die Testziele zu erreichen.

1.2.1 Test-Stakeholder

Test-Stakeholder sind Einzelpersonen oder Gruppen, die ein direktes oder indirektes Interesse an der Qualität des Produkts haben. Die nachfolgende Liste fasst potenzielle Stakeholder inklusive ihrer unterschiedlichen Interessen am Testen zusammen:

- **Entwickler, Entwicklungsleiter und Entwicklungsmanager:** Neben der Implementierung des Systems unter Test und dem Handeln aufgrund der der Testergebnisse sind diese Stakeholder auch an den Unit-Tests beteiligt und tragen zum Testprozess bei.
- **Tester, Testleiter und Testmanager:** Diese Stakeholder bereiten Testmittel vor. Dazu gehören die Entwicklung von Testkonzept und die Mitwirkung am Testprozess durch Aktivitäten wie Anforderungsanalyse, Testentwurf, Testdurchführung, Fehlerverfolgung und-berichterstattung, Testautomatisierung und Berichterstattung bezüglich des Testfortschritts.
- **Projektmanager, Produktverantwortliche und Fachbereichsmitarbeiter:** Sie spezifizieren die Anforderungen, definieren die gewünschte Qualität und empfehlen die erforderliche Überdeckung auf der Grundlage der wahrgenommenen Risiken. Sie reviewen auch Arbeitsprodukte, nehmen am Benutzerabnahmetest (User Acceptance Testing, UAT) teil und treffen Entscheidungen auf der Grundlage von Testergebnissen.
- **Betriebsteam:** Im Rahmen der betrieblichen Abnahmetests stellt es die Produktionsreife des Systems sicher und trägt zur Definition nicht-funktionaler Anforderungen bei.
- **Kunden und Benutzer:** Die Kunden kaufen das Produkt, während die Benutzer es direkt nutzen. Beide spielen eine wichtige Rolle bei der Definition der Anforderungen und sollten in die Benutzerabnahmetests (User Acceptance Testing, UAT) einbezogen werden, um zu validieren, ob das Produkt ihre Bedürfnisse erfüllt.

Diese Liste enthält nicht alle potenziellen Stakeholder. Testmanager müssen im Rahmen der Erstellung der Teststrategie und des Testkonzepts eine Stakeholder-Analyse durchführen, die unter Berücksichtigung des Testumfangs die spezifischen Stakeholder für ihr Projekt identifiziert.

1.2.2 Die Bedeutung des Wissens der Stakeholder für das Testmanagement

Beim Testmanagement ist es von entscheidender Bedeutung, die Perspektiven und den Einfluss der verschiedenen Stakeholder zu berücksichtigen. Die Stakeholder-Matrix, die oft auch als Einfluss-Interessen-Matrix bezeichnet wird, hilft Testmanagern dabei, Prioritäten bei der Einbindung von Stakeholdern zu setzen und Erwartungen effizient zu verwalten. Die Stakeholder-Matrix ist ein strategisches Werkzeug für das Testmanagement:

- Sie nutzt das Fachwissen der Stakeholder, wobei Endbenutzer und technische Teams Feedback und Einblicke in Performanz und IT-Sicherheit geben.
- Sie unterstützt das Risikomanagement, indem sie die Interessen und den Einfluss der Stakeholder hervorhebt und so proaktive Bemühungen zur Risikominderung fördert.
- Sie fördert die Berücksichtigung unterschiedlicher Perspektiven und liefert wertvolles Feedback.

Die Stakeholder-Matrix besteht aus vier Quadranten:

- **Förderer (hoher Einfluss, hohes Interesse):** Sie sind wichtige Schlüsselpersonen mit hohem Einfluss und Interesse, die wesentlich sind für die Gestaltung der Teststrategie und des Testkonzepts.
- **Latente Personen (hoher Einfluss, geringes Interesse):** Sie haben zwar kein großes Interesse an den täglichen Aufgaben, aber ihre Entscheidungen sind maßgeblich für die Ressourcenzuweisung und die Ausrichtung des Projekts auf höchster Ebene.
- **Verteidiger (geringer Einfluss, hohes Interesse):** Sie geben oft qualitatives Feedback und können durch regelmäßige Updates und Beteiligung an bestimmten Diskussionen eingebunden werden.
- **Apathische Personen (geringer Einfluss, geringes Interesse):** Sie sind zwar nicht eng eingebunden, aber wenn sie über wichtige Meilensteine auf dem Laufenden gehalten werden und ihre Meinung zu bestimmten Themen eingeholt wird, können dadurch einzigartige Erkenntnisse gewonnen werden.

Zu den Aufgaben des Testmanagers gehört es, eine detaillierte Liste der Stakeholder zu erstellen und die Beziehung der einzelnen Stakeholder zu den Testaktivitäten zu verstehen, um die Effektivität des Testmanagements zu verbessern.

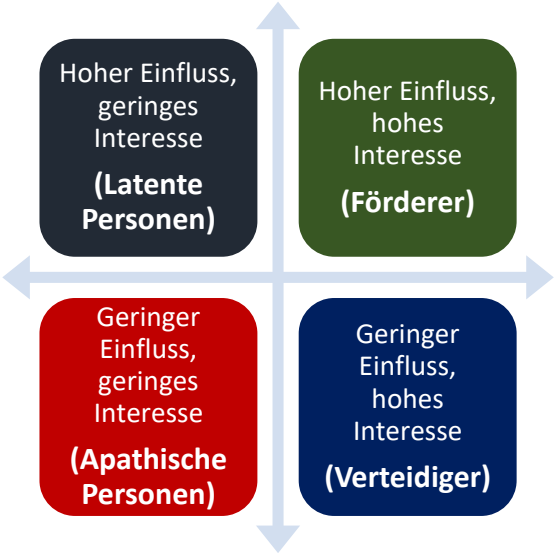

Abbildung 1: Verschiedene Arten von Stakeholdern

1.2.3 Testmanagement in einem hybriden Softwareentwicklungsmodell

Hybride Softwareentwicklungsmodelle integrieren Elemente sowohl traditioneller sequenzieller Ansätze als auch agiler Praktiken, um spezifischen Projektanforderungen oder organisatorischen Wandel gerecht zu werden. Die folgenden Gründe sind häufige Gründe für die Verwendung eines hybriden Softwareentwicklungsmodells, obwohl es je nach Organisation und Projekt auch andere Gründe geben kann:

- **Hybrid als Übergang zu Agile:** Der Übergang von traditionellem zu agilem Ansatz kann aufgrund der grundlegenden Veränderungen im Arbeitsablauf, Kultur und Teamdynamik eine Herausforderung sein. Hybride Modelle erleichtern diesen Übergang durch einen ausgewogenen Ansatz, in dem die die Struktur traditioneller Ansätze mit der Flexibilität agiler Praktiken kombiniert wird.
- **Hybride Lösungen:** Einigen Organisationen oder Projekten ist es nicht möglich, auf agiles Vorgehen umzustellen. Projekte mit hohem Risiko erfordern möglicherweise für einige Artefakte sequenzielle Vorgehensweisen und für andere agile Praktiken. Diese Projekte können ein hybrides Modell verwenden, wenn es für den Zweck geeignet ist.

In einer hybriden Umgebung kann das Testmanagement folgende Aktivitäten umfassen:

- Bewertung des Verständnisses und der Fähigkeit des Teams, nahtlos zwischen traditionellem und agilem Vorgehen zu wechseln.
- Identifizierung von Stärken und Schwächen bei der Anpassung an einen hybriden Ansatz.
- Sicherstellen, dass das Team in der Lage ist, strukturierte Prozesse mit agiler Flexibilität zu kombinieren.
- Verbesserung der Zusammenarbeit zwischen dem Testteam und den Stakeholdern, um das Testen innerhalb von Sprints und traditionellen Testen besser zu verwalten.
- Teilnahme an koordinierten Maßnahmen, wie Scrum-of-Scrums für Tester, um den Fokus auf das Testen beizubehalten und gleichzeitig zu den allgemeinen Entwicklungszielen beizutragen.
- Verfolgung und Review von Tests und Testfällen innerhalb von Sprints, um sicherzustellen, dass sie mit den agilen Praktiken übereinstimmen.

Weitere Informationen finden Sie in (Fowler, 2010).

1.2.4 Testmanagementaktivitäten für verschiedene Softwareentwicklungslebenszyklus-Modelle

Um das Testen innerhalb des SDLC-Modells richtig auszurichten, muss ein Testmanager die verschiedenen SDLC-Modelle, die in seinem Unternehmen verwendet werden, verstehen und dieses Wissen nutzen, um das Testen richtig auf die Entwicklungsaktivitäten abzustimmen.

Die folgende Tabelle zeigt einen Vergleich zwischen verschiedenen Testmanagementaktivitäten auf der Grundlage unterschiedlicher SDLC-Modelle:

Aspekt	Sequenzielles Entwicklungsmodell z. B. V-Modell	Iteratives Entwicklungsmodell z. B. Scrum
Schätzung	Frühzeitige detaillierte Schätzung für jede Teststufe	Iterative Schätzung, Teil der Story-Planung pro Iteration
Testmittel	Enthält Strategie, Testkonzept, Testfälle, Zeitplan und Berichte.	Konzentriert sich auf Akzeptanzkriterien und die “Definition-of-Done"; minimale Dokumentation.
Rollen	Der Testmanager überwacht die Entscheidungen und das Management des Teams.	Die Rollen sind integriert; ein Moderator oder Coach ersetzt den traditionellen Testmanager.
Werkzeuge	Vorwiegend Testmanagementwerkzeuge, die für traditionelles Testen geeignet sind.	Werkzeuge für CI/CD und Automatisierung sind von zentraler Bedeutung und unterstützen kontinuierliches Testen.
Testansatz	Im Voraus geplant, entsprechend den Projektphasen.	Eingebettet in Iterationen, mit dem Schwerpunkt auf Anpassbarkeit und Feedback.
Testautomatisierung	Strategisch umgesetzt, kann in verschiedenen Phasen erfolgen.	Von Anfang an integriert, mit Schwerpunkt auf automatisierter Regression in CI/CD.
Überwachung und Berichterstattung	Meilensteinbasierte Berichterstattung mit optionalen automatisierten Dashboards	Kontinuierliche Berichterstattung mit Dashboards in Echtzeit und täglichen Status-Updates
Metriken	Konzentriert sich auf traditionelle Metriken für Tests und Fehlerzustände (z. B. Testdurchführung, Fehlerzustände).	Enthält zusätzlich zu den traditionellen Metriken auch agile Metriken für die Iterationsverfolgung (z. B. Team Velocity (Team-Produktivität), Burndown-Charts).

Tabelle 1: Testmanagementaktivitäten für verschiedene SDLC-Modelle

1.2.5 Testmanagementaktivitäten auf verschiedenen Teststufen

Komponententest:

- Definition des Umfangs, der Ziele und der Endekriterien für Komponententests (Unit-Tests).
- Einbeziehung der Tester in Aktivitäten, die über die traditionellen Testrollen hinausgehen, wie z. B. Code-Reviews, bei denen ihre analytischen Fähigkeiten einen Mehrwert darstellen.
- Koordination mit dem Entwicklungsteam bei der Lösung von Problemen und der Unterstützung von Komponententests.

Komponentenintegrationstest:

- Bestimmung von Integrationsabläufen und Testzyklen in Zusammenarbeit mit dem Entwicklungsteam unter Berücksichtigung des SDLC-Modells, der Testwerkzeuge und der Prozesse.
- Überwachung des Fortschritts, um sicherzustellen, dass er mit den Teststrategien für System- und Abnahmetests abgestimmt ist.
- Management dieser Stufe in Zusammenarbeit mit den Entwicklern unter Berücksichtigung der Komponentenintegrationstests.

Systemintegrationstest:

- Sicherstellung, dass der Umfang und die Ziele der Systemintegrationstests klar und auf die Risikobewertung und die Qualitätsziele abgestimmt sind.
- Behalten des Überblicks über den Fortschritt, die Ergebnisse und das Fehlermanagement während der Systemintegrationstests.

Systemtest:

- Anpassung der Planung an das SDLC-Modell mit sorgfältiger Zuweisung von Ressourcen, Auswahl von Werkzeugen und Zeitplanung
  - Bei agilen Projekten Integration von Systemtests mit iterativen Story-Tests, Vermeidung getrennter Testzyklen und Sicherstellung, dass das Testen kontinuierlich und integriert erfolgt. Bei sequenziellen Modellen kann das Testen in geplanten Phasen erfolgen.

Abnahmetests:

- Zusammenarbeit mit Stakeholdern, um die Erfüllung der Akzeptanzkriterien zu überprüfen, zu bestätigen und Testaktivitäten zu planen, einschließlich des Managements von Benutzertests im Benutzerakzeptanztests (User acceptance testing, UAT).
- Koordination der Abnahmetests und Vorbereitung von Tests beim Kunden, um sicherzustellen, dass das Produkt die geschäftlichen Anforderungen und Qualitätsstandards außerhalb der Entwicklungsumgebung erfüllt.
- Unterstützung bei der Behebung von Problemen während der Ausführung von Benutzerakzeptanztests sowie Führung der Stakeholder durch den Prozess der Produktfreigabe, wenn die Akzeptanzkriterien erfüllt sind.

1.2.6 Testmanagementaktivitäten für verschiedene Testarten

Ein effektives Testmanagement erfordert einen integrierten Testansatz, der die besonderen Anforderungen von funktionalen, nicht-funktionalen, Black-Box-Tests und White-Box-Tests berücksichtigt. Bei Managern, die funktionale Tests verantworten, liegt der Schwerpunkt darauf, sicherzustellen, dass alle Funktionalitäten gründlich getestet werden und die definierten Anforderungen erfüllt werden. Beim Testmanagement für nicht-funktionale Tests geht es um die Überprüfung von Systemeigenschaften, wie Performanz und IT-Sicherheit. Beim Black-Box-Testmanagement gilt es, sicherzustellen, dass die Tests benutzerorientiert sind und alle möglichen externen Interaktionen abgedeckt werden. Beim White-Box-Testmanagement geht es darum, die Codestruktur des Testobjekts zu verstehen und sicherzustellen, dass die Tests die interne Logik gründlich überdecken.

Management funktionaler Tests:

- Strategische Planung und Fortschrittsverfolgung: Ausarbeitung einer detaillierten Teststrategie, die sich an den funktionalen Anforderungen und den Projektzielen orientiert, sowie die Überwachung des Testfortschritts.
- Ressourcen-Koordination: Effiziente Zuteilung von menschlichen und technischen Ressourcen, um alle funktionalen Aspekte des Systems abzudecken.

Management nicht-funktionaler Tests:

- Festlegung von Performanzanforderungen: Festlegung von Performanz-Benchmarks und Verwaltung der Testaktivitäten, die das System anhand dieser Kriterien bewerten.
- Verifizierung der Konformität: Beaufsichtigung von Tests, die sicherstellen, dass das System nicht-funktionale Qualitätsmerkmale wie IT-Sicherheit, Gebrauchstauglichkeit und Zuverlässigkeit erfüllt.

Management von Black-Box-Tests:

- Analyse der Testüberdeckung: Sicherstellen, dass Black-Box-Tests alle Benutzerszenarien und geschäftlichen Anforderungen abdecken.
- Einarbeitung von Feedback: Management des Prozesses der Einholung von Feedback von Stakeholdern, um Black-Box-Tests zu verfeinern und Fehlerzustände zu beheben.

Management von White-Box-Tests:

- Optimierung der Codeüberdeckung: Überwachung des Einsatzes von Werkzeugen zur Codeüberdeckung, um Abweichungen bei White-Box-Tests zu ermitteln und Ressourcen zur Behebung dieser Abweichungen bereitzustellen.
- Integration technischer Erkenntnisse: Management der Einbeziehung technischer Erkenntnisse in die Testplanung, um sicherzustellen, dass die Tests mit einem Verständnis der internen Struktur der Anwendung entworfen werden.

1.2.7 Testmanagementaktivitäten zur Planung, Überwachung und Steuerung

Ein effektives und effizientes Testmanagement ist der Eckpfeiler jeder erfolgreichen Testarbeit. Es umfasst eine Vielzahl von Aktivitäten, die eine sorgfältige Planung, eine aufmerksame Überwachung und eine strategische Steuerung erfordern. Testmanager spielen eine entscheidende Rolle, wenn es darum geht, sicherzustellen, dass der Testprozess nicht nur effektiv und effizient, sondern auch auf die besonderen Anforderungen des jeweiligen Projekts zugeschnitten ist.

Testplanung:

- **Umfassende Definition des Testumfangs:** Ein Testkonzept muss sorgfältig ausgearbeitet werden und eine gründliche Definition des Testumfangs enthalten. Dazu gehört die Identifizierung aller funktionalen und nicht-funktionalen Anforderungen, um eine vollständige Überdeckung durch Tests zu gewährleisten. Außerdem müssen die Auswirkungen von Black-Box-Tests und White-Box-Tests berücksichtigt werden, um sicherzustellen, dass die entwickelten Testfälle in der Lage sind, das System im Test aus allen Blickwinkeln zu validieren.
- **Risikobewertung und Planung zur Risikominderung:** Integraler Bestandteil des Testkonzepts ist ein robustes Risikomanagement-Rahmenwerk. Testmanager müssen eine detaillierte Risikoanalyse durchführen und potenzielle Schwachstellen und

Herausforderungen aufzeigen, die sich sowohl auf den Projektablauf als auch auf das Endprodukt auswirken könnten. Die Entwicklung von Strategien zur Risikominderung ist von entscheidender Bedeutung. Dazu gehört eine vorausschauende Planung, um diese Risiken effektiv zu umgehen oder zu minimieren.

- **Strategie der Ressourcenzuweisung:** Die Ressourcenplanung ist ein weiteres wichtiges Element. Dabei geht es nicht nur um die bloße Zuweisung von Ressourcen, sondern auch darum, die Struktur des Teams zu definieren, die Rollen abzugrenzen und Kommunikationsprozesse festzulegen. In Umgebungen, in denen die Teams verteilt sind, wie z. B. bei Onsite/Offsite-Modellen, ist dies besonders wichtig, um eine Synchronisierung und möglichst nahtlose Zusammenarbeit zu gewährleisten.

Testüberwachung:

- **Überwachung der Ausführung:** Die Überwachung spielt eine zentrale Rolle im Testmanagementprozess. Sie umfasst eine kontinuierliche Überprüfung der Testdurchführung hinsichtlich der festgelegten Testplanung, die Verfolgung des Fortschritts der Testausführung und das Management aller auftretenden Fehlerzustände. Die Anpassung der Prioritäten für die Tests auf der Grundlage von Risikobewertungen und Echtzeitentwicklungen stellt sicher, dass die Tests auf die kritischsten Bereiche ausgerichtet bleiben.
- **Auswahl und Überwachung von Testwerkzeugen und -umgebungen:** Die umsichtige Auswahl und Verwendung von Testwerkzeugen und -umgebungen sind entscheidend für die Unterstützung der Teststrategie. Die kontinuierliche Überwachung stellt sicher, dass sie effektiv in die CI/CD-Pipeline integriert sind, was kontinuierliche Tests und unmittelbare Feedbackschleifen ermöglicht, die für den agilen Entwicklungsprozess unerlässlich sind.
- **Zusammenarbeit mit der Entwicklung:** Die Pflege einer engen Zusammenarbeit mit dem Entwicklungsteam ist für erfolgreiche Testergebnisse unerlässlich. Diese Zusammenarbeit sollte einen umfassenden Testansatz unterstützen, bei dem Erkenntnisse aus allen testrelevanten Perspektiven genutzt werden, um potenzielle Probleme präventiv anzugehen.

Teststeuerung:

- **Anpassungsfähiges Prozessmanagement:** Bei der Teststeuerung geht es darum, den Testprozess dynamisch als Reaktion auf neue Erkenntnisse, Herausforderungen und die sich entwickelnde Projektdynamik anzupassen. Dies erfordert, dass ein Testmanager schnell reagieren und flexibel in der Lage sein soll, Änderungen am Testansatz vorzunehmen, die dem aktuellen Stand des Projekts entsprechen.
- **Quality Gate Management:** Ein strukturierter Ansatz für das Quality Gate Management ist von grundlegender Bedeutung. Dazu gehören die Definition, was ein Quality Gate innerhalb des Testlebenszyklus darstellt und wie man man fundierte Entscheidungen über den Fortschritt des Testzyklus trifft, was entscheidend für die Sicherstellung der Produktintegrität ist.

Indem sich Testmanager auf diese spezifischen Aktivitäten innerhalb der Testplanung, Testüberwachung und Teststeuerung konzentrieren, können Testmanager sicherstellen, dass der Testprozess gut definiert ist, an Projektänderungen angepasst werden kann und zu einem Produkt führt, das sowohl die Anforderungen des Projekts als auch die Erwartungen der Stakeholder erfüllt.

1.3 Risikobasiertes Testen

Einführung

Der risikobasierte Test beinhaltet die Identifizierung, Bewertung, Überwachung und Minderung bzw. Vermeidung von Risiken zur Steuerung des Testens. Diese Risiken werden von verschiedenen Stakeholdern identifiziert und können für die Auswahl und Priorisierung von Tests verwendet werden. Je höher die Risikostufe, desto früher sollte mit dem Testen begonnen werden und desto intensiver und umfangreicher sollte der Testaufwand sein.

1.3.1 Testen als Aktivität zur Risikominderung

Ein Produktrisiko ist eine potenzielle Situation, in der Qualitätsprobleme in einem Produkt vorhanden sein können. Wenn Tests Fehlerzustände aufdecken, hat das Testen dazu beigetragen, das Produktrisiko zu mindern bzw. zu eliminieren, indem es auf Fehlerzustände aufmerksam macht und die Gelegenheit bietet, diese vor der Freigabe zu beseitigen. Wenn Tests keine Fehlerzustände finden, deutet dies darauf hin, dass das Produktrisiko geringer ist als erwartet.

Die Testmanager sind unter anderem dafür verantwortlich, eine korrekte und zuverlässige Bewertung der Produktqualität zu liefern. Dies erfordert eine aktive Beteiligung am Projektrisikomanagement mit Schwerpunkt auf Projektrisiken im Zusammenhang mit der Qualitätssicherung (z. B. unklare Anforderungen, die zu großen Problemen bei der späten Validierung führen, unzureichende Testumgebungen, die die Testdurchführung behindern).

Der risikobasierte Test konzentriert sich beim Testen auf die Qualitätsrisiken. Er folgt dem generischen Risikomanagementprozess, der aus den folgenden Hauptaktivitäten besteht:

- Risikoanalyse, bestehend aus Risikoidentifizierung und Risikobewertung
- Risikosteuerung, bestehend aus Risikoüberwachung und Risikominderung

Diese Hauptaktivitäten sind logisch nacheinander angeordnet, können sich aber auch überschneiden.

Um das Testen auf Qualitätsrisiken auszurichten, müssen die Qualitätsrisiken identifiziert und bewertet werden. Um möglichst effektiv zu sein, sollte die Risikoanalyse verschiedene Stakeholder einbeziehen. Als einer der Hauptbeteiligten an der Analyse der Qualitätsrisiken sollten die Testmanager diese Aktivitäten verstehen und überwachen und in der Lage sein, sie zu moderieren.

Die Testüberwachung sollte eine Risikoüberwachung beinhalten. Sie sollte überwachen, wie sich bekannte Qualitätsrisiken entwickeln, und auch eine Analyse neuer Qualitätsrisiken sowie die Anpassung des Risikoregisters umfassen.

Der Testmanager ist eine von mehreren Personen, die für die Minderung von Qualitätsrisiken zuständig sind. Die Risikominderung verteilt sich auf mehrere Testaktivitäten. Zum Beispiel werden die Ergebnisse der Analyse des Qualitätsrisikos bei der Testplanung verwendet, um das Testen auf die richtigen Bereiche und die Nutzung der richtigen Testverfahren zu konzentrieren. Bei der Testanalyse dienen die Risikostufen als Leitfaden für die Auswahl der zu überdeckenden Testbedingungen. Bei der Testdurchführung definiert eine risikobasierte Priorisierung die Reihenfolge der Testdurchführung.

1.3.2 Identifizierung von Qualitätsrisiken

Die Aufgabe der Testmanager besteht darin, die Risiken von den Stakeholdern zu sammeln. Die Stakeholder können Qualitätsrisiken durch eine oder mehrere der folgenden Verfahren identifizieren:

- Experteninterviews
- Unabhängige Bewertungen
- Retrospektiven
- Risiko-Workshops
- Brainstorming
- Checklisten
- Verwendung von Erfahrungswerten

Durch die Einbeziehung einer möglichst großen Bandbreite von Stakeholdern werden bei der Risikoidentifizierung oft die meisten kritischen Produktrisiken identifiziert. Es ist sehr wichtig, festzulegen, welche Stakeholder an dieser Phase beteiligt werden sollen. Entscheidend ist, dass die Liste der teilnehmenden Stakeholder umfassend und mit dem Projektmanager abgestimmt ist. Eine Risikoidentifizierung, bei der wichtige Stakeholder übersehen werden, kann sehr problematisch sein. Das Wichtigste ist, dass alle relevanten Stakeholder die Möglichkeit bekommen, sich zu beteiligen. Wenn sie nicht teilnehmen können, sollten sie zumindest die Möglichkeit haben, die Aufgabe zu delegieren. Wenn es sein kann, dass wichtige Stakeholder nicht vertreten sind, kann ein Kick-off-Meeting genutzt werden, um festzustellen, ob sie fehlen.

Beim risikobasierten Test ist es wichtig zu verstehen, dass die Risiken innerhalb der Testobjekte nicht gleichmäßig verteilt sind. Beispielsweise können kundenseitige Komponenten einer Self-service-Anwendung bei der Gebrauchstauglichkeit ganz andere Risiken aufweisen als Verwaltungskomponenten. Die Identifizierung der einzelnen Risiken der verschiedenen Testelemente ist eine wichtige Aufgabe bei der Testplanung.

Die Risikoidentifizierung führt häufig zu weiteren Aspekten (d. h. zur Identifizierung von Problemen, die keine Produktrisiken sind). Beispiele hierfür sind allgemeine Fragen oder Probleme in Bezug auf das Produkt oder das Projekt oder Probleme in referenzierten Dokumenten wie Anforderungen und Entwurfsdokumenten. Projektrisiken werden oftmals auch als weiteres Ergebnis der Identifizierung des Qualitätsrisikos erkannt, stehen aber nicht im Mittelpunkt des risikobasierten Testens. Die Testmanager können häufig eine wichtige Rolle spielen, wenn es darum geht, diese Aspekte hervorzuheben und deutlich zu machen, dass Qualität für alle von Belang ist. Schlechte oder fehlende Anforderungen sind oft ein Hinweis auf ein grundlegenderes Problem bei der Planung und Vorbereitung, da die Qualitätssicherung während des gesamten SDLC beteiligt ist.

1.3.3 Bewertung von Qualitätsrisiken

Die Bewertung von Qualitätsrisiken umfasst die Kategorisierung der Risiken nach Art (Produktrisiko oder Projektrisiko) und nach betroffenen Qualitätsmerkmalen.

Die Bestimmung der Risikostufe beinhaltet in der Regel die Bewertung der Eintrittswahrscheinlichkeit und das Schadensausmaß beim Eintreten des Risikos. Zu den Faktoren, die die Eintrittswahrscheinlichkeit von Qualitätsrisiken beeinflussen, gehören:

- Komplexität der Technologie, der Werkzeuge oder der Systemarchitektur
- Reife der Organisation
- Probleme der Mitarbeitenden bei Kompetenz, Verfügbarkeit, Motivation oder selbstständigem Arbeiten, einschließlich der Kenntnis des eingesetzten SDLC
- Konflikte innerhalb des Teams
- Vertragliche Probleme mit Lieferanten
- Probleme durch geografisch verteilte Teams
- Schwächen im Management oder der technischen Leitung
- Zeit-, Ressourcen-, Budget- und Managementdruck
- Mangel an frühzeitigen Aktivitäten zur Qualitätssicherung
- Hohe Änderungsraten bei der Testbasis, dem Produkt oder dem Personal.

Zu den Faktoren, die das Schadensausmaß des Risikos beeinflussen, gehören:

- Häufigkeit der Nutzung des betroffenen Merkmals
- Kritikalität des betroffenen Merkmals
- Kritikalität des betroffenen Geschäftsziels
- Schädigung des Rufs
- Verlust von Geschäftsumsatz
- Potenzielle finanzielle, ökologische oder soziale Schäden oder Haftung
- Zivilrechtliche oder strafrechtliche Sanktionen
- Schnittstellen- und Integrationsprobleme
- Fehlen von wirksamen Workarounds
- Sicherheitsbedürfnisse.

Die Testmanager kombinieren Eintrittswahrscheinlichkeit und Schadensausmaß des Risikos, um die Risikostufe zu bestimmen.

Wenn die Risikoanalyse auf umfangreichen und statistisch validen Risikodaten beruht, ist eine quantitative Bewertung angebracht. Zum Beispiel kann die Eintrittswahrscheinlichkeit des Risikos als Prozentsatz und das Schadensausmaß des Risikos als Betrag ausgedrückt werden. In einem solchen Fall kann die Risikostufe als das Produkt dieser beiden Faktoren berechnet werden. In der Regel können Eintrittswahrscheinlichkeit und Schadensausmaß des Risikos jedoch nur qualitativ auf Ordinalskalen ermittelt werden, z. B. als sehr hoch, hoch, mittel, niedrig oder sehr niedrig. Die Werte für die Eintrittswahrscheinlichkeit und das Schadensausmaß des Risikos werden dann in einer Risikomatrix kombiniert, um eine aggregierte Risikostufe zu erhalten. Diese aggregierte Risikostufe sollte als qualitative, relative Bewertung auf einer Ordinalskala interpretiert werden.

Sofern die Risikoanalyse nicht auf umfangreichen und statistisch validen Risikodaten beruht, wird die Risikoanalyse qualitativ sein und auf der subjektiven Wahrnehmung der Beteiligten hinsichtlich der Eintrittswahrscheinlichkeit und des Schadensausmaßes des Risikos basieren.

1.3.4 Minderung von Qualitätsrisiken durch geeignetes Testen

Bei der Softwareentwicklung ist das Testen der wichtigste Aspekt zur Minderung des Qualitätsrisikos. Durch Testen lässt sich die Eintrittswahrscheinlichkeit von Fehlerwirkungen verringern. Andere mögliche Maßnahmen zur Risikominderung sind ein Notfallplan (z. B. durch die Bereitstellung von Workarounds), die Risikoübertragung auf einen Dritten (z. B. den Anbieter einer Komponente) oder die Akzeptanz des Risikos.

Bei der **Testplanung** sollte der mit der Entwicklung und Durchführung eines Tests verbundene Aufwand proportional zur Risikostufe sein: Das Testen für höhere Risikostufen sollte früh beginnen und strengere Testverfahren verwenden, während das Testen für niedrigere Risikostufen später beginnen kann und weniger strenge Testverfahren verwenden sollte. Um das Gesamtrisiko durch Testen bestmöglich zu mindern, sollten die Testmanager die folgenden kontextbezogenen Faktoren analysieren und einen geeigneten Testansatz wählen:

- **Die Testelemente:** Verschiedene Testelemente innerhalb eines Testobjekts können unterschiedliche Stufen desselben Risikos haben, so dass ein Testobjekt nicht mit einheitlicher Strenge getestet werden muss.
- **Die Qualitätsmerkmale:** Risiken, die sich auf bestimmte Qualitätsmerkmale auswirken, sollten durch passende Testarten verringert werden, die einen spezifischen Testaufwand, Testumgebungen und Testfähigkeiten erfordern.
- **Die Teststufen und Testarten:** Bestimmte Risiken können nur dynamisch auf bestimmten Teststufen getestet werden, andere durch statische Tests (z. B. durch statische Analyse und Code-Reviews zur Wartbarkeit) oder durch eine Kombination aus beidem (z. B. durch eine Überprüfung der Architektur und dynamische Tests des integrierten Systems auf Sicherheitsschwachstellen). Das Testen jedes Testelements so früh wie möglich verringert das Risiko, kritische Fehlerzustände erst spät im Lebenszyklus zu finden, was sonst zu höheren internen Fehlerkosten und Verzögerungen führen würde.
- **Der SDLC:** Testaktivitäten haben ihre eigenen Eingangskriterien. Verschiedene SDLCs erfüllen sie zu unterschiedlichen Zeiten.
- **Das Testteam:** Die am besten qualifizierten Personen sollten die Testelemente mit den höchsten Risikostufen testen.
- **Die regulatorischen Anforderungen:** Einige sicherheitsbezogene Normen (z. B. die IEC 61508 Sicherheitsgrundnorm für funktionale Sicherheit) schreiben die Testverfahren und die erforderliche Überdeckung auf der Grundlage der Integritätsstufe vor. Die Testmanager müssen sicherstellen, dass diese Normen eingehalten werden.

Darüber hinaus sollte die Risikostufe Entscheidungen in Bezug auf die Qualitätssicherung beeinflussen, z. B. den Einsatz von Reviews für Arbeitsergebnisse wie Testfälle, den Grad der Unabhängigkeit des Testens von der Entwicklung und den Umfang der durchgeführten Regressionstests.

Während der **Testüberwachung und Teststeuerung** ermöglicht risikobasiertes Testen die Berichterstattung über den Testfortschritt in Bezug auf die verbleibende Risikostufe zu jedem beliebigen Zeitpunkt. Dies unterstützt das Entwicklungsteam und die Stakeholder bei der Überwachung und Steuerung der Softwareentwicklung, einschließlich der Freigabeentscheidungen auf der Grundlage der verbleibenden Risikostufe. Dazu ist es erforderlich, die Testergebnisse in Bezug auf die Risiken in einer für die Stakeholder verständlichen Weise zu berichten.

Während der **Testrealisierung** erfolgt die Priorisierung der Tests auf der Grundlage der Risikostufen. Während der Testdurchführung sorgt dies für eine frühzeitige Überdeckung der kritischsten Bereiche und für die Minderung der Risiken der höchsten Stufe.

- In einigen Fällen werden die Tests für die Durchführung in streng absteigender Reihenfolge der Risikostufen, die sie überdecken, priorisiert, beginnend mit der höchsten Stufe. Dieser Ansatz wird als „depth-first“ (Tiefentest-Ansatz) bezeichnet und ist dann angebracht, wenn es wichtig ist, die höchsten Risiken so früh wie möglich zu verringern.
- Alternativ wird für jedes Risiko mindestens ein Test mit der höchsten Priorität versehen. Alle anderen Tests werden entsprechend der von ihnen überdeckten Risikostufen priorisiert. Dieser Ansatz wird als „breadth-first“ (Breitentest-Ansatz) bezeichnet und ist dann angebracht, wenn die Stakeholder so früh wie möglich einen Gesamtüberblick über die Qualität des Produkts erhalten möchten.

In der Praxis beginnt das Testen häufig mit dem „depth-first“-Ansatz. Wenn die Zeit jedoch knapp wird, geht man zum „breath-first“--Ansatz über und testet alle verbliebenen Risikoelemente mindestens einmal.

Unabhängig davon, ob der risikobasierte Test in der Tiefe, in der Breite oder kombiniert durchgeführt wird, kann die für das Testen vorgesehene Zeit verbraucht werden, ohne dass alle geplanten Tests durchgeführt wurden. An diesem Punkt erleichtert der risikobasierte Test die Abgabe einer begründeten Empfehlung an das Management, ob die Tests ausgeweitet werden sollen oder das verbleibende Risiko akzeptiert werden soll.

1.3.5 Verfahren für risikobasierte Tests

Es gibt eine Reihe spezifischer Verfahren mit unterschiedlichem Formalitätsgrad, um risikobasierte Tests zu implementieren. Die Eignung eines Verfahrens hängt von Projekt-, Prozess- und Produktüberlegungen ab. Es gibt zwei grundlegende Arten von Verfahren: schwergewichtige und leichtgewichtige. In sicherheitskritischen Systemen werden sehr häufig schwergewichtige Verfahren verwendet. Bei nicht sicherheitskritischen Anwendungen werden normalerweise leichtgewichtige Verfahren eingesetzt.

Schwergewichtige Verfahren sind formal und verwenden definierte Prozesse und eine detaillierte Dokumentation. Sie beziehen breite Gruppen von Stakeholdern ein. Bei der Risikobewertung im Rahmen schwergewichtiger Verfahren werden detaillierte Faktoren für die Eintrittswahrscheinlichkeit und das Schadensausmaß des Risikos genutzt sowie mathematische Formeln, um die Eintrittswahrscheinlichkeit und das Schadensausmaß aus diesen Faktoren zu berechnen. Beispiele für schwergewichtige Verfahren sind:

- **Gefährdungsanalyse:** Weitet den analytischen Prozess auf frühere Phasen aus, indem sie versucht, die Gefahren zu identifizieren, die jedem Risiko zugrunde liegen.
- **Kostenanalyse der Fehleraufdeckung:** Bestimmt für jedes Qualitätsrisikos die Eintrittswahrscheinlichkeit einer Fehlerwirkung, die Verlustkosten bei Auftreten einer typischen Fehlerwirkung und die Kosten des Testens auf solche Fehlerwirkungen.
- **Fehlermöglichkeits- und Einflussanalyse (FMEA) und ihre Varianten:** Identifiziert Qualitätsrisiken, ihre möglichen Ursachen und ihre wahrscheinlichen Auswirkungen und ordnet dann Fehlerschweregrad, Priorität und Entdeckungsraten zu.
- **Fehlerbaum-Analyse:** Verknüpft potenzielle Fehlerwirkungen mit Fehlerzuständen, die diese verursachen können, anschließend mit Fehlhandlungen, die diese Fehlerzustände hervorrufen können, und fährt fort, bis die verschiedenen Grundursachen identifiziert sind.

Im Gegensatz dazu sind leichtgewichtige Verfahren des risikobasierten Testens weniger gründlich und erfordern einen geringeren Aufwand für das Testteam und die Stakeholder. Wie die schwergewichtigen Verfahren basieren auch sie auf der Einbeziehung der Stakeholder und nutzen die Ergebnisse der Risikoanalyse als Grundlage für die Testplanung und die Testbedingungen. Allerdings ist der Kreis der Stakeholder möglicherweise nicht so groß und die Risikofaktoren werden in der Regel auf einer Ordinalskala auf das Schadensausmaß und die Eintrittswahrscheinlichkeit des Risikos begrenzt. Einige dieser Testverfahren, wie Systematic Software Testing (SST) (Craig & Jaskiel, 2002) können nur angewendet werden, wenn eine Anforderungsspezifikation verfügbar ist. Andere Verfahren, wie Pragmatische Risikoanalyse und -management (PRAM) (Black, 2009) und Produktrisikomanagement (PRISMA) (van Veenendaal, The PRISMA Approach: Practical Risk-Based Testing, 2012), verwenden zwar die Anforderungen und/oder andere Spezifikationen als Eingabe für die Risikoanalyse, können aber auch vollständig auf der Basis von Stakeholder-Angaben funktionieren.

1.3.6 Erfolgsmetriken und Schwierigkeiten im Zusammenhang mit risikobasiertem Testen

In einer Retrospektive sollte das Testteam messen, inwieweit es die Vorteile des risikobasierten Testens verwirklicht hat. In vielen Fällen beinhaltet dies die Beantwortung einiger oder aller der folgenden Fragen unter Einsatz von Metriken und Befragung:

- Wurden die relevanten Stakeholder an der Risikoanalyse beteiligt oder vertreten?
- War die Einbeziehung der Beteiligten in die Risikoanalyse angemessen?
- Wenn es in der Produktion kritische Störfälle gegeben hat, die auf übersehene Fehler hinweisen, wurden diese behoben?
- Wurden bei der Testdurchführung die meisten hoch priorisierten Fehlerzustände früh gefunden?
- War das Testteam in der Lage, den Stakeholdern die Testergebnisse im Hinblick auf das Risiko zu erläutern?
- Hatten die ausgelassenen Tests eine geringere Risikostufe als die durchgeführten?

In den meisten Fällen führt ein erfolgreicher risikobasierter Test zu einer positiven Antwort auf all diese Fragen. Langfristig sollten Prozessverbesserungsziele für Erfolgsmetriken festgelegt werden, zusammen mit dem Bestreben, die Effizienz des Prozesses der Qualitätsrisikoanalyse zu verbessern.

Das Management von Risiken stößt oft auf unerwartete Schwierigkeiten, weil die Komplexität häufig übersehen wird.

- **Schwierigkeiten bei der Bewertung der Risikostufe:** Die Abschätzung des Schadensausmaßes und der Eintrittswahrscheinlichkeit des Risikos kann sehr schwierig sein. Lösung: Verwendung historischer Daten und Befragung der wichtigsten Stakeholder des Projekts nach ihrer Einschätzung.
- **Zaghafte Anfänge:** Die Einrichtung und Pflege eines angemessenen risikobasierten Testansatzes wird angesichts eines hohen kurzfristigen Erfolgsdrucks oft vernachlässigt. Lösung: regelmäßige Überwachung und Berichterstattung der Risiken an die Stakeholder.
- **Déjà-vu:** Bei jedem Projekt werden die gleichen Risiken angesprochen, was zu einer Gleichgültigkeit gegenüber Risiken führt. Lösung: Einbeziehung der richtigen Personen

zur Risikoidentifizierung und Beschränkung der Risikominderung auf solche Risiken, die als wichtig erachtet werden.

- **Wichtige Risiken werden übersehen:** Die Grundursache für dieses Problem liegt in der Regel darin, dass unerfahrene oder ungeeignete Personen in den Prozess eingebunden sind. Lösung: Einbeziehung geeigneter Personen und ihre Schulung.
- **Stakeholder-Fluktuation:** Stakeholder können im Laufe der Zeit wechseln und auch neue Risiken können auftauchen. Daher ist die Risikoanalyse eine fortlaufende, iterative Tätigkeit und sollte nicht nur einmal zu Beginn durchgeführt werden.

1.4 Die Teststrategie für das Projekt

Einführung

Dieser Lehrplan geht davon aus, dass eine organisationsweite Teststrategie vorhanden ist. Ihre Entwicklung und Wartung wird in ISO/IEC/IEEE-29119-2 (wo sie als „organisationsbezogene Testpraktiken“ bezeichnet wird) und in den Lehrplänen ISTQB® Expert Level – Test Management und ISTQB® Certified Tester – Agile Test Leadership at Scale detailliert dargestellt.

Falls keine organisationsweite Teststrategie vorhanden ist oder diese unvollständig ist, muss das Testmanagement versuchen, die fehlenden Details mit den relevanten Stakeholdern zu klären.

Im Kontext dieses Abschnitts stellt die Definition einer Projekt-Teststrategie ein Beispiel für jede Art von detaillierter Projekt-Teststrategie dar. Dies kann ein Release, ein Produkt oder jede andere Art von Systementwicklungs- oder Beschaffungsmaßnahme sein. Eine Projekt-Teststrategie (in der ISO/IEC/IEEE 29119-3 als "Teststrategie" bezeichnet) beschreibt den Testansatz in einem spezifischen Kontext, um die Ziele der Organisation zu erreichen, insbesondere im Hinblick auf Produktqualität und Testaktivitäten. Eine Teststrategie kann auch für eine einzelne Teststufe oder eine Testart existieren.

Die Projekt-Teststrategie stellt das wichtigste Ergebnis der Testplanung für ein Projekt dar und wird in der Regel in einem Testkonzept oder als Teil anderer Dokumente beschrieben. Eine Dokumentation der Teststrategie wird empfohlen, muss jedoch nicht zwingend in Form eines formellen Testkonzepts erfolgen. Der Bedarf an Dokumentation ist vom Kontext des Testens abhängig (siehe Abschnitt 1.2 „Der Kontext des Testens“). Wenn ein Projekt ein sequenzielles Entwicklungsmodell nutzt, wird die Teststrategie des Projekts üblicherweise – vorzugsweise im Testkonzept (siehe ISO/IEC/IEEE 29119-3) – dokumentiert. Die Dokumentation wird häufig auch von Verträgen, Vereinbarungen, Aufsichtsbehörden oder Gesetzen gefordert.

1.4.1 Auswahl eines Testansatzes

Die Projekt-Teststrategie definiert Leitlinien für sämtliche Testaktivitäten eines Projekts und enthält detaillierte Angaben zu Testzielen, Ressourcen, Zeitplänen und Verantwortlichkeiten. Diese Strategie muss auf die besonderen Anforderungen des Projekts zugeschnitten sein. Zu den wichtigsten Entscheidungen gehören die Auswahl von Teststufen, Testarten und Testverfahren für statische und dynamische Tests sowie weiterer Testverfahren (z. B. Testskripte, manuelle Tests, vergleichende Tests).

Theoretisch können alle Testarten auf jeder Teststufe durchgeführt werden und jedes Testverfahren kann in jeder Testart auf jeder Teststufe angewendet werden. In der Praxis haben die richtige Auswahl und Kombination dieser Möglichkeiten einen erheblichen Einfluss auf die Effektivität und Effizienz des Testens. Die Wartbarkeit von Codes kann beispielsweise oft effektiver und effizienter durch statische Analyse oder Code-Reviews beurteilt werden. Andererseits kann die Performanz aufgrund der Interaktion interner Komponenten besser durch Systemtestskripte bewertet werden. Zudem kann die Zweckmäßigkeit der Funktionalität besser durch gemeinsam mit den Benutzern entwickelte manuelle Abnahmetests überprüft werden. Die Wahl des adäquaten Testansatzes für eine Teststrategie kann ein komplexer Prozess sein, der durch die organisationsweite Teststrategie, den Projektkontext und weitere Aspekte beeinflusst werden kann.

Die Auswahl und Kombination von Teststufen, Testarten und Testverfahren ist daher von entscheidender Bedeutung für eine effektive Teststrategie im Projekt, da sie die Effizienz und Effektivität des Testens maßgeblich beeinflusst.

1.4.2 Analyse der organisationsweiten Teststrategie, des Projektkontexts und anderer Aspekte

Die Entwicklung einer Projekt-Teststrategie erfordert das vollständige Verständnis der organisationsweiten Teststrategie, des Projektkontexts sowie weiterer Faktoren oder Randbindungen im Zusammenhang mit dem Testen.

Bei der Wahl des geeigneten Testansatzes sind in der Regel die folgenden Faktoren zu analysieren:

- **Domäne:** Der Anwendungsbereich, für den das Produkt erstellt oder angepasst werden soll. Alle bereichsspezifischen Vorschriften, Standards und Praktiken können die Stringenz des Testens, die erforderliche Dokumentation und den Detaillierungsgrad des Testens beeinflussen. In der Pharmazie und Medizin liegt der Schwerpunkt des Testansatzes beispielsweise häufig auf intensiven Benutzerabnahmetests, die sich auf Risiken für die Gesundheit der Patienten konzentrieren, wobei die Testfälle auf den funktionalen Anforderungen der Benutzer basieren, während sich die Benutzerabnahmetests für webbasierte Versicherungsanwendungen auf die Gebrauchstauglichkeit und die Erhöhung der Wahrscheinlichkeit neuer Versicherungsverträge durch A/B-Tests konzentrieren können.
- **Organisationsweite Ziele und übergreifende Qualitätsmerkmale:** Zu den organisationsweiten Zielen kann die Notwendigkeit gehören, den Mehrwert des Testens nachzuweisen und den Grad der Testautomatisierung oder der Qualitätsmerkmale des Testprozesses, wie die Testreife oder die Effizienz der Fehlererkennung, zu erhöhen. Dies kann auch die Teststufen und Testarten vorgeben, die eingehalten werden müssen.
- **Die Projektziele und die Art des Projekts:** Die Projektziele (z. B. in Bezug auf Budget, Zeit und Qualität) im Gegensalz zu der Art des Projekts (d. h. kundenspezifische versus marktorientierte Produktentwicklung) beinhalten in der Regel Einschränkungen, Risiken und Chancen, die sich auf das Testen auswirken. Beispielsweise können enge Budget- und Zeitvorgaben den stringenten Einsatz von risikobasierten Tests erfordern, um Testfälle für die Testdurchführung zu priorisieren, während die Entwicklung eines kundenspezifischen Produkts möglicherweise Tests erfordert, die vordefinierte vertragliche Akzeptanzkriterien überdecken.
- **Testressourcen:** Alle Einschränkungen bezüglich der Verfügbarkeit von Testressourcen, einschließlich der Testwerkzeuge, der Testinfrastruktur, der Technologie und der Entwicklungsumgebung, die im Projekt verwendet werden, sowie des verfügbaren Personals und seiner Fähigkeiten müssen berücksichtigt werden (siehe Abschnitt 3.1 „Das Testteam“). Beispielsweise erfordert erfahrungsbasiertes Testen Tester mit guten Fachkenntnissen; mobile Anwendungen müssen in der Regel auf einer begrenzten Anzahl verschiedener Geräte getestet werden; der Einsatz von Testwerkzeugen kann durch die Anzahl der verfügbaren Lizenzen begrenzt sein.
- **Das für das Projekt verwendete Softwareentwicklungslebenszyklus-Modell:** Zur Bestimmung geeigneter Teststufen, des Testaufwands, geeigneter Eingangskriterien und Endekriterien (siehe ISTQB®-Lehrplan Foundation Level V4.0, Abschnitte 2.2 und 5.1). Ein Softwarelebenszyklus mit kontinuierlicher Integration erfordert mehr automatisierte

Tests als eine einmalige Entwicklung nach dem Wasserfallmodell. Daher können unterschiedliche Testarten und Testverfahren zum Einsatz kommen.

- **Schnittstellen mit anderen Systemen:** In einem System von Systemen ist die Abstimmung der Tests mit anderen Teams oder Projekten und die Auswahl geeigneter Teststufen insbesondere für Systemintegrationstests unerlässlich. Beispielsweise hilft risikobasiertes Testen bei der Priorisierung und Skalierung von Systemintegrationstests.
- **Verfügbarkeit von Testdaten:** Einschränkungen bei der Verfügbarkeit von Testdaten müssen berücksichtigt werden, wie z. B. der Bedarf an anonymisierten Testdaten aus der Produktion oder die Erstellung spezieller Testdaten, die möglicherweise schwer zu beschaffen sind und validiert werden müssen, wie z. B. Daten für KI-Tests. Modellbasiertes Testen kann beispielsweise die Erstellung und Verwaltung von Testdaten unterstützen.

Das Testmanagement sollte festlegen, welche Kombination von Testverfahren, Teststufen und Testarten am besten geeignet ist, um die organisationsweite Teststrategie, den Projektkontext und zusätzliche Faktoren oder Einschränkungen im Zusammenhang mit dem Testen angemessen zu berücksichtigen.

1.4.3 Definition der Testziele

Für jedes Testprojekt sollte ein Testkonzept definiert werden, das u. a. den Testumfang, die Testziele und die Endekriterien enthält. Der Testplan kann auf Release-Ebene, als Projekttestkonzept (auch Mastertestkonzept genannt) und ggf. auch als Stufentestkonzept für die verschiedenen Teststufen erstellt werden. Zusätzlich können Testkonzepte für die verschiedenen Qualitätsmerkmale, wie z. B. ein Testkonzept für IT-Sicherheit oder Performanz, definiert werden. Bei agiler Softwareentwicklung und hybriden Softwareentwicklungsprojekten kann ein Testkonzept für die Iteration vereinbart werden. Für jedes Release und jede Iteration kann im Testkonzept der Umfang der zu liefernden funktionalen Features und nicht-funktionalen Eigenschaften definiert, für jede Iteration fortgeschrieben und mit den Stakeholdern abgestimmt werden.

Im Zusammenhang mit den Features, die in einem Projekt zum Testen bereitgestellt werden, müssen die Testziele und die Endekriterien des Projekts definiert werden. Dies kann mithilfe von S.M.A.R.T. als Methode zur Definition von Zielen geschehen:

- S = spezifisch (specific): Ein Projekttestziel und das Endekriterium sollten klar und eindeutig sein.
- M = messbar (measurable): Es sollte quantifizierbar sein und über spezifische Kriterien zur Messung des Fortschritts verfügen, um feststellen zu können, ob das Testziel erreicht wurde.
- A = erreichbar (achievable): Es sollte unter Berücksichtigung der verfügbaren Ressourcen, des Zeitrahmens und der Fähigkeiten realistisch umsetzbar sein.
- R = relevant (relevant): Es sollte auf die allgemeinen Projektziele abgestimmt sein.
- T = rechtzeitig (timely): Es sollte einen definierten Zeitrahmen und eine definierte Frist für den Abschluss haben.

Die Testziele des Projekts sollten alle qualitativen und quantitativen Aspekte abdecken, soweit diese messbar oder bewertbar sind. Beispiele für Projekttestziele sind:

- Erreichen der definierten Endekriterien innerhalb des definierten Zeitrahmens.
- Erfüllung der Qualitätsziele des Unternehmens (z. B. gemessen als Key Performance Indicator (KPI) für die Anzahl der Kundenreklamationen für ein Produkt).
- Einhaltung der Regeln und Vorschriften der jeweiligen Branche.
- Sicherstellung der Verfügbarkeit der Daten nur für autorisierte Nutzer (z. B. durch Zugriffsrechte).
- Prüfung einer Datenmigration auf funktionale Vollständigkeit, funktionale Korrektheit, Performanz, Effizienz, Übertragbarkeit und IT-Sicherheit.
- Erhöhung der Testautomatisierung (z. B. um einen bestimmten Prozentsatz für Regressions- oder Performanztests zu erreichen).
- Erfolgreiches Refactoring des Codes und Nachweis, dass dadurch keine neuen Fehlerzustände entstanden sind (z. B. Beseitigung von schlecht strukturiertem Quellcode oder technischen Schulden unter Beibehaltung der bestehenden Funktionalität, nachgewiesen durch einen Regressionstest).
- Nachweis der IT-Sicherheit von Schnittstellen (z. B. durch Validierung von Extensible-Markup-Language-(XML-)Nachrichten gegen ihre XML-Schemadefinition, um sicherzustellen, dass schädliche Daten abgewiesen werden).
- Überprüfung der Gebrauchstauglichkeit einer Benutzerschnittstelle und Erreichen eines bestimmten Grades eines Teilmerkmals (z. B. durch Messung der Zeit, die benötigt wird, eine bestimmte Aufgabe in einem Onlineshop zu erledigen).

Neben der Erfassung und Messung der Testziele des Projekts sollte auch die Qualitätsbewertung durch Fachexperten und Stakeholder berücksichtigt werden.

Abhängig vom Projektkontext und den Testzielen können manchmal mehrere Testumgebungen mit den verfügbaren Ressourcen und/oder Testwerkzeugen erforderlich sein. Die Testumgebungen sind möglicherweise nicht alle gleichzeitig verfügbar. Dies muss bei der Formulierung der erreichbaren Testziele und Endekriterien berücksichtigt werden.

Je nach Projektkontext müssen zusätzliche Faktoren bei der Festlegung der Testziele und des Testumfangs des Projekts berücksichtigt werden (siehe Abschnitt 1.2 „Der Kontext des Testens“).

1.5 Verbesserung des Testprozesses

Einführung

Testen ist ein wichtiger Bestandteil der Softwareentwicklung und macht oft mindestens 30-40 % der gesamten Projektkosten aus. Neben den vielen (technischen) Herausforderungen, mit denen Softwareprojekte konfrontiert sind (z. B. zunehmende Komplexität und Größe, neue Technologien, eine Vielzahl von Geräten und Betriebssystemen sowie Sicherheitsschwachstellen), besteht die Notwendigkeit, die Effektivität und Effizienz des Testens zu optimieren und die Testprozesse entsprechend zu verbessern. Die Anwendung bewährter Best Practices und das Lernen aus eigenen Fehlern ermöglicht es, den Testprozess zu verbessern und Projekte erfolgreicher durchzuführen.

Ein Verbesserungsprozess ist auf der Organisationsebene in der Regel nützlicher als ein Verbesserungsprozess auf Projekt- oder Teamebene. Es ist dennoch auch möglich und vorteilhaft, eine Prozessverbesserung auf Projekt- oder Teamebene vorzunehmen. Sie sollte allerdings auf die Bedürfnisse des Projekts oder Teams zugeschnitten sein. Eine Testprozessverbesserung kann z. B. durch die Unzufriedenheit mit den Ergebnissen aktueller Tests, unerwartete Fehlerzustände, veränderte Umstände, Benchmark-Ergebnisse oder mangelnde Kommunikation ausgelöst werden. Es gibt unterschiedliche Ansätze, um das Testen zu verbessern (Bath & van Veenendaal, 2014). Einige dieser Ansätze werden in den folgenden Abschnitten beschrieben. Die in diesem Lehrplan beschriebenen Ansätze können sowohl auf sequenzielle als auch auf agile und inkrementelle Entwicklungsmodelle angewendet werden. Der ISTQB® Expert Level Improving the Test Process Syllabus bietet einen tieferen Einblick.

1.5.1 Der Testverbesserungsprozess (IDEAL)

Wenn beschlossen wurde, die Testprozesse zu verbessern, können die Aktivitäten zur Umsetzung der Testprozessverbesserung wie im IDEAL-Modell definiert werden, das auf ähnlichen Ideen wie der bekannte Plan-Do-Check-Act-(PDCA-)Zyklus basiert. IDEAL ist ein Akronym, das für Initiieren (Initiating), Diagnostizieren (Diagnosing), Etablieren (Establishing), Agieren (Acting) und Lernen (Learning) steht.

Obwohl IDEAL ursprünglich definiert wurde, um Verbesserungsaktivitäten auf organisatorischer Ebene zu unterstützen, kann es auch auf der Ebene eines Projekts oder eines Teams in der agilen Softwareentwicklung angewendet werden. Im Projektkontext müssen die Ziele der Aktivitäten (siehe unten) durch geeignete Maßnahmen erreicht werden. Der Hauptunterschied besteht wahrscheinlich in der Initiierungsphase, die auf Projekt- oder Teamebene viel kleiner ist als auf Organisationsebene. Die Diagnose durch eine Retrospektive und die Erstellung eines Plans fallen dabei womöglich deutlich kleiner aus als auf einer organisatorischen Ebene. Agieren und Lernen sind dagegen gleichermaßen auch auf Projekt- oder Teamebene von Bedeutung.

Initiierung des Verbesserungsprozesses

Zu Beginn des Verbesserungsprozesses werden die Ziele und der Umfang der Prozessverbesserungen von den Stakeholdern vereinbart.

Diagnose der aktuellen Situation

Der aktuelle Testprozess wird bewertet, um mögliche Verbesserungen zu identifizieren. Im Falle der modellbasierten Testprozessverbesserung (siehe Abschnitt 1.5.2 „Modellbasierte Testprozessverbesserung“) wird typischerweise ein Testverbesserungsmodell zur Bewertung verwendet, während bei der analytisch basierten Testprozessverbesserung Daten von

gesammelten Metriken analysiert werden (siehe Abschnitt 1.5.3 „Analytisch basierter Ansatz zur Testprozessverbesserung“).

Erstellung und Etablierung eines Plans zur Testprozessverbesserung

Ein Plan zur Testprozessverbesserung kann ein formelles Dokument sein, das alle detaillierten Aktionen auflistet, die durchgeführt werden müssen, um die Verbesserungen zu erzielen. Je nach Kontext kann der Plan auch sehr informell und leichtgewichtig sein. Die Liste der möglichen Prozessverbesserungen sollte priorisiert werden. Die Priorisierung kann auf der Grundlage eines Return on Investment (ROI), der Risiken, der Ausrichtung auf die Projekt- oder Teamstrategien und/oder der messbaren quantitativen oder qualitativen Nutzen, die erreicht werden sollen, erfolgen.

Maßnahmen zur Implementierung einer Testprozessverbesserung („Agieren“)

Um die Verbesserungen zu erreichen wird der Plan zur Testprozessverbesserung umgesetzt. Dazu gehören in der Regel Schulungen, die Pilotierung der geänderten Prozesse und schließlich deren vollständige Einführung in das Projekt oder Team.

Nutzenanalyse der implementierten Verbesserungsmaßnahmen

Nachdem alle Prozessverbesserungen vollständig umgesetzt wurden, sollte unbedingt verifiziert werden, welcher Nutzen tatsächlich erzielt wurde (geplant oder unerwartet). Erst wenn verstanden wurde, was funktioniert hat und was nicht, kann der nächste Verbesserungszyklus basierend auf diesen Informationen beginnen.“

1.5.2 Modellbasierte Testprozessverbesserung

Eine Annahme, die sowohl für die modellbasierte als auch für die analytisch basierte Verbesserung von Testprozessen gilt, ist, dass die Produktqualität in hohem Maße von der Qualität der Prozesse beeinflusst wird, die genutzt und angewendet werden. Bei der Anwendung der modellbasierten Testprozessverbesserung wird ein Testprozessverbesserungsmodell eingesetzt. Testprozessverbesserungsmodelle basieren auf Best Practices im Test und organisieren schrittweise die Testprozessverbesserung.

Es gibt mehrere empfohlene Prozessmodelle, die die Testprozessverbesserung unterstützen. Dazu gehören Test Maturity Model Integration (TMMi®) und TPI NEXT®.

Die modellbasierte Verbesserung kann auch auf Projektebene angewendet werden. In solchen Fällen konzentrieren sich die Bewertung und der Verbesserungsprozess speziell auf die im Modell definierten Testprozesse oder Kernbereiche, die sich auf die Aktivitäten auf Projektebene beziehen (z. B. Testplanung und Testentwurf) und lassen die Bereiche auf der Organisationsebene (z. B. Testrichtlinie und Testorganisation) oft weitgehend aus. Alternativ können auch die Praktiken, die sich auf die Organisationsebene beziehen, auf den Projektkontext zugeschnitten werden.

Für weitere Informationen zur modellbasierten Testprozessverbesserung siehe Lehrplan ISTQB® Expert Level Improving the Test Process Syllabus.

Test Maturity Model Integration

Das TMMi® (van Veenendaal & Cannegieter, 2011) (van Veenendaal, 2020) besteht aus fünf Reifegraden. Jede Reifegradstufe, mit Ausnahme der TMMi®-Stufe 1, enthält Testprozessbereiche und Verbesserungsziele. Darüber hinaus enthält TMMi® Praktiken, Teilpraktiken und Beispiele, um die Umsetzung zu erleichtern und zu unterstützen. TMMi® wurde ursprünglich als eine Ergänzung zum Capability Maturity Model Integration (CMMI®) entwickelt, wird aber heute weitgehend unabhängig von CMMI® verwendet.

Um die Anwendung von TMMi® in der agilen Softwareentwicklung zu erleichtern und zu unterstützen, wurde ein spezieller Leitfaden entwickelt, der erläutert, wie TMMi® in der agilen Softwareentwicklung nutzbringend eingesetzt und angewendet werden kann.

Weitere Informationen zu TMMi® siehe www.tmmi.org.

TPI NEXT®

Das TPI NEXT®-Modell (van Ewijk, 2013) definiert insgesamt 16 Kernbereiche, von denen jeder einen bestimmten Aspekt des Testprozesses abdeckt (z. B. Teststrategie, Testmetriken, Testwerkzeuge und Testumgebung). Vier Reifegrade sind im Modell für jeden dieser 16 Kernbereiche definiert.

Für die Bewertung der einzelnen Kernbereiche auf den verschiedenen Reifegraden werden spezifische Kontrollpunkte definiert. Die Bewertungsergebnisse werden in einer Reifematrix, die alle Kernbereiche abdeckt, zusammengefasst und visualisiert.

Weitere Informationen über TPI NEXT® siehe http://www.tmap.net/book/tpir-next.

1.5.3 Analytisch basierter Ansatz zur Testprozessverbesserung

Mit einem modellbasierten Verbesserungsansatz, wie er im vorherigen Abschnitt beschrieben wurde, werden Verbesserungen eingeführt, indem der Testansatz eines Projekts oder Teams mit externen Best Practices verglichen wird. Analytische Ansätze identifizieren Probleme auf der Grundlage von Daten aus dem Projekt oder Team selbst. Aus einer Analyse dieser Probleme lassen sich geeignete Verbesserungen ableiten. Analytische Ansätze können zusammen mit einem modellbasierten Ansatz verwendet werden, um Ergebnisse zu verifizieren und verschiedene Methoden zu nutzen.

Probleme können durch die Verwendung quantitativer und qualitativer Daten identifiziert werden. In diesem Abschnitt des Lehrplans werden analytische Ansätze vorgestellt, bei denen hauptsächlich quantitative Daten aus dem Testprozess und Daten aus Fehlerzuständen zur Bewertung des aktuellen Testansatzes verwendet werden. In Abschnitt 1.5.4 „Retrospektiven“ werden Retrospektiven vorgestellt, in denen qualitative Daten von Mitgliedern des Entwicklungs- und des Testteams im Hinblick darauf gesammelt werden, was gut und was nicht gut funktioniert.

Die Analyse von Daten ist wichtig für objektive Testprozessverbesserungen und bietet eine wertvolle Ergänzung für rein qualitative Beurteilungen, die andernfalls zu ungenauen Empfehlungen führen können, die nicht durch Daten gestützt sind. Die Anwendung eines analytischen Ansatzes zur Verbesserung beinhaltet meist eine quantitative Analyse des Testprozesses, um Problembereiche zu identifizieren und projektspezifische Ziele festzulegen. Die Definition und Messung von wesentlichen Metriken ist erforderlich, um den Testprozess zu bewerten und zu beurteilen, ob die Verbesserungen erfolgreich sind.

Beispiele für analytische Ansätze sind:

- Grundursachenanalyse (engl. Root cause analysis, RCA)
- Analyse anhand von Kennzahlen, Metriken und Indikatoren
- Der Ziel-Frage-Metrik-Ansatz (Goal-Question-Metric-(GQM-)Ansatz)

Bei der Grundursachenanalyse werden Probleme untersucht, um deren Grundursachen zu ermitteln. Dies ermöglicht die Identifizierung von Lösungen, die die Ursachen von Problemen beseitigen, anstatt nur die unmittelbar offensichtlichen Symptome zu bekämpfen. Ein mögliches Analyseverfahren besteht in der Auswahl einer geeigneten Menge von Fehlerzuständen, der Identifizierung von Clustern in diesen Daten und der Verwendung von

Ursache-Wirkungs-Diagrammen (auch Ishikawa- oder Fischgräten-Diagramme genannt), um die Grundursachen wichtiger Fehlerzustände zu ermitteln. Daraus werden dann Verbesserungsmaßnahmen abgeleitet, um das Auftreten ähnlicher Fehlerzustände zu verhindern.

Kennzahlen, Metriken und Indikatoren werden quantitativ verwendet, um zu beurteilen, wie gut der Testprozess im Projekt oder Team durchgeführt wird. Die wichtigsten zu berücksichtigenden Merkmale des Testprozesses sind Effektivität, Effizienz und Vorhersagbarkeit. Für jedes dieser Attribute können eine oder mehrere Metriken ausgewählt werden. Durch das Sammeln und Analysieren der entsprechenden Daten lassen sich die wichtigsten Bereiche identifizieren, die verbessert werden müssen.

Der GQM-Ansatz (Basili, et al., 2014) (van Solingen & Berghout, 1999) bietet ein Rahmenwerk für die Definition und Analyse von Metriken, die auf den Informationsbedarf der relevanten Projektbeteiligten zugeschnitten sind. Messziele definieren einen Qualitätsaspekt eines Objekts, der für einen bestimmten Zweck, eine bestimmte Perspektive und einen bestimmten Kontext gemessen werden soll. Diese Messziele werden durch Fragen verfeinert, die den Qualitätsaspekt aus der Sicht der Beteiligten definieren. Anschließend werden Metriken ausgewählt, die die notwendigen Informationen zur Beantwortung der Fragen liefern. Die für die Metriken gesammelten Daten dienen der Beantwortung der Fragen, der Bewertung der Messziele und erfüllen die Informationsbedürfnisse der Stakeholder.

Weitere Informationen über diese analytisch basierten Ansätze zur Testprozessverbesserung sind im ISTQB® Expert Level Improving the Test Process Syllabus zu finden.

1.5.4 Retrospektiven

Retrospektiven sind Besprechungen, in denen ein Team seine Methoden und seine Zusammenarbeit überprüft, die (guten und schlechten) Erfahrungen festhält und über Änderungen und Maßnahmen zur Verbesserung entscheidet (sowohl für das Testen als auch für andere Themen). Retrospektiven befassen sich beispielsweise mit dem Prozess, den Mitarbeitern, der Organisation, der Zusammenarbeit und den Werkzeugen.

Retrospektiven werden sowohl in sequenziellen Entwicklungsmodellen als auch in der agilen Softwareentwicklung eingesetzt. In sequenziellen Entwicklungsmodellen sind sie ein Teil des Testabschlusses. In diesem Zusammenhang zielen Retrospektiven darauf ab, Erkenntnisse zu gewinnen (Lessons Learned), um zukünftige Projekte besser zu steuern. In der agilen Softwareentwicklung werden Retrospektiven in der Regel am Ende jeder Iteration abgehalten, um zu diskutieren, was erfolgreich war und was verbessert werden muss und wie diese Verbesserungen in die nächste Iteration einfließen können. Retrospektiven werden vom gesamten Team durchgeführt und unterstützen so den Whole-Team-Ansatz (Whole Team Approach) und fördern die kontinuierliche Verbesserung. Dedizierte Retrospektiven können manchmal erforderlich sein, um Probleme anzugehen, die spezifisch für das Testen sind.

Eine typische Retrospektive besteht aus den folgenden Schritten:

Einleitung: Das Ziel und die Agenda der Retrospektive werden besprochen, und es wird eine Atmosphäre des gegenseitigen Vertrauens geschaffen, so dass Probleme ohne Schuldzuweisungen oder Beurteilungen (Blame Game) besprochen werden können.

Sammlung von Daten: Es werden Daten darüber erhoben, was während der Iteration oder des Projekts passiert ist. Es können qualitative Daten gesammelt werden, z. B. ein Zeitstrahl mit den wichtigsten Ereignissen, der Probleme aufzeigt und dokumentiert, wie die einzelnen Teammitglieder über diese Probleme denken. Darüber hinaus können quantitative Daten aus Metriken dargestellt werden. Zum Beispiel können Daten zum Testfortschritt, zur

Fehlerfindung, Testeffektivität, Testeffizienz und Vorhersagbarkeit von Tests einen objektiven Einblick in die Tests des Projekts oder der Iteration geben.

Ableitung von Verbesserungen: Die gesammelten Daten werden analysiert, um die aktuelle Situation zu verstehen und Ideen für Verbesserungen zu entwickeln. So kann z. B. eine Grundursachenanalyse durchgeführt werden, um die Ursachen der festgestellten Probleme zu ermitteln, und es kann ein Brainstorming abgehalten werden, um Ideen zur Beseitigung der Ursachen zu finden.

Entscheidung über Verbesserungsmaßnahmen: Die Maßnahmen zur Umsetzung der Verbesserungsideen werden abgeleitet und nach Prioritäten geordnet. Ein Verbesserungsplan und Verantwortlichkeiten werden festgelegt. Es können Ziele und zugehörige Metriken definiert werden, um die Auswirkungen der Maßnahmen auf die identifizierten Probleme zu bewerten. Wenn zu viele Verbesserungen auf einmal umgesetzt werden sollen, ist es schwierig, dies in nachvollziehbaren Schritten zu bewerkstelligen.

Abschluss der Retrospektive: In diesem letzten Schritt wird die Retrospektive selbst bewertet, um Stärken und Verbesserungen im retrospektiven Prozess zu identifizieren. Eine Retrospektive wird regelmäßig durchgeführt, insbesondere bei der agilen Softwareentwicklung. Auch bei der Retrospektive selbst wird eine kontinuierliche Verbesserung angestrebt.

Es ist wichtig, die Ergebnisse einer Retrospektive angemessen zu dokumentieren. Bei einem sequenziellen Entwicklungsmodell müssen Befunde, Schlussfolgerungen und Empfehlungen an die Mitglieder der Organisation verbreitet und auf verständliche Weise kommuniziert werden. Bei der agilen Softwareentwicklung sollten Probleme und Maßnahmen ebenfalls dokumentiert werden, um ein Review auf die Maßnahmen und ihrer potenziellen Auswirkungen auf die Probleme in der nächsten Iteration zu ermöglichen.

Tester, die Teil des (Projekt-)Teams sind, bringen ihre eigene Perspektive ein. Sie können testbezogene Probleme (und andere) ansprechen und das Team dazu anregen, über mögliche Verbesserungen nachzudenken.

Weitere Informationen finden sich in (Derby & Larsen, 2006).

1.6 Testwerkzeuge

Einführung

Es gibt drei Arten von Werkzeugen:

- Kommerzielle Werkzeuge
- Open-Source-Werkzeuge
- Maßgeschneiderte Werkzeuge

Bei der Auswahl eines Werkzeugs müssen alle Anforderungen und regulatorische Vorgaben des Unternehmens und der Stakeholder berücksichtigt werden.

Es gibt auch technische Werkzeuge wie Testautomatisierungswerkzeuge, Testmanagementwerkzeuge und viele mehr.

Beispiele für den Einsatz von Testwerkzeugen finden Sie im ISTQB®-Lehrplan Foundation Level V4.0.

1.6.1 Best Practices für die Einführung von Werkzeugen

Dieser Abschnitt enthält die notwendigen Schritte für die Bewertung und Einführung eines Testwerkzeugs.

Testmanager können an der Einführung eines Werkzeugs beteiligt sein oder den Einführungsprozess unterstützen oder erleichtern. Testmanager sind in der Regel für ein spezielles Testwerkzeug oder ein anderes auf das Testen bezogenes Werkzeug verantwortlich, z. B. ein Werkzeug für das Anforderungsmanagement, Fehlermanagement oder die Testüberwachung.

Es gibt allgemeine Best Practices und Überlegungen bei der Bewertung und Auswahl eines Testwerkzeugs. Nachfolgend eine Übersicht über diese Best Practices und Überlegungen:

- Identifizierung von Möglichkeiten zur Prozessverbesserung mithilfe geeigneter Werkzeuge.
- Verständnis für die in einem Unternehmen verwendeten Technologien und Wahl des mit diesen Technologien kompatiblen Werkzeugs.
- Verständnis, wie ein Werkzeug technisch und organisatorisch in den Softwareentwicklungslebenszyklus integriert werden kann.
- Bewertung des Werkzeugs anhand klarer Anforderungen und objektiver Kriterien.
- Beurteilung des Anbieters, wenn ein kommerzielles Werkzeug verwendet werden soll, Bewertung des Supports für nicht kommerzielle Werkzeuge (z. B. Open-Source-Werkzeuge).
- Identifikation der internen Anforderungen für Coaching, Betreuung oder Training im Umgang mit dem Werkzeug.
- Abwägung von Vor- und Nachteilen verschiedener Lizenzierungsmodelle.
- Durchführung einer Proof-of-Concept-Evaluierung als letzten Schritt.

Zu den allgemeinen Best Practices bei der Einführung und Nutzung eines Werkzeugs gehören:

- Durchführung eines Pilotprojekts, um die Auswahlkriterien und Anforderungen zu validieren und zu bewerten, wie das Werkzeug zu den bestehenden Prozessen und Arbeitsweisen passt.
- Anpassung und Verbesserung von bestehenden Prozessen an die Verwendung des Werkzeugs, ggf. auch Anpassung des Werkzeugs an bestehende Prozesse.
- Definition von Richtlinien für die Verwendung des Werkzeugs.
- Angebot von Schulungen, Coaching und Mentoring für die Anwender des Werkzeugs.
- Schrittweise Einführung des Werkzeugs in die Organisation.
- Implementierung einer Methode zur Sammlung von Informationen aus der Anwendung des Werkzeugs für künftige Verbesserungen.
- Definition der Verantwortlichkeit für das Werkzeug.

1.6.2 Technische und fachliche Aspekte für Werkzeugentscheidungen

Mehrere Faktoren beeinflussen die Entscheidung über die Implementierung und Verwendung eines Werkzeugs. Für Testmanager ist es wichtig, diese zu kennen und zu berücksichtigen.

- **Regulatorische Anforderungen und IT-Sicherheit:** Unternehmen, die sicherheits- oder unternehmenskritische Software entwickeln oder den gesetzlichen Vorschriften unterliegen, bevorzugen möglicherweise kommerzielle Werkzeuge, da diese häufiger die geforderten Standards erfüllen und über eine entsprechende Zertifizierung verfügen.
- **Finanzielle Aspekte:** Open-Source-Werkzeuge sind in der Regel günstiger in der Anschaffung, da sie von einer Community unterstützt und weiterentwickelt werden. Kommerzielle Werkzeuge können sowohl einen einmaligen Kaufpreis als auch wiederkehrende Lizenzkosten haben. Die Anschaffungskosten eines maßgeschneiderten Werkzeugs sind schwer zu bestimmen, da sie von den Anforderungen und dem Entwicklungsstadium des Werkzeugs abhängen. Neben den Anschaffungskosten müssen auch die Kosten für die Schulung und Wartung über die gesamte Lebensdauer eines Werkzeugs berechnet und berücksichtigt werden. Alle Werkzeuge können hohe Kosten für Wartung und Support verursachen.
- **Anforderungen der Stakeholder:** Es ist wichtig, die Anforderungen aller Gruppen von Stakeholdern zu sammeln, um das am besten geeignete Werkzeug zu bewerten und zu identifizieren. Kommerzielle Werkzeuge und Open-Source-Werkzeuge erfüllen nicht unbedingt alle Anforderungen im Detail. Maßgeschneiderte Werkzeuge können die beste Wahl sein, um alle individuellen Anforderungen zu erfüllen, insbesondere in Fällen, in denen kein anderes Werkzeug die erforderliche Funktionalität bietet.
- **Bestehende Softwarelandschaft und Werkzeugstrategie:** Die bestehende Zusammenstellung von Werkzeugen (Softwarelandschaft) und die damit verbundene Werkzeugstrategie müssen bewertet werden, da es möglicherweise bevorzugte oder gesperrte Anbieter, integrierte Systeme, die Abhängigkeiten zu anderen Produkten aufweisen, oder ein spezielles Full-Service-Supportmodell für die gesamte Softwarelandschaft mit spezifischen Regelungen gibt.

1.6.3 Auswahlprozess und Bewertung des Return on Investment (ROI)

Testwerkzeuge können eine langfristige Investition sein, die sich möglicherweise über viele Iterationen eines einzelnen Projekts erstreckt und/oder auf viele Projekte anwendbar ist. Ein potenzielles Werkzeug muss daher aus verschiedenen Blickwinkeln betrachtet werden.

- Für das höhere Management ist ein positiver ROI erforderlich.
- Für das Support- und Betriebsteam ist eine begrenzte, aber notwendige Anzahl von Werkzeugen, die unternehmensweit eingesetzt werden, vorzuziehen. Die Wartung und der Betrieb einer größeren Anzahl von Werkzeugen, die Überwachung ihrer Lizenzen und die Verwaltung der Werkzeuglandschaft sollten zeit- und kosteneffizient sein.
- Für die Projektleiter muss das Werkzeug einen messbaren Mehrwert für das Projekt oder die Organisation bringen.
- Für die Anwender des Werkzeugs ist die Gebrauchstauglichkeit sehr wichtig. Zur Gebrauchstauglichkeit gehören z. B. die Unterstützung für bestimmte Aufgaben, die Erlernbarkeit und die Bedienbarkeit.
- Für das Betriebspersonal ist die Wartbarkeit wichtig.

Die Funktionen müssen für jede geschäftliche und technische Art von Werkzeug analysiert werden. Verschiedene Perspektiven und Interessen haben Einfluss auf diese Analyse: Testmanagement, (technische) Testanalyse, Testautomatisierung oder Entwicklung. Die Person in der Organisation, die für das Werkzeug verantwortlich ist, muss sicherstellen, dass die Analyse durchgeführt wird und die oben genannten Punkte berücksichtigt werden.

Alle in den Testprozess eingeführten Werkzeuge sollten auch einen positiven Return on Investment (ROI) für das Unternehmen gewährleisten. Es liegt in der Verantwortung des Testmanagers, sich um die Berechnung und weitere Evaluierung des ROI zu kümmern. Bei der agilen Softwareentwicklung kann es in der Verantwortung des gesamten Entwicklungsteams liegen.

Vor der Anschaffung oder der Entwicklung eines Werkzeugs sollte eine Kosten-Nutzen-Analyse durchgeführt werden, um sicherzustellen, dass der Einsatz des Werkzeuges für das Team bzw. Unternehmen einen Nutzen erbringt. Diese Analyse sollte sowohl die wiederkehrenden als auch die einmaligen Kosten berücksichtigen.

Zu den nicht wiederkehrenden Aktivitäten und Kosten gehören die folgenden:

- Definition der Anforderungen an das Werkzeug zur Erreichung der Ziele
- Bewertung und Auswahl des richtigen Werkzeugs und Anbieters, Proof of Concept
- Kauf, Anpassung oder Entwicklung des Werkzeugs für die initiale Nutzung
- Festlegung von Richtlinien für die Verwendung des Werkzeugs
- Einführungsschulung für das Werkzeug
- Einbindung des Werkzeugs in die bestehende Werkzeuglandschaft
- Beschaffung der für den Betrieb und die Wartung des Werkzeugs erforderlichen Hardware/Software

Zu den wiederkehrenden Aktivitäten und Kosten gehören die nachfolgend aufgeführten:

- Wiederkehrende Gebühren für Lizenzierung und Support
- Kosten für Wartung

- Laufende Trainingskosten
- Portierung des Werkzeugs auf verschiedene Umgebungen

Die Opportunitätskosten sind ebenfalls zu berücksichtigen; d. h. die Zeit, die für die Evaluierung, Verwaltung, Training und Anwendung des Werkzeugs aufgewendet werden musste und stattdessen für die eigentlichen Testaufgaben hätte verwendet werden können. Daher werden möglicherweise mehr Testressourcen benötigt, bevor das Werkzeug für die beabsichtigten Aktivitäten eingesetzt werden kann.

Die folgenden Risiken in Bezug auf den ROI sollten bei der Auswahl der Werkzeuge berücksichtigt werden:

- Eine Unreife der Organisation kann zu einer ineffizienten Nutzung des Werkzeugs führen.
- Änderungen in der Wartungspolitik des Anbieters können negative Auswirkungen haben (z. B. Erhöhung der Arbeitsbelastung, Anstieg der Lizenzkosten, Änderung der Update-Intervalle).
- Höhere Kosten als erwartet.
- Geringerer Nutzen als erwartet.

Die folgenden Vorteile können für Testwerkzeuge gelten:

- Reduzierung manueller, sich wiederholender Tätigkeit (z. B. Regressionstests)
- Beschleunigung des Testzyklus durch Automatisierung
- Einsparung von Kosten für die Testdurchführung durch eine Verringerung der manuellen Tätigkeiten
- Erhöhte Überdeckung für bestimmte vom Werkzeug unterstützte Testarten
- Verringerung menschlicher Fehlhandlungen durch weniger manuelle Tätigkeiten
- Frühzeitiger Überblick über Teststatus und Testergebnisse

Weitere Befunde zu Nutzen und Risiken, insbesondere für Testautomatisierungswerkzeuge, finden Sie im ISTQB®-Lehrplan Foundation Level V4.0 und im ISTQB®-Lehrplan Advanced Level Testautomatisierungsentwickler.

Im Allgemeinen verwendet eine Testorganisation selten nur ein einziges Werkzeug. Der Gesamt-ROI für eine Organisation ist in der Regel eine Mischung aus dem ROI aller Werkzeuge, die zum Testen verwendet werden. Die Werkzeuge müssen Informationen austauschen und kooperativ arbeiten. Eine langfristige, umfassende Strategie für Testwerkzeuge, die Risiken, Kosten und Nutzen berücksichtigt, ist ratsam.

1.6.4 Lebenszyklus eines Werkzeugs

Es gibt vier verschiedene Phasen im Lebenszyklus eines Werkzeugs. Es muss ein Werkzeug-Administrator ernannt werden, der sicherstellt, dass die Aktivitäten dieser Phasen definiert, durchgeführt und verwaltet werden.

Die folgende Liste zeigt die wichtigsten Aktivitäten dieser Phasen:

- **Akquisition:** Zunächst einmal wurde die Entscheidung getroffen, ein Werkzeug auszuwählen. Im zweiten Schritt muss ein Verantwortlicher des Werkzeugs bestimmt werden. Diese Person trifft Entscheidungen über die Verwendung des Werkzeugs (z. B. Namenskonventionen für Arbeitselemente und wo diese Arbeitsprodukte gespeichert

werden). Die im Vorfeld getroffene Entscheidung kann einen erheblichen Einfluss auf den ROI des Werkzeuges haben.

- **Support und Wartung:** Die Verantwortlichkeit für die Wartung sollte beim Administrator des Werkzeugs oder einem speziellen Werkzeugteam liegen. Im Falle der Interoperabilität müssen der Datenaustausch und die Prozesse für die Zusammenarbeit und Kommunikation berücksichtigt werden. Auch Entscheidungen über die Sicherung und Wiederherstellung von Artefakten im Zusammenhang mit dem Werkzeug sind erforderlich.
- **Entwicklung:** Im Laufe der Zeit können die Umgebung, geschäftliche Anforderungen oder Entscheidungen des Anbieters Änderungen am Werkzeug erfordern. Je komplexer eine Betriebsumgebung für ein Werkzeug wird, desto leichter kann eine Änderung seine Verwendung erschweren.
- **Außerbetriebnahme:** Am Ende seiner Lebensdauer sollte das Werkzeug außer Betrieb genommen werden. In den meisten Fällen wird die vom Werkzeug bereitgestellte Funktionalität ersetzt und die Daten müssen erhalten und/oder archiviert werden. Dies kann auf einer Entscheidung des Anbieters beruhen oder wenn der Punkt erreicht ist, an dem die Vorteile und Chancen eines Wechsels zu einem neuen Werkzeug die Risiken und Kosten aufwiegen.

1.6.5 Werkzeug-Metriken

Objektive Metriken von Werkzeugen werden auf der Grundlage der Bedürfnisse des Testteams und anderer Stakeholder entworfen und gesammelt. Testwerkzeuge erfassen meist wertvolle Echtzeitdaten und reduzieren den Aufwand für die Datenerfassung. Diese Daten werden verwendet, um den gesamten Testaufwand zu verwalten und Bereiche mit Optimierungsbedarf zu identifizieren.

Verschiedene Werkzeuge sind auf die Erhebung unterschiedlicher Arten von Daten ausgerichtet. Beispiele hierfür sind:

- Testmanagementwerkzeuge können eine Vielzahl verschiedener Metriken in Bezug auf verfügbare Testelemente, Tests, geplante Tests sowie den aktuellen und vergangenen Status der Testdurchführung (z. B. bestanden, fehlgeschlagen, blockiert oder geplant) liefern.
- Anforderungsmanagementwerkzeuge bieten eine Verfolgbarkeit der Anforderungsüberdeckung durch bestandene und fehlgeschlagene Testfälle.
- Fehlermanagementwerkzeuge können Informationen über Fehlerzustände wie Status, Fehlerschweregrad, Priorität und Fehlerdichte von Testelementen liefern. Weitere wertvolle Daten wie der Fehlerfindungsanteil, die Teststufen, auf denen Fehlerwirkungen und Fehlerzustände entdeckt oder gefunden werden, und die ermittelte Durchlaufzeit für Fehler tragen zur Prozessverbesserung bei, werden aber nicht unbedingt nur vom Fehlermanagementwerkzeug bereitgestellt.
- Statische Analysewerkzeuge liefern unter anderem Metriken zur Codekomplexität.
- Performanztestwerkzeuge können wertvolle Informationen wie Reaktions- und Antwortzeiten und Fehlerwirkungen liefern sowie Ausfallraten bei Spitzenbelastungen.
- Werkzeuge zur Codeüberdeckung helfen dabei, zu verstehen, welche Teile des Testobjekts durch das Testen ausgeführt wurden.

- Obwohl Testwerkzeuge zum Sammeln von Metriken verwendet werden können, sollten sie sich auch selbst überwachen. In diesem Zusammenhang kann die Qualität des Testprozesses bewertet werden (z. B. die Anzahl der gefundenen Fehlerzustände mit und ohne Werkzeuge und die Überdeckung der Anforderungen).
- Werkzeuge, die die Effizienz des Testens messen (z. B. Dauer der Testdurchführung und Anzahl der durchgeführten Tests)

Weitere Einzelheiten über die Erfassung und Verwendung von Metriken bietet dieser Lehrplan in Abschnitt 2.1 Testmetriken.

2 Das Produkt managen – 390 Minuten

Schlüsselbegriffe

Anomalie, Fehlerbericht, Fehlerwirkung, Fehlerworkflow, Fehlerzustand, Metrik, Testfortschritt, Testschätzung, Testziel

Domänenspezifische Schlüsselbegriffe

Breitband-Delphi, Drei-Punkt-Schätzung, Planungspoker

Lernziele für Kapitel 2: Der Lernende kann …

2.1 Testmetriken

TM-2.1.1 (K2) … Beispiele für Metriken zur Erreichung der Testziele nennen.

TM-2.1.2 (K2) … erklären, wie man den Testfortschritt mithilfe von Testmetriken steuern kann.

TM-2.1.3 (K4) … Testergebnisse analysieren, um Testberichte zu erstellen, die es Stakeholdern ermöglichen, Entscheidungen zu treffen.

2.2 Testschätzung

TM-2.2.1 (K2) … die Hauptmerkmale erklären, die bei der Testschätzung berücksichtigt werden müssen.

TM-2.2.2 (K2) … Beispiele für Faktoren nennen, die Testschätzungen beeinflussen können.

TM-2.2.3 (K4) … für einen vorgegebenen Kontext ein geeignetes Verfahren oder einen geeigneten Ansatz für die Testschätzung auswählen.

2.3 Fehlermanagement

TM-2.3.1 (K3) … einen Fehlermanagementprozess samt Fehlerworkflow umsetzen, der zur Überwachung und Steuerung von Fehlerzuständen verwendet werden kann.

TM-2.3.2 (K2) … den Prozess und die erforderlichen Teilnehmer für ein effektives Fehlermanagement erklären.

TM-2.3.3 (K2) … die Besonderheiten des Fehlermanagements in der agilen Softwareentwicklung erklären.

TM-2.3.4 (K2) … die Herausforderungen für das Fehlermanagement bei der hybriden Entwicklung von Software erklären.

TM-2.3.5 (K3) … die Daten und Klassifizierungen verwenden, die während des Fehlermanagements gesammelt werden sollten.

TM-2.3.6 (K2) … erklären, wie Statistiken aus Fehlerberichten verwendet werden können, um daraus Prozessverbesserungen abzuleiten.

2.1 Testmetriken

Einführung – Warum gibt es Testmetriken?

Es gibt ein englisches Management-Sprichwort: "What gets measured, gets done." Was nicht gemessen wird, wird demnach wahrscheinlich auch nicht getan, da es nicht leicht ignoriert werden kann. Daher ist es wichtig, für jedes Vorhaben, auch für das Testen, eine Auswahl geeigneter Metriken festzulegen.

Testziele sind die Antwort auf die Frage, warum getestet wird (siehe Abschnitt 1.4 „Die Teststrategie für das Projekt“). Um festzustellen, ob die Testziele erreicht wurden, muss man einen Weg finden, sie zu messen. Testmetriken sind die Indikatoren, die uns helfen, diese Frage zu beantworten.

Testmetriken können wie folgt kategorisiert werden:

- **Projektmetriken** messen den Fortschritt anhand von bestehenden Endekriterien, wie z. B. dem Prozentsatz der ausgeführten, bestandenen und fehlgeschlagenen Tests.
- **Produktmetriken** messen Produktattribute wie den Grad, zu dem das Produkt die Qualitätserwartungen der vorgesehenen Nutzer erfüllt.
- **Prozessmetriken** messen die Leistungsfähigkeit des Testprozesses und die Effektivität des Testens. Prozessmetriken werden daher verwendet, um über prozessbezogene Effektivität und Effizienz zu berichten.

Weitere Informationen zum Management von Produkt- und Prozessmetriken finden Sie im ISTQB® Expert Level Test Management Syllabus.

Weitere Informationen über die Verwendung von Metriken für den Testprozess finden Sie im ISTQB® Expert Level Improving the Test Process Syllabus.

In den folgenden Abschnitten werden die Metriken für Testplanung, Testüberwachung, Teststeuerung und Testabschluss behandelt. Dies sind die vier wichtigsten Managementaktivitäten im Zusammenhang mit Metriken.

2.1.1 Metriken für Testmanagementaktivitäten

Der Lehrplan Certified Tester Advanced Level Test Management konzentriert sich auf die folgenden allgemeinen Aktivitäten des Testmanagements:

- Testplanung
- Testüberwachung und Teststeuerung
- Testabschluss (siehe Abschnitt 1.1 „Der Testprozess“)

Das Testmanagement muss in der Lage sein, eine Reihe von Metriken für die Testüberwachung, die Teststeuerung und den Testabschluss als Teil der Testplanungsaktivitäten zu definieren. Jede Metrik muss definiert, gemessen, überwacht und ausgewertet werden, wobei darüber Bericht erstattet wird.

Während der Testplanung werden geeignete Metriken definiert, die den Testzielen der Teststrategie des Projekts entsprechen.

Die Metriken, die während der Testüberwachung und Teststeuerung verwendet werden, können sich von denen unterscheiden, die bei Testabschluss eingesetzt werden. Bei der Testüberwachung und Teststeuerung geht es bei den Metriken um den Fortschritt der Testaktivitäten. Beim Testabschluss sollten die Testziele erreicht werden. Eine oder mehrere

Metriken können kombiniert werden, um die Endekriterien zu bewerten, die bestimmten Testzielen zugeordnet sind.

Die folgende Tabelle enthält Beispiele für Metriken (es gibt viele weitere), die im Testmanagement verwendet werden:

Metrik (geplant/überwacht für definierte Meilensteine)	**Testüberwachung und Teststeuerung**	**Testabschluss**
Anforderungsüberdeckung	X	X
Produktrisikoüberdeckung	X	X
Codeüberdeckung	X	
Tatsächlicher vs. geplanter Aufwand (in Stunden) für Testaktivitäten	X	
Prozentsatz der ausgeführten Testfälle pro Status (z. B. fehlgeschlagen, blockiert) im Vergleich zu den geplanten Testfällen	X	X
Anzahl der insgesamt behobenen Fehler im Vergleich zur Anzahl der insgesamt gefundenen Fehlerzustände	X	
Tatsächliche automatisierte Testfälle im Vergleich zu den geplanten automatisierten Testfällen		X
Tatsächliche vs. geplante Testkosten	X	

Tabelle 2: Beispiele für Metriken, die bei Testmanagementaktivitäten verwendet werden

Die in einer bestimmten Testaktivität verwendete Metrik ist in Tabelle 2 aufgeführt. Metriken mit einem X in Testüberwachung und Teststeuerung dienen in erster Linie der Messung des Fortschritts und sie werden in Testfortschrittsberichten (siehe auch CTFL V4.0 Syllabus) genutzt. Metriken mit einem X in der Spalte Testabschluss werden in erster Linie zur Messung des Erreichens der Testziele eingesetzt und im Testabschlussbericht (siehe auch ISTQB®-Lehrplan Foundation Level V4.0) gemeldet. Metriken mit einem X in beiden Spalten können für beides verwendet werden.

Es gibt auch Metriken zur Überwachung der Effektivität von Tests (z. B. Fehlerfindungsanteil (Defect Detection Percentage, DDP)).

DDP wird im ISTQB® Expert Level Improving the Test Process Syllabus, speziell im Abschnitt 4.4.2.1 Defect Detection Percentage behandelt.

2.1.2 Überwachung, Steuerung und Abschluss

Testmetriken sind Indikatoren, die u.a. zeigen, wie weit der Test fortgeschritten ist und ob die Endekriterien oder die damit verbundenen Testaufgaben erfüllt wurden.

Bei der Testüberwachung werden Daten über den Test erfasst, ausgewertet und beurteilt. Sie dient dazu, den Testfortschritt zu bewerten und die Erfüllung der Endekriterien oder der

zugehörigen Testaktivitäten zu überprüfen (siehe Abschnitt 1.1.2 „Testüberwachungs- und Teststeuerungsaktivitäten“). Die Endekriterien werden aus den Testzielen abgeleitet.

Die Teststeuerung nutzt die Informationen der Testüberwachung, um Anleitungen und Korrekturmaßnahmen für effektives und effizientes Testen zu liefern. Beispiele für die Teststeuerung sind die erneute Priorisierung von Tests, wenn ein identifiziertes Risiko zu einem Problem wird, die Neubewertung, ob ein Testelement die Eingangskriterien oder die Endekriterien aufgrund von Nacharbeiten erfüllt, die Anpassung des Testzeitplans, um auf eine Verzögerung bei der Lieferung der Testumgebung zu reagieren, und das Hinzufügen neuer Ressourcen, wenn und wo dies erforderlich ist.

Beim Testabschluss werden Daten aus abgeschlossenen Testaktivitäten gesammelt, um die gewonnenen Erkenntnisse, Testmittel und andere relevante Informationen zu konsolidieren. Der Testabschluss erfolgt an Projektmeilensteinen wie dem Abschluss einer Teststufe, dem Abschluss einer Iteration, dem Abschluss (oder Abbruch) eines Testprojekts, der Freigabe eines Produkts oder dem Abschluss eines Wartungsrelease.

Zu den gängigen Metriken für das Testmanagement gehören Metriken über den Projektfortschritt und Metriken, die den Fortschritt gegenüber dem geplanten Zeitplan und Budget, die aktuelle Qualität der Testelemente und die Effektivität der Tests gegenüber den Testzielen oder Iterationszielen anzeigen.

2.1.3 Testberichterstattung

Das Testmanagement sollte wissen, wie Metriken zu interpretieren und zu verwenden sind, um den Teststatus zu verstehen und zu berichten. Für höhere Teststufen wie Systemtests, Systemintegrationstests, Abnahmetests und IT-Sicherheitstests besteht die primäre Testbasis in der Regel aus Arbeitsergebnissen wie Anforderungsspezifikationen, Anwendungsfällen, User Stories und Produktrisiken. Metriken zur strukturellen Überdeckung sind eher für niedrigere Teststufen wie Komponententests (z. B. Anweisungsüberdeckung) und Komponentenintegrationstests (z. B. Schnittstellenüberdeckung) geeignet. Während das Testmanagement Metriken zur Codeüberdeckung verwenden kann, um zu messen, inwieweit Tests die Struktur des zu testenden Systems abdecken, sollte die Berichterstattung über Testergebnisse auf höheren Stufen auf den spezifischen Kontext und die Bedürfnisse des Projekts zugeschnitten sein. Zum Beispiel können in Umgebungen, die sich häufig ändern, Metriken zur Codeüberdeckung nützlich sein, um die Auswirkungen von Codeänderungen auf die Testsuite zu überwachen und potenzielle Lücken oder Risiken zu erkennen. Darüber hinaus sollte das Testmanagement verstehen, dass, selbst wenn Komponententests und Komponentenintegrationstests eine strukturelle Überdeckung von 100% erreichen, Fehlerzustände und Qualitätsrisiken verbleiben, die auf höheren Teststufen behandelt werden müssen.

Das Ziel der Berichterstattung über Metriken ist es, für das Management ein unmittelbares Verständnis der Informationen zu schaffen. Metriken können als eine Momentaufnahme zu einem bestimmten Zeitpunkt oder als Entwicklung einer Metrik im Laufe der Zeit berichtet werden, um Trends zu bewerten.

Produktrisiken, Fehlerzustände, Testfortschritt, die Überdeckung und die damit verbundenen Kosten und der Testaufwand werden am Ende des Projekts auf spezifische Weise gemessen und berichtet.

Nachfolgend sind Beispiele für Metriken aufgeführt, die für verschiedene Zwecke verwendet werden können:

Zu den Metriken über Produktrisiken gehören:

- Prozentualer Anteil der Risiken, bei denen alle Tests bestanden wurden
- Prozentualer Anteil der Risiken, bei denen einige oder alle Tests fehlgeschlagen sind
- Prozentualer Anteil der noch nicht vollständig getesteten Risiken

Diese Metriken können dazu verwendet werden, die Qualität der Testbasis und die Effektivität der Testfälle bei der Überdeckung der Produktrisiken zu bewerten.

Zu den Metriken über Fehlerzustände gehören:

- Kumulierte Anzahl der behobenen Fehlerzustände gegenüber der kumulierten Anzahl der Fehlerzustände
- Aufschlüsselung der Anzahl oder des Prozentsatzes der Fehlerzustände nach Kategorien:
  - Testelemente oder Komponenten
  - Quelle des Fehlerzustands (z. B. Anforderungsspezifikation, neue Funktion oder Regression)
  - Getestete Versionen
  - Teststufe oder Iteration der Entstehung, Erkennung und Behebung
  - Priorität/Schweregrad
  - Grundursache
  - Status (z. B. offen, abgelehnt, in Bearbeitung, gelöst, geschlossen)

Diese Metriken können dazu verwendet werden, den Fehlererkennungs- und -behebungsprozess zu überwachen, die Bereiche mit hoher Fehlerdichte oder hohem Fehlerschweregrad zu identifizieren und die Effizienz und Effektivität der Tests zu bewerten.

Metriken über den Testfortschritt umfassen:

- Testausführungsstatus: Gesamtzahl der geplanten, realisierten, ausgeführten, bestandenen, fehlgeschlagenen, blockierten und übersprungenen Tests.
- Testaufwand: Anzahl der für das Testen tatsächlich aufgewendeten Ressourcenstunden gegenüber den geplanten Ressourcenstunden.

Zu den Metriken über die Überdeckung gehören:

- Anforderungsüberdeckung: Der Prozentsatz der Anforderungen, die durch Testfälle überdeckt sind.
- Produktrisikoüberdeckung: Der Prozentsatz der identifizierten Produktrisiken, die durch Testfälle überdeckt werden.
- Codeüberdeckung: Der Prozentsatz der Codeanweisungen, Zweige, Pfade oder Bedingungen, die von Testfällen ausgeführt werden.

Zu den Metriken über die verbleibenden Risiken und den Testkosten gehören:

- Verbleibende Risiken für nicht getestete Komponenten: Die potenziellen Auswirkungen und die Wahrscheinlichkeit von Fehlerzuständen in den nicht getesteten Komponenten.
- Testkosten: Die tatsächlichen Testkosten im Vergleich zu den geplanten Testkosten.

Außerdem ist es sinnvoll, Metriken aus verschiedenen Kategorien zu kombinieren (z. B. eine Metrik, die die Korrelation zwischen den Trends der offenen Fehler und den Trends der ausgeführten Tests anzeigt, oder eine Metrik, die die Qualität der Testbasis anhand der Anzahl der in den Anforderungen gefundenen Fehlerzustände anzeigt). Wenn die Testdurchführung fortgesetzt wird und immer weniger Fehlerzustände identifiziert werden, kann die Entscheidung getroffen werden, die Tests zu beenden. Diese Entscheidung sollte sich auf die Berichterstattung über die Metriken und die vereinbarten Endekriterien stützen.

2.2 Testschätzung

Einführung

Im Projektmanagement gibt es Best Practices für die Schätzung der System- und Softwareentwicklung, die sich auf alle Arten von Ressourcen (z. B. Kosten, Personal oder Zeit) beziehen. Testschätzung ist die Anwendung dieser Best Practices auf das Testen in Zusammenhang mit einem Projekt oder einer Aufgabe.

2.2.1 Hauptmerkmale der Testschätzung

Bei der Testschätzung, als Aktivität des Testmanagements, wird geschätzt, wie viel Zeit, Aufwand und Kosten für die Durchführung einer Aufgabe erforderlich sind. Die Testschätzung ist eine der wesentlichsten und wichtigsten Aufgaben des Testmanagements.

Die Hauptmerkmale der Testschätzung im Testmanagement sind:

- **Der Aufwand** wird normalerweise in Personenstunden oder Story Points berechnet, die für die Durchführung der Testaufgaben des Projekts erforderlich sind. Oft sind Testaufwand und Testdauer (verstrichene Zeit) nicht identisch, so dass das Testmanagement die Gesamtdauer der Aktivität schätzen muss . Wie viele Personenstunden werden für die Aufgabe benötigt?
- **Die Zeit,** die benötigt wird, um das Projekt abzuschließen. Zeit ist eine kritische Ressource im Projekt. Bei der Testplanung muss der Testaufwand in Kalendertagen und in Arbeitstagen geschätzt werden. Jedes Projekt hat Meilensteine und einen Endtermin. Wie lange wird es dauern, das Testprojekt abzuschließen?
- **Die Kosten** sind das Projektbudget. Es umfasst die Ausgaben für die Testressourcen, Testwerkzeuge und Testinfrastruktur. Was wird das Testprojekt kosten?

Testen ist oft ein Teilprojekt innerhalb eines (größeren) Projekts und manchmal auf mehrere Teststandorte (z. B. Testzentren) verteilt. Um eine Testschätzung durchzuführen, müssen zunächst die Teststufen, Testaktivitäten und Testaufgaben identifiziert werden. Anschließend wird das Testprojekt in seine hauptsächlichen Testaktivitäten (z. B. Testplanung und Testdurchführung) innerhalb des Testprozesses unterteilt (siehe ISTQB®-Lehrplan Foundation Level V4.0). In agilen Projekten werden Testaktivitäten oft als Teil der Entwicklungsarbeit geschätzt und nicht als separate Werte. Der nächste Schritt ist die Schätzung des erforderlichen Testaufwands für die Fertigstellung der Aufgaben oder der Arbeitsprodukte und der daraus zu erwartenden Kosten.

Da Testen in einem Projekt eine Teilaktivität darstellt, gibt es naturgemäß immer Randbedingungen im Projekt, die eine Beeinflussung und Kompromisse nach sich ziehen. Diese Werte können nicht willkürlich manipuliert werden. Dies wird im Qualitätsmanagement als Zeit-Kosten-Qualitäts-Dreieck bezeichnet. Im Projektmanagement umfasst das Zeit-Kosten-Qualitäts-Dreieck drei Werte, die voneinander abhängen, d. h., sie sind eng miteinander verbunden und beeinflussen sich gegenseitig. Dieser Zusammenhang ist häufig in Projektszenarien zu beobachten.

2.2.2 Faktoren, die den Testaufwand beeinflussen können

Bei der Schätzung des Testaufwands geht es darum, den Arbeitsaufwand für die Testaktivitäten vorherzusagen, die zur Erreichung der Testziele für ein bestimmtes Projekt, ein bestimmtes Release oder eine bestimmte Iteration erforderlich sind. Zu den Faktoren, die den Testaufwand beeinflussen, gehören unter anderem folgende Eigenschaften des zu testenden Produkts und des Entwicklungs- und Testprozesses:

Produkt:

- Die Qualität der Testbasis
- Die Größe des zu testenden Produkts (d. h. des Testobjekts)
- Die Komplexität der Produktdomäne (z. B. Umgebung, Infrastruktur und Historie)
- Die Anforderungen an das Testen von Qualitätsmerkmalen (z. B. Sicherheit und Zuverlässigkeit)

Diese produktbezogenen Faktoren können die Testschätzungen beeinflussen, da sie einen spezifischen Kontext für das zu testende System schaffen.

Entwicklungsprozess:

- Die Stabilität und Reife der Entwicklungsprozesse im Unternehmen
- Das verwendete Entwicklungsmodell (z. B. agiles, iteratives oder hybrides Softwareentwicklungsmodell)
- Die materiellen Faktoren (z. B. Verfügbarkeit von Testautomatisierung, Testwerkzeugen und Testumgebungen)

Diese auf den Entwicklungsprozess bezogenen Faktoren können die Testschätzungen beeinflussen, da das Testen direkt mit der Entwicklung zusammenhängt.

Menschen:

- Zufriedenheit der Mitarbeiter (z. B. durch Urlaubszeiten, andere erwartete Leistungen)
- Die Kompetenzen und Erfahrungen der beteiligten Personen, insbesondere in Bezug auf ähnliche Projekte und Produkte (z. B. Fachwissen)

Menschen sind die wichtigste Ressource, deshalb sollte jede Unbeständigkeit berücksichtigt werden. Daher ist das Personal ein wichtiger Faktor bei der Schätzung des Testaufwands. Siehe auch Abschnitt 3.1 „Das Testteam“ in diesem Lehrplan.

Testergebnisse:

- Anzahl und Fehlerschweregrad der während der Testdurchführung aufgedeckten Fehler
- Der Umfang der erforderlichen Nacharbeit für die Fehlerbehebung

Statistiken aus früheren Projekten unterstützen die Testschätzung. Wenn diese Faktoren bekannt sind, ermöglichen sie eine Schätzung mit genaueren Werten.

Testkontext:

- Verteilung des Testens auf mehrere Niederlassungen, die Zusammensetzung und der Standort der Teams, die Komplexität des Projekts (z. B. mehrere Teilsysteme)
- Die Art des Arbeitens (z. B. virtuell oder vor Ort)

Kontextbezogene Faktoren sind diejenigen, die die gesamte Testschätzung beeinflussen. Siehe auch Abschnitt 1.2 „Der Kontext des Testens“.

2.2.3 Auswahl der Testschätzverfahren

Die Testschätzung sollte alle am Testprozess beteiligten Aktivitäten abdecken. Die geschätzten Werte für Kosten, Aufwand und vor allem Dauer der Testdurchführung sind oft die wichtigsten Faktoren für das Testmanagement, da diese das Projekt beeinflussen werden.

Schätzungen für die Dauer der Testdurchführung sind jedoch in der Regel schwierig, wenn die Gesamtqualität der Software gering oder unbekannt ist. Darüber hinaus beeinflussen Vertrautheit und Erfahrung mit dem Produkt wahrscheinlich die Qualität der Schätzungen. Eine gängige Praxis ist es, die Anzahl der Testfälle zu schätzen, die sich aus der Testbasis (z. B. Anforderungen oder User Stories) ergeben. Die Annahmen, die bei der Testschätzung gemacht werden, sollten immer als Teil der Schätzung dokumentiert werden. Verfahren oder Ansätze zur Testschätzung können entweder metrik- oder expertenbasiert sein. Weitere Details über Verfahren zur Testschätzung werden im ISTQB®-Lehrplan Foundation Level V4.0 erläutert.

In den meisten Fällen muss die Schätzung nach ihrer Erstellung zusammen mit einer Begründung der Projektleitung vorgelegt werden. Häufig werden einige Eingabeparameter geändert (z. B. der Testumfang), was zu einer Anpassung der Schätzung führt. Im Idealfall stellt die endgültige Testschätzung die bestmögliche Balance zwischen den Unternehmenszielen und den Projektzielen in Bezug auf Qualität, Zeitplan, Budget und Funktionalität dar.

Es ist wichtig zu beachten, dass jede Schätzung auf den zum Zeitpunkt ihrer Erstellung verfügbaren Informationen basiert. Zu Beginn eines Projekts können die Informationen sehr begrenzt sein. Außerdem können sich diese Informationen im Laufe der Zeit ändern. Die Schätzungen sollten daher regelmäßig aktualisiert werden, um neue und geänderte Informationen zu berücksichtigen.

Die Wahl des Schätzverfahrens hängt von verschiedenen Faktoren ab, wie z. B.

- **Schätzfehler:** Einige Verfahren bieten die Möglichkeit, die Standardabweichung zu berechnen, die ein Maß für die Unsicherheit oder Variabilität der Schätzung ist. Beispielsweise werden bei der Drei-Punkt-Schätzung die optimistischste, die pessimistischste und die wahrscheinlichste Schätzung verwendet, um den Erwartungswert und die Standardabweichung der Schätzung zu berechnen (siehe ISTQB®-Lehrplan Foundation Level V4.0, Abschnitt 5.1.4 für weitere Einzelheiten).
- **Verfügbarkeit von Daten:** Einige Schätzverfahren erfordern historische Daten aus früheren oder ähnlichen Projekten, die möglicherweise nicht verfügbar oder unzuverlässig sind. Beispielsweise sind Schätzungen auf der Grundlage von Verhältnissen und Extrapolation auf historische Daten angewiesen, um die Verhältnisse oder Trends für das aktuelle Projekt abzuleiten.
- **Verfügbarkeit von Experten:** Einige Schätzverfahren erfordern die Einbeziehung von Experten mit ihrem Wissen und ihrer Erfahrung, um genaue und realistische Schätzungen abzugeben. So stützen sich beispielsweise die Breitband-Delphi-Methode und das Planungspoker auf die Meinungen und Urteile von Experten oder Teammitgliedern.
- **Modellierungskenntnisse:** Einige Schätzverfahren erfordern die Verwendung mathematischer Modelle oder Formeln zur Berechnung der Schätzungen, was bestimmte Kompetenzen und Kenntnisse in der Modellierung voraussetzt. Beispielsweise werden bei der Extrapolation und der Drei-Punkt-Schätzung Formeln verwendet, um den Erwartungswert und die Standardabweichung der Schätzung abzuleiten.
- **Zeitliche Beschränkungen:** Einige Schätzverfahren erfordern mehr Zeit und Aufwand als andere, was sich auf ihre Durchführbarkeit und Eignung auswirken kann. Planungspoker ist z. B. leicht durchzuführen, während eine Extrapolation schwieriger sein kann.

Dies zeigt, dass die Auswahlkriterien für die richtigen Testschätzungsverfahren in hohem Maße vom Kontext des Testens abhängen (z. B. SDLC, Stakeholder, Teststufen und

Testarten, die im Projekt verwendet werden) (siehe Abschnitt 1.2 „Der Kontext des Testens“). Testmanager müssen in der Lage sein, Verfahren zur Testschätzung zu koordinieren und anzuwenden (z. B. mit verschiedenen SDLC-Modellen in einem Projekt über verschiedene Standorte hinweg).

Um beispielsweise das geeignete Schätzverfahren auszuwählen, muss zunächst die Komplexität der Aufgabe bestimmt werden. Ist die Komplexität gering, könnten metrikbasierte Verfahren verwendet werden. Ist sie hoch, könnten expertenbasierte Verfahren genutzt werden. Wird ein sequenzielles Entwicklungsmodell verwendet, könnte das Breitband-Delphi-Verfahren zum Einsatz kommen. Bei Verwendung eines agilen Softwareentwicklungsmodells könnte Planungspoker eingesetzt werden.

2.3 Fehlermanagement

Einführung

Der ISTQB®-Lehrplan Foundation Level V4.0 beschreibt Aktivitäten, die nach Feststellung einer Abweichung der tatsächlichen von den erwarteten Ergebnissen beginnen. Der Lehrplan bezeichnet diese Aktivitäten als Fehlermanagement. Andere Standards verwenden den Begriff "Incident Management" (ISO/IEC/IEEE 29119-3 Standard) oder "Management von Anomalien" (TMAP), um die Tatsache zu betonen, dass zu Beginn des Prozesses möglicherweise nicht bekannt ist, ob die Diskrepanz durch einen Fehlerzustand in einem Arbeitsprodukt oder durch etwas anderes verursacht wird (z. B. eine Fehlerwirkung der Testautomatisierung oder ein falsches Verständnis der Anforderungen durch den Tester). Das Fehlermanagement und das zur Verwaltung von Fehlern verwendete Werkzeug sind für die Tester und die anderen an der Softwareentwicklung beteiligten Teammitglieder von entscheidender Bedeutung. Die Informationen aus einem effektiven Fehlermanagementprozess ermöglichen es dem Testteam und anderen Projektbeteiligten, einen Einblick in den Zustand eines Projekts während des gesamten SDLC zu erhalten. Das Fehlermanagement ist für die Entscheidung, welche Fehlerzustände mit welcher Priorität behoben werden sollen, ausschlaggebend. Auf diese Weise wird sichergestellt, dass die Arbeit an den richtigen Fehlerzuständen geleistet wird. Das Sammeln und Analysieren fehlerbezogener Daten im Laufe der Zeit kann dazu beitragen, Bereiche mit Verbesserungspotenzial sowohl für das Testen als auch für andere Prozesse innerhalb des SDLC zu identifizieren (z. B. bessere Fehlerprävention durch Verbesserungen in der Architektur und im technischen Design).

Der Testmanager und die Tester (bzw. das gesamte agile Team in der agilen Softwareentwicklung) müssen nicht nur den gesamten Fehlerlebenszyklus verstehen und wissen, wie dieser sowohl für den Softwareentwicklungs- als auch den Testprozess zur Überwachung und Steuerung eingesetzt wird, sondern auch, welche Daten unbedingt erfasst werden müssen. Der Testmanager muss ein Verfechter der korrekten Anwendung des Fehlermanagements und des ausgewählten Fehlermanagementwerkzeugs sein.

2.3.1 Fehlerlebenszyklus

Jede Phase des SDLC sollte Aktivitäten zur Erkennung und Beseitigung potenzieller Fehlerzustände umfassen. So können beispielsweise statische Testverfahren (d. h. Reviews und statische Analysen) auf Designspezifikationen, Anforderungsspezifikationen und Codes angewendet werden, bevor diese Arbeitsprodukte in die nachfolgenden Aktivitäten einfließen. Je früher Fehlerzustände erkannt und beseitigt werden, desto geringer sind die Qualitätskosten für das Produkt. Die Qualitätskosten werden minimiert, wenn jeder Fehlerzustand in derselben Phase beseitigt wird, in der er entstanden ist (d. h., wenn der Softwareprozess eine optimale Fehlereindämmung innerhalb der Phase erreicht).

Beim statischen Testen wird nach Fehlerzuständen gesucht ohne das Testelement auszuführen. Dagegen wird beim dynamischen Testen, bei dem das Testelement ausgeführt wird, das Vorhandensein eines Fehlerzustandes erst dann aufgedeckt, wenn er eine Fehlerwirkung verursacht, die zu einer Diskrepanz zwischen den tatsächlichen Ergebnissen und den erwarteten Ergebnissen eines Tests, d. h. einer Anomalie, führt. In manchen Fällen erscheint ein falsch negatives Ergebnis, was bedeutet, dass der Tester die Anomalie nicht bemerkt. Wenn eine Anomalie beobachtet wird, sollte eine weitere Untersuchung durchgeführt werden. Diese Untersuchung beginnt in der Regel mit dem Erfassen eines Fehlerberichts in Übereinstimmung mit dem definierten Test- und Fehlermanagementprozess. Ein fehlgeschlagener Test führt nicht immer zur Erstellung eines Fehlerberichts (z. B. in der

testgetriebenen Entwicklung, wo Komponententests, in der Regel automatisiert, als eine Form der ausführbaren Designspezifikation verwendet werden). Bis die Entwicklung der Komponente abgeschlossen ist, müssen einige oder alle Tests fehlschlagen. Daher ist das Ergebnis eines solchen Tests nicht unbedingt auf einen Fehlerzustand zurückzuführen und wird in der Regel nicht über einen Fehlerbericht nachverfolgt.

Ein Fehlerbericht durchläuft einen Fehlerworkflow, bei dem verschiedene Fehlerzustände gesetzt werden. In den meisten dieser Zustände ist eine Person für den Fehlerbericht und die Durchführung einer Aufgabe verantwortlich (z. B. Analyse, Fehlerbeseitigung oder Fehlernachtest). Das folgende Diagramm stellt einen einfachen Fehlerworkflow dar:

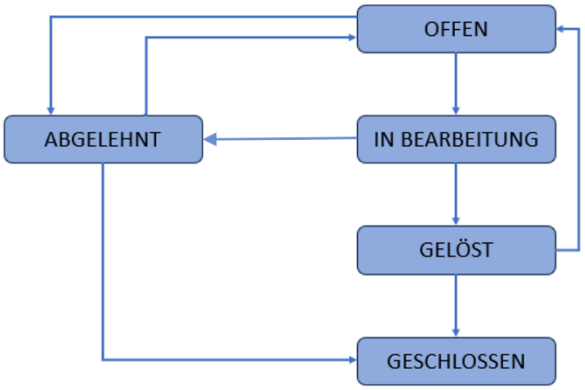

Abbildung 2: Ein einfacher Fehlerworkflow

Ein einfacher Fehlerworkflow kann die folgenden Fehlerzustände umfassen:

- OFFEN (kann auch NEU genannt werden): Der Ausgangszustand, solange der Fehlerbericht erstellt wird.
- IN BEARBEITUNG: Das Team arbeitet an Fehlerbericht, Analyse und/oder Behebung.
- ABGELEHNT (ZURÜCKGEWIESEN): Ein Fehlerbericht wird von der Person abgelehnt, die ihn bearbeitet hat (in der Regel ein Entwickler oder ein Analyst). Es kann viele Gründe für die Ablehnung geben (z. B. ungültige Informationen, falscher Test, doppelter Fehlerzustand). Diese Informationen werden dem Fehlerbericht hinzugefügt.
- GELÖST (kann auch BEHOBEN, BEREIT FÜR NACHTEST genannt werden): Ein Tester führt einen Fehlernachtest durch, wobei er häufig die Schritte zur Reproduktion der Fehlerwirkung aus dem Fehlerbericht entnimmt, um festzustellen, ob der Fehlerzustand durch die Korrektur tatsächlich behoben wurde.
- GESCHLOSSEN: Der Fehlerbericht hat seinen Endzustand erreicht, und es ist keine weitere Arbeit mehr vorgesehen. Der Tester überführt den Fehlerbericht in diesen Zustand entweder nach einem erfolgreichen Fehlernachtest oder um die Ablehnung des Fehlerberichts zu bestätigen.

Ein einfacher Fehlerworkflow wird in vielen Unternehmen eingesetzt und durch die Verwendung anderer Fehlerzustände erweitert, die für einen bestimmten Kontext relevant sind (z. B. WIEDER GEÖFFNET, AKZEPTIERT, ZUR KLÄRUNG oder ZURÜCKGESTELLT).

Der Fehlerworkflow kann sich in verschiedenen Organisationen in Bezug auf die Namen der Fehlerzustände, die Regeln für die Übergänge zwischen den Fehlerzuständen und die für die Aufgaben in bestimmten Fehlerzuständen zuständigen Rollen unterscheiden. Oft ist der Fehlerworkflow in der agilen Softwareentwicklung einfacher als in sequenziellen

Entwicklungsmodellen. Der Fehlerworkflow sollte an den jeweiligen Kontext angepasst werden. Bei der Gestaltung des Fehlerworkflows ist es ratsam, einige Best Practices zu beachten:

- Wenn möglich, sollte der Fehlerworkflow organisationsweit definiert werden, um ein einheitliches Fehlermanagement für alle Projekte zu gewährleisten.
- Doppelte und falsch positive Fehlerzustände sollten durch einen separaten Status oder eine Kombination des ABGELEHNT-Status mit der Auswahl des Ablehnungsgrundes dargestellt werden. Sie können bei weiteren Fehlerzuständen hilfreich sein, um den Testprozess zu verbessern.
- Es wird empfohlen, nur einen Endstatus zu verwenden (z. B. GESCHLOSSEN). Der Übergang zu diesem Zustand erfordert oft die Auswahl eines Grundes für die Schließung, was für Prozess-Assessment und Prozessverbesserungsaktivitäten nützlich ist.
- Die Namen der Zustände im Fehlerworkflow sollten die gleichen Namen haben wie die entsprechenden Zustände für andere Entitäten (z. B. User Stories und Testaufgaben), um die Arbeit mit ihnen zu vereinfachen.
- Aufeinanderfolgende Fehlerzustände sollten zu verschiedenen verantwortlichen Rollen gehören. Wenn zwei oder mehr aufeinanderfolgende Zustände zur gleichen verantwortlichen Rolle gehören, sollte es einen guten Grund dafür geben (z. B. um die in einem Fehlerzustand verbrachte Zeit zu messen).
- Jeder Fehlerzustand, außer dem Endzustand, sollte mehr als einen ausgehenden Übergang haben, damit die verantwortliche Rolle eine Entscheidung über den nächsten Schritt treffen kann. Ausnahmen von dieser Regel sollten begründet werden (z. B., um den Zeitaufwand für eine bestimmte Aktivität zu überwachen).
- Die Menge der Attribute, die bei der Durchführung eines Zustandsübergangs eingegeben werden müssen, sollte auf diejenigen beschränkt werden, die für das Fehlermanagement von erheblichem Wert sind.

2.3.2 Funktionsübergreifendes Fehlermanagement

Obwohl die Testorganisation und der Testmanager häufig für das gesamte Fehlermanagement und das Fehlermanagementwerkzeug verantwortlich sind, ist in der Regel ein funktionsübergreifendes Team für die Verwaltung der Fehlerzustände in einem bestimmten Projekt zuständig. Diesem Team, das manchmal auch als Fehlermanagement-Ausschuss bezeichnet wird, können der Testmanager, Vertreter der Entwicklung, der Zulieferer, des Projektmanagements, des Produktmanagements oder der Product Owner sowie andere Stakeholder angehören, die ein Interesse an der zu testenden Software haben.

Wenn Anomalien entdeckt und in das Fehlermanagementwerkzeug eingegeben werden, sollte der Fehlermanagement-Ausschuss entscheiden, ob jeder Fehlerbericht einen gültigen Fehlerzustand darstellt und ob er behoben (und von welcher Partei, falls mehrere Entwicklungsteams an der Lieferung beteiligt sind), abgelehnt oder zurückgestellt werden sollte. Bei dieser Entscheidung muss der Fehlermanagement-Ausschuss die Vorteile, Risiken und Kosten abwägen, die mit der Behebung des Fehlerzustands verbunden sind. Es ist von Vorteil, die Überlegungen in einem Meeting (oft Triage Meeting genannt) zu besprechen. Wenn der Fehlerzustand behoben werden soll, sollte das Team die Priorität der Behebung des Fehlerzustands im Vergleich zu anderen Aufgaben festlegen. Der Testmanager und das Testteam können bezüglich der relativen Wichtigkeit eines Fehlerzustands konsultiert werden und sollten die verfügbaren objektiven Informationen zur Verfügung stellen.

Bei sehr großen Projekten kann die Ernennung eines Vollzeit-Fehlermanagers durch den Aufwand gerechtfertigt sein, der für die Vor- und Nachbereitung der in den Sitzungen des Fehlermanagement-Ausschusses getroffenen Entscheidungen erforderlich ist, zumindest in den SDLC-Phasen, in denen das Testen am intensivsten ist. In anderen Situationen können sich mehrere große Projekte einen Fehlermanager teilen.

Ein Fehlermanagementwerkzeug sollte nicht als Ersatz für eine gute Kommunikation verwendet werden und ein Fehlermanagement-Ausschuss sollte auch nicht als Ersatz für die Effektivität eines guten Fehlermanagementwerkzeugs dienen. Kommunikation, angemessene Unterstützung durch das Werkzeug, ein gut definierter Fehlerworkflow (einschließlich der Eigenschaften des Fehlerberichts) und ein engagiertes Fehlermanagementteam sind für ein effektives und effizientes Fehlermanagement notwendig.

2.3.3 Besonderheiten des Fehlermanagements in agilen Teams

Das Fehlermanagement in Unternehmen, die nach einem agilen SDLC arbeiten, ist oft leichtgewichtig und/oder weniger formell als in sequenziellen Entwicklungsmodellen. Wenn agile Teams zusammenarbeiten oder über gut etablierte Kommunikationsmittel verfügen, werden Informationen über einen Fehlerzustand oder eine Fehlerwirkung häufig zwischen Testern, Kundenvertretern und Entwicklern ausgetauscht, ohne dass ein formeller Fehlerbericht erstellt wird. Fehlerberichte sollten jedoch erstellt werden für:

- Fehlerzustände, die andere laufende Sprint-Aktivitäten (d. h. Entwicklung, Testen oder andere) blockieren und nicht sofort innerhalb des agilen Teams behoben werden konnten.
- Fehlerzustände, die nicht innerhalb der gleichen Iteration behoben werden können. Einige agile Teams haben es sich zur Regel gemacht, einen Fehlerbericht zu erstellen, wenn der Fehlerzustand nicht an dem Tag behoben werden kann, an dem der Fehler gefunden wird.
- Fehlerzustände, die von oder in Zusammenarbeit mit anderen Teams innerhalb der Organisation behoben werden müssen.
- Fehlerzustände, die von einem Lieferanten behoben werden müssen.
- Fehlerzustände, für die ein Fehlerbericht ausdrücklich angefordert wird (z. B., wenn ein Entwickler nicht sofort an einer Korrektur arbeiten kann).

Es ist gängige Praxis, Fehlerzustände, die nicht innerhalb derselben Iteration behoben werden können, dem Product Backlog hinzuzufügen, damit sie mit anderen Fehlern und User Stories für eine spätere Iteration priorisiert werden können.

Obwohl die Grundlagen des Fehlermanagements in der Teststrategie eines Unternehmens festgelegt werden sollten, können viele Aspekte, wie der Grad der Formalität, die Auslöser für die Erstellung eines Fehlerberichts und die zu erfassenden Fehlerzustände von den Mitgliedern des agilen Teams vereinbart werden. Im Allgemeinen sollte der Grad der Formalität des Fehlermanagements und der Ansatz zur Erstellung von Fehlerberichten folgendes berücksichtigen:

- Teammitglieder, die gemeinsam Vor-Ort arbeiten
- Teammitglieder, die in unterschiedlichen Zeitzonen zusammenarbeiten
- Die Anzahl der Teams, die bei der Produktentwicklung zusammenarbeiten
- Teamreife
- Teamgröße

- Produktrisiken
- Regulatorische, vertragliche oder andere Anforderungen (falls und wo zutreffend)

Die endgültige Entscheidung des agilen Teams über die Einzelheiten des Fehlermanagements sollte immer dokumentiert werden (z. B. mit Richtlinien in einem Wissensmanagementwerkzeug).

2.3.4 Herausforderungen beim Fehlermanagement in der hybriden Softwareentwicklung

In der Praxis arbeiten oft mehrere Teams gemeinsam an der Bereitstellung des Systems oder des Systems von Systemen. Beispiele hierfür sind die hybride Softwareentwicklung, wenn ein Kunde die agile Softwareentwicklung einsetzt und einer seiner Zulieferer ein sequenzielles Entwicklungsmodell verwendet oder wenn ein Unternehmen, das ein sequenzielles Entwicklungsmodell verwendet, die Lieferung eines Teilsystems von einem Team verlangt, das eine agile Softwareentwicklung einsetzt. Eine solche Multi-Team-Umgebung bringt verschiedene Herausforderungen mit sich:

- **Für das Fehlermanagement zu verwendende Werkzeuge:** In einem idealen Szenario verwenden alle Teams ein einziges Werkzeug für das Fehlermanagement. In der Praxis ist es üblich, dass jedes Team ein anderes Werkzeug für das Fehlermanagement einsetzt, insbesondere wenn mehrere Zuliefererteams an der Projektlieferung beteiligt sind. In solchen Fällen ist es sinnvoll, eine Synchronisation zwischen den Werkzeugen für das Fehlermanagement (vorzugsweise automatisch) herzustellen.
- **Priorisierung von Fehlerzuständen:** Product Owner sollten in die Besprechungen zum Fehlermanagement einbezogen werden und sich aktiv um Informationen über die mit den Fehlern verbundenen Konsequenzen und Risiken bemühen. Meetings zum Fehlermanagement sollten bei der agilen Softwareentwicklung häufiger stattfinden als bei sequenziellen Entwicklungsmodellen, um mit der schnelleren Bereitstellung von Produktinkrementen durch das agile Team Schritt zu halten. Diese Besprechungen können bei agilen Teams jedoch auch kürzer ausfallen. Manchmal ist es von Vorteil, wenn eine kleinere Gruppe von am Fehlermanagement beteiligten Stakeholdern das letzte Wort über die Priorisierung von Fehlern hat.
- **Ausrichtung und Transparenz des Testkonzepts für Neuentwicklungen und Fehlerzustände:** Die Arbeit aller Teams sollte sich an demselben Projektplan orientieren, unabhängig davon, ob sie agile oder sequenzielle Entwicklungsmodelle verwenden. Alle Ergebnisse, einschließlich Fehlerzustände, sollten an diesem Projektplan ausgerichtet sein. Eine bessere Abstimmung kann durch die aktive Teilnahme der Mitglieder aller Teams am Planungsprozess erreicht werden (z. B. Teilnahme von Teams des sequenziellen Entwicklungsmodells an Meetings zur agilen Softwareentwicklung, in denen Fehlerzustände diskutiert und priorisiert werden). Die Transparenz von Entwicklungsplänen kann verbessert werden, indem sie zwischen den Teams ausgetauscht werden (z. B. über Dashboards oder über das Product Backlog).

2.3.5 Informationen im Fehler-Lebenszyklus

Die Informationen in einem Fehlerbericht sollten folgenden Zwecken genügen:

- Management des Fehlerberichts durch den gesamten Fehlerlebenszyklus
- Bewertung des Gesamtprojektstatus, insbesondere im Hinblick auf die Produktqualität und des Testfortschritts

- Bewertung des Status einer Produktinkrements in Bezug auf die Qualität des Produkts
- Bewertung der Prozessfähigkeit

Die für das Fehlermanagement und den Projektstatus benötigten Informationen können variieren, je nachdem, wann der Fehlerzustand im SDLC entdeckt wird. Darüber hinaus können Fehlerberichte, die sich auf nicht-funktionale Qualitätsmerkmale beziehen, zusätzliche Informationen erfordern (z. B. Lastbedingungen für Probleme mit der Performanz). Die gesammelten Schlüsselinformationen sollten jedoch über den gesamten SDLC und idealerweise über alle Projekte in einem Unternehmen hinweg konsistent sein, um einen aussagekräftigen Vergleich der Fehlerzustände im gesamten Projekt und über alle Projekte hinweg zu ermöglichen.

In einem Fehlerbericht können viele Datenelemente gesammelt werden. Der Testmanager sollte entscheiden, welche Informationen für ein effektives Fehlermanagement in einem bestimmten Projektkontext geeignet sind. Da jedes zusätzliche Attribut den Zeitaufwand für die Fehlerberichterstattung erhöht und bei der Person, die den Fehlerbericht eingibt, für Verwirrung sorgen kann, ist es ratsam, nur Daten zu erfassen, die für das Fehlermanagement im gegebenen Kontext erforderlich sind und/oder zur Prozessverbesserung verwendet werden.

Zur Verwaltung des Fehlerberichts sind in den meisten Umgebungen die folgenden Punkte obligatorisch:

- Ein Fehlertitel mit einer kurzen Zusammenfassung der Anomalie
- Eine detaillierte Beschreibung der Anomalie, vorzugsweise mit Schritten zur Reproduktion der Fehlerwirkung
- Fehlerschweregrad der Auswirkungen auf das System unter Test (SuT) und/oder die Stakeholder des Produkts
- Priorität der Behebung der Anomalie

Zusätzliche wichtige Detailinformationen werden oft vom Fehlermanagementwerkzeug erstellt:

- Eindeutige Kennung des Fehlerzustands
- Datum/Uhrzeit der Erstellung des Fehlerberichts
- Name der Person, die die Anomalie entdeckt und/oder gemeldet hat
- Projekt und SDLC-Phase, in der die Anomalie entdeckt wurde
- Aktueller Stand des Fehlerberichts
- Aktueller Eigentümer (d. h. die Person, die derzeit mit der Bearbeitung des Fehlerzustands betraut ist)
- Änderungshistorie, wie z. B. die Abfolge der Maßnahmen, einschließlich Datum/Uhrzeit, die von den Mitgliedern des Projektteams ergriffen wurden, um den Fehlerzustand zu isolieren, zu reparieren und als behoben zu bestätigen
- Referenzen (z. B. zum Testfall, zu verbundenen Fehlerzuständen)

Je nach Kontext können auch weitere Informationen (z. B. Verfolgbarkeit) in einem Fehlerbericht gesammelt werden (siehe ISO/IEC/IEEE 29119-3 für weitere Informationen). Die folgenden Aufzählungspunkte gruppieren die Informationen nach dem beabsichtigten Zweck:

- **Zur Unterstützung bei der Behebung des Fehlerzustands:** Das Subsystem oder die Komponente, in der der Fehler liegt, das spezifische Testelement und seine Versionsnummer, in dem die Anomalie beobachtet wurde, oder die Testumgebung, in der der Fehlerzustand beobachtet wurde
- **Um den Gesamtstatus des Projekts zu bewerten:** Informationen zur Überwachung des Fortschritts (z. B. Risiken, Kosten, Chancen und Vorteile, die mit der Behebung oder Nichtbehebung des Fehlerzustands verbunden sind, eine Beschreibung aller verfügbaren Umgehungslösungen oder von den Fehlern betroffene Anforderungen)
- **Zur Beurteilung des Status eines Produktinkrements in Bezug auf die Produktqualität:** Die Art des Fehlers (in der Regel entsprechend einer Fehlertaxonomie), das Arbeitsprodukt, in dem der Fehler aufgetreten ist, oder das Qualitätsmerkmal, das von dem Fehler betroffen ist
- **Zur Bewertung der Prozessfähigkeit:** Informationen zur Überwachung der Effektivität und Effizienz der Entwicklungsprozesse (z. B. die SDLC-Phase der Einführung, Erkennung und Beseitigung des Fehlerzustands oder der Grundursache des Fehlers)

2.3.6 Definition von Maßnahmen zur Prozessverbesserung anhand von Informationen aus Fehlerberichten

Wie in Abschnitt 2.3.5 „Informationen zum Fehlerzustand“ erläutert, können Fehlerberichte für die Überwachung des Projektstatus und die Berichterstattung nützlich sein. Während die Auswirkungen von Metriken auf den Testprozess in erster Linie im ISTQB® Expert Level Test Management Syllabus behandelt werden, sollten sich verantwortliche Leiter auf der Stufe des fortgeschrittenen Testmanagements (Advanced Level) darüber im Klaren sein, welche Bedeutung Fehlerberichte für die Beurteilung der Fähigkeit der Softwareentwicklungs- und Testprozesse haben.

Zusätzlich zu den Informationen über den Testfortschritt, die in diesem Lehrplan in Abschnitt 2.1.2 „Überwachung, Steuerung und Abschluss“ und in Abschnitt 2.1.3 „Testberichterstattung“ erwähnt werden, sollten die Fehlerzustände die Initiativen zur Prozessverbesserung unterstützen, die in Retrospektiven diskutiert werden. Beispiele hierfür sind:

- Verwendung von Informationen über die Phasen der Einführung, Erkennung und Beseitigung von Fehlern zur Bewertung der Fehlereindämmung innerhalb der Phase und/oder zur Durchführung einer Qualitätskostenanalyse mit dem Ziel, mögliche Wege vorzuschlagen, um die Effektivität der Fehlererkennung in jeder Phase zu verbessern und die mit Fehlern verbundenen Kosten zu minimieren.
- Verwendung von Informationen über die Phasen, in denen Fehlern entstehen zur Analyse im Hinblick darauf, in welcher Phase die meisten Fehler entstehen, um gezielte Verbesserungen zur Fehlerprävention zu ermöglichen.
- Informationen zur Ermittlung der Grundursache von Fehlerzuständen, um Prozessverbesserungen zu ermöglichen, die die Gesamtzahl der Fehlerzustände reduzieren.
- Verwendung von Informationen über den Fehlerzustand zur Analyse der Anhäufung von Fehlerzuständen zum besseren Verständnis technischer Risiken (für risikobasierte Tests) und um das Refactoring problematischer Komponenten zu ermöglichen.
- Verwendung von Informationen über wieder geöffnete Fehlerzustände, um die Qualität von Debugging-Implementierungen zu bewerten.

- Verwendung von Informationen über doppelte und zurückgewiesene Fehlerzustände, um die Qualität der Erstellung von Fehlerberichten zu bewerten.
- Ermöglichung von Prozessverbesserungen, die die Gesamtzahl der Fehlerzustände reduzieren, indem proaktive Maßnahmen eingeführt werden, um Fehlerzustände im Vorfeld zu vermeiden.

Die Verwendung von Metriken zur Bewertung der Effektivität und Effizienz des Testprozesses wird im ISTQB® Expert Level Test Management Syllabus behandelt.

In manchen Fällen entscheiden sich Teams dafür, Fehlerzustände, die in einigen oder allen Phasen des SDLC gefunden werden, nicht zu verfolgen. Dies geschieht zwar oft zu Gunsten der Effizienz und um den Prozessaufwand zu reduzieren, verringert aber den Einblick in die Prozessfähigkeiten der Softwareentwicklung und des Testens erheblich. Dies erschwert die Durchführung der oben vorgeschlagenen Verbesserungen, da es an zuverlässigen Daten zur Unterstützung fehlt.

3 Das Team managen – 225 Minuten

Schlüsselbegriffe

externe Fehlerkosten, Fehlerpräventionskosten, Fehlerwirkung, Fehlerzustand, interne Fehlerkosten, Qualitätskosten, Überprüfungskosten

Lernziele für Kapitel 3: **Der Lernende kann ...**

3.1 Das Testteam

TM-3.1.1 (K2) … Beispiele für typische Kompetenzen nennen, die von Mitgliedern eines Testteams in den vier Kompetenzbereichen benötigt werden.

TM-3.1.2 (K4) … einen gegebenen Projektkontext analysieren, um daraus die geforderten Kompetenzen der Mitglieder des Testteams abzuleiten.

TM-3.1.3 (K2) … typische Verfahren zur Bewertung der Kompetenz von Mitgliedern eines Testteams erklären.

TM-3.1.4 (K2) … die typischen Ansätze zur Entwicklung der Kompetenzen von Mitgliedern eines Testteams voneinander unterscheiden.

TM-3.1.5 (K2) … die für die Leitung eines Testteams erforderlichen Managementkompetenzen erklären.

TM-3.1.6 (K2) … Beispiele für motivierende Faktoren und Hygienefaktoren für Mitglieder eines Testteams erklären.

3.2 Stakeholder-Beziehungen

TM-3.2.1 (K2) … Beispiele für jede der vier Kategorien nennen, aus denen sich die Qualitätskosten ermitteln lassen.

TM-3.2.2 (K3) … eine Berechnung des Kosten-Nutzen-Verhältnisses zur Abschätzung des Mehrwerts des Testens durch die Stakeholder vornehmen.

3.1 Das Testteam

Einführung

Jedes Team, das an Testaufgaben arbeitet, setzt sich aus Mitgliedern mit unterschiedlichen Kompetenzen zusammen. In einigen Unternehmen organisieren sich diese Teams selbst, in anderen rekrutiert und entwickelt das Testmanagement diese Teams. Für alle Teams gilt, dass die richtige Mischung von Kompetenzen[1] ein entscheidender Faktor für die erfolgreiche Ausübung von Testaufgaben ist.

Die Anforderungen an die Mitglieder eines Testteams können sich im Laufe der Zeit ändern. Es ist wichtig, die richtigen Personen für ein Testteam auszuwählen und ihnen angemessene Schulungs- und Entwicklungsmöglichkeiten zu bieten. Darüber hinaus können Personen außerhalb eines Testteams zusätzliche spezifische Kompetenzen einbringen.

Dieser Abschnitt befasst sich mit dem grundlegenden Prozess der Analyse und Entwicklung der Kompetenzen, die für die Mitglieder eines Testteams erforderlich sind, sowie mit den Kompetenzen, die für die Leitung oder das Coaching eines Testteams benötigt werden. Dazu gehört auch die Kenntnis der Faktoren, die die Mitglieder eines Testteams motivieren oder demotivieren, sowie anderer Faktoren, die eine erfolgreiche Teamarbeit gewährleisten.

Jeder Einzelne verfügt bereits über Kompetenzen und kann diese auf verschiedene Weise weiterentwickeln, z. B. durch Berufserfahrung, Ausbildung und Schulung. Das ideale Testteam verfügt über alle für die Testaufgaben erforderlichen Kompetenzen oder ist nur für Aufgaben verantwortlich, für die es die erforderlichen Kompetenzen besitzt. Um erfolgreich zu sein, benötigt ein Testteam verschiedene Kompetenzen auf unterschiedlichen Ebenen. Je nach Projektkontext sind einige Kompetenzen wichtiger oder notwendiger als andere. Für bestimmte Testaufgaben, die die Kompetenzen des Testteams übersteigen, kann es sinnvoll sein, externe Experten hinzuzuziehen.

3.1.1 Typische Kompetenzen in den vier Kompetenzbereichen

Die Kompetenzen einer Person können in vier Kompetenzbereiche unterteilt werden (Sonntag & Schmidt-Rathjens, 2005) (Erpenbeck & von Rosenstiel, 2017)[2]:

- **Fachkompetenz:** Besteht aus Kompetenzen zur Ausübung fachlicher Aufgaben. Beispiele sind Fähigkeiten in Testverfahren, technologisches und fachliches Wissen im Anwendungsbereich sowie Fähigkeiten im Projektmanagement.
- **Methodenkompetenz:** Umfasst allgemeine Kompetenzen, die eine Person in einem Bereich selbstständig anwenden kann und die es ihr ermöglichen, komplexe oder neue Aufgaben selbstständig zu erledigen. Beispiele sind analytische und konzeptionelle Fähigkeit sowie Urteilsvermögen.
- **Sozialkompetenz:** Umfasst Kommunikations-, Kooperations- und Konfliktkompetenzen in intra- und interkulturellen Kontexten. Sie ermöglichen es, mit anderen in Beziehung zu treten, um in einer bestimmten Situation angemessen zu handeln und individuelle und gemeinsame Ziele zu erreichen. Beispiele hierfür sind Kommunikationsfähigkeit,

[1] Der Begriff „Kompetenz“ wird als Oberbegriff verwendet für das „Wissen“ über etwas, die „Fertigkeit“, dieses Wissen praktisch anzuwenden, und die „Fähigkeit“, in verschiedenen Situationen damit erfolgreich zu handeln.

[2] Die hier verwendeten vier Kompetenzbereiche basieren auf dem in diesen Referenzen beschriebenen Modell, das weit verbreitet ist. In der Literatur werden auch andere Modelle beschrieben, die Kompetenzen anders gruppieren. Diese sind nicht Teil dieses Lehrplans.

Konfliktlösungsfähigkeit, Teamfähigkeit, Anpassungsfähigkeit und Durchsetzungsvermögen.

- **Selbstkompetenz** (Individualkompetenz): Umfasst die Fähigkeit und Bereitschaft, sich selbst und das eigene Talent weiterzuentwickeln, die Motivation und die Leistungsbereitschaft, sich zu entfalten, sowie bestimmte Einstellungen und die individuelle Persönlichkeit zu entwickeln. Beispiele sind Selbstmanagement, Eigenverantwortung, Kritikfähigkeit, Zuverlässigkeit, Belastbarkeit, Selbstvertrauen, Disziplin, Offenheit für Veränderungen, Hilfsbereitschaft, Lernbereitschaft und die Fähigkeit, zu delegieren.

Alle Kompetenzbereiche sind wichtig für den Erfolg eines Testteams. Da Methoden-, Sozial- und Selbstkompetenz nicht spezifisch für das Testen sind, konzentriert sich das ISTQB® auf die Entwicklung von Fachkompetenz. Dazu gehören Kompetenzen im Management von Testaufgaben, in der Analyse der Testbasis, im Entwurf von Tests, in der Identifikation und Analyse von Risiken sowie in der Entwicklung, Konfiguration und Pflege von Testdaten, Testumgebungen und Testskripten.

3.1.2 Analyse der erforderlichen Kompetenzen der Testteammitglieder

Die Besetzung mit Personal ist eine Aktivität im Rahmen der Testplanung. Sie umfasst die Bestimmung der Rollen und Kompetenzen der Mitarbeiter, die für die Ausübung des Testens im Rahmen der Teststrategie erforderlich sind. Eine detaillierte Kontextanalyse ist erforderlich, um die Kompetenzanforderungen für ein Projekt zu bestimmen.

Fach- und Methodenkompetenz

Beim Testen liegt der Schwerpunkt auf den Testfähigkeiten, die für die Testaufgaben erforderlich sind. Nachfolgend sind einige Beispiele aufgeführt:

- Testplanung erfordert konzeptionelle Fähigkeiten zur Entwicklung einer Teststrategie.
- Testüberwachung und Teststeuerung erfordern Projektmanagementfähigkeiten für die Verwaltung aller Testaufgaben.
- Testanalyse erfordert analytische Fähigkeiten zur Analyse der Testbasis und der Produktrisiken.
- Testentwurf erfordert Kenntnisse von Testverfahren für den Entwurf von Testfällen und konzeptionelles Wissen für den Entwurf von Testumgebungen.
- Testrealisierung erfordert Urteilsvermögen bei der Vorbereitung der Testmittel wie z. B. die Auswahl relevanter Tests und technisches Fachwissen bei der Programmierung von Testskripten und der Einrichtung von Testumgebungen.
- Testdurchführung erfordert technisches Fachwissen, um automatisierte Tests auszuführen, explorative Tests durchzuführen und Testergebnisse auszuwerten.
- Testabschluss erfordert die Fähigkeit, Projektergebnisse zu kommunizieren und persönliche Verantwortung für getroffene Entscheidungen zu übernehmen.

Unterschiedliche Testarten und Teststufen erfordern unterschiedliche Kompetenzen (z. B. Fachwissen im Anwendungsbereich, um die funktionale Eignung eines Systems zu beurteilen, oder technisches Fachwissen, um die Wartbarkeit des Codes zu beurteilen).

Darüber hinaus liefert der Projektkontext wertvolle Informationen über die erforderlichen Kompetenzen:

- Die Systemdomäne erfordert Fachwissen im Anwendungsbereich, z. B. in der Informationstechnologie, der Automobilindustrie oder der Pharmaindustrie.
- Die Software- und Systemarchitektur und die im Projekt verwendeten Technologien erfordern z. B. technisches Wissen über Programmiersprachen, Schnittstellentechnologien oder Sicherheitslücken.
- Der SDLC erfordert z. B. Kenntnisse über Teststufen, Testrollen und spezifische Testverfahren.

Sozialkompetenz

Im Kontext des Testens befähigt die Sozialkompetenz die Mitglieder des Testteams, sich im Umgang mit anderen Teammitgliedern angemessen zu verhalten und die Testziele zu erreichen. Sie umfasst insbesondere Kommunikations-, Kooperations- und Konfliktlösungsfähigkeiten (z. B. konstruktiver Umgang mit suboptimalen Testbedingungen oder bei der Meldung von Fehlern an die Entwickler).

Softwareentwicklung und Softwaretests werden in der Regel von verschiedenen Mitgliedern (unterschiedlicher) Gruppen durchgeführt, die ihre Aufgaben durch Kommunikation koordinieren. Kommunikationsfähigkeit, Teamfähigkeit und die Fähigkeit, Konflikte zu lösen, sind Voraussetzungen für den Projekterfolg. Die Anforderungen an soziale Kompetenzen können jedoch je nach Projektkontext variieren. Beispielsweise kann agile Softwareentwicklung höhere Anforderungen an die Sozialkompetenz stellen als dokumentenzentrierte, sequenzielle Entwicklungsmodelle oder Offsite-Testen (z. B. Testen durchgeführt von externen Dienstleistern an einem externen Standort).

Selbstkompetenz

Die Effektivität und Effizienz der Testteammitglieder hängen auch von ihrer Kompetenz und Bereitschaft ab, sich selbst, ihre Fähigkeiten und ihre Einstellungen weiterzuentwickeln. So kann die Arbeit in einem selbstorganisierten agilen Team von allen Teammitgliedern ein höheres Maß an Selbstmanagement und Disziplin erfordern, während in einem hierarchischen Testteam das Testmanagement beispielsweise in der Lage sein muss, Arbeit zu delegieren. Ein hohes Maß an Zuverlässigkeit und Belastbarkeit wird häufig vorausgesetzt, insbesondere bei zeitkritischen Projekten. Darüber hinaus sind Hilfsbereitschaft, Lernbereitschaft und Offenheit für Veränderungen in allen Software Development Life Cycle (SDLC)-Modellen wichtig.

3.1.3 Bewertung der Kompetenzen der Testteammitglieder

In vielen Fällen werden die Testteams aus dem vorhandenen Personal gebildet. Um die Kompetenzen der Teammitglieder und den Bedarf an persönlicher Entwicklung zu verstehen, muss das Testmanagement die vorhandenen Kompetenzen des Testteams bewerten und diese mit den erforderlichen Kompetenzen vergleichen, die in einer Kompetenzmatrix dokumentiert werden können.

Es gibt Modelle, die Teams und Teammitgliedern helfen, effektiver zu arbeiten (z. B. Meredith Belbins "Team Roles"). Nach Belbin (Belbin, 2010) arbeiten Teams dann effektiv, wenn sie sich aus unterschiedlichen Persönlichkeits- und Rollentypen zusammensetzen. Diese Modelle helfen Teams herauszufinden, welche Kompetenzen sie haben und welche ihnen möglicherweise fehlen.

Die Fach- und Methodenkompetenz der Mitglieder des Testteams kann durch Demonstration typischer Testaufgaben beurteilt werden:

- Entwurf einer Teststrategie und Diskussion des Feedbacks mit Kollegen.
- Review der Testbasis und Kommunikation der Ergebnisse, was auch die Analyse und Kommunikationsfähigkeit aufzeigen kann.
- Festlegung von Testverfahren zur Erreichung spezifischer Testziele für einen bestimmten Projektkontext.
- Angemessene Anwendung unterschiedlicher Testverfahren.
- Erstellung eines abschließenden Testberichts mit einer Bewertung der Testergebnisse.

Darüber hinaus können Kompetenzen durch externe Nachweise, Zertifizierungen, Berufserfahrungen und Abschlüsse bewertet werden.

Insbesondere in der agilen Softwareentwicklung identifizieren Teams die erforderlichen Kompetenzen, indem sie regelmäßig an Retrospektiven teilnehmen und Feedback erhalten. Erfahrene Coaches oder Mentoren unterstützen sie dabei, ihre Kompetenzen weiterzuentwickeln und Wissenslücken zu erkennen und zu schließen.

3.1.4 Entwicklung der Kompetenzen der Testteammitglieder

Ein Testteam verfügt zu Beginn eines Projekts möglicherweise nicht über alle erforderlichen Kompetenzen. Auch wenn es keine perfekte Besetzung gibt, kann ein starkes Team die Stärken und Schwächen einzelner Personen ausgleichen.

Das Testmanagement oder das Testteam kann den Entwicklungsbedarf ermitteln, indem es die erforderlichen Kompetenzen mit den vorhandenen Kompetenzen in einer Kompetenzmatrix vergleicht. Auf dieser Basis können Ansätze zur Kompetenzentwicklung definiert werden:

- Training und Ausbildung vermitteln vorgegebene Kenntnisse und Praktiken, normalerweise in einem (virtuellen) Klassenzimmer (z. B. durch Entsendung von Mitarbeitern zu einem Trainingskurs, durch interne Schulungen, durch die Entwicklung maßgeschneiderter Trainingskurse oder durch den Einsatz von Live-E-Learning-Kursen).
- Selbststudium ist eine Form des Lernens, bei der der Lernende nicht in einem (virtuellen) Klassenzimmer, sondern für sich lernt (z. B. durch das Lesen von Büchern oder Lehrplänen, das Ansehen von Videos oder eine Recherche im Internet).
- Beim Peer-Learning (kooperatives bzw. gemeinschaftliches Lernen) tauschen Kollegen Wissen, Ideen und Erfahrungen aus und lernen mit- bzw. voneinander.
- Mentoring oder Coaching sind Ansätze, bei denen ein Teammitglied, das neu in einer Rolle ist, individuelle Beratung durch einen Coach erhält oder Wissen, Fähigkeiten und/oder Erfahrung von einem versierten Mentor vermittelt bekommt. Die erfahrene Person fungiert als verfügbare Ressource, die Rat und Unterstützung bietet.
- Weit verbreitet ist auch die Ausbildung am Arbeitsplatz (Training on the Job), bei der Selbststudium, Peer-Learning und Mentoring oder Coaching kombiniert werden.

Nicht alle Ansätze zur Kompetenzentwicklung sind gleichermaßen effektiv und effizient. Selbststudium und Schulungen sind beispielsweise gut geeignet, um Fach- und Methodenkompetenzen zu entwickeln. Zum Beispiel können grundlegende Testkompetenzen durch die Teilnahme an ISTQB®-Schulungen oder im Selbststudium anhand der ISTQB®-

Lehrpläne erworben werden. Für die Entwicklung von Sozial- und Selbstkompetenz empfehlen sich jedoch Ansätze wie Training und Coaching, die häufig erfolgversprechender sind als das Selbststudium. Sozialer Austausch, Feedback und Reflexion gehören zu den wichtigsten Erfolgsfaktoren für die Entwicklung von Sozial- und Selbstkompetenz.

3.1.5 Erforderliche Managementfähigkeiten für die Leitung eines Testteams

Die erfolgreiche Leitung eines Testteams erfordert Managementfähigkeiten. Dazu gehören Fach- und Methodenkompetenz für grundlegende Managementaufgaben (z. B. Planung, Überwachung des Projektfortschritts, Steuerung und Berichterstattung). Für das Testen sind spezifische Kenntnisse und Fähigkeiten im Testmanagement erforderlich (z. B. Kenntnisse über verschiedene Testansätze, die Entwicklung von Teststrategien oder die Anwendung von Testverfahren oder des eingesetzten SDLC).

Ein Testteam zu leiten oder zu coachen bedeutet, angemessen mit anderen Mitgliedern des Testteams umzugehen und die Fähigkeit und Bereitschaft zu haben, sich unter sich ändernden Umständen weiterzuentwickeln. Daher sind Sozial- und Selbstkompetenz wesentliche Erfolgsfaktoren für die Leitung eines Testteams. Dazu gehören Belastbarkeit, die Fähigkeit, zu delegieren, und die Eigenschaft, vom Testteam akzeptiert zu werden. Darüber hinaus gehört dazu die Fähigkeit, die Interessen des Testens im Projekt durchzusetzen, die Vorteile des Testens zu fördern, mit allen Beteiligten zu kommunizieren und Konflikte zu lösen.

Um Mitglieder für ein Testteam zu gewinnen, sind Fähigkeiten zur Analyse der sozialen, Team- und Arbeitsbedingungen erforderlich. Diese Fähigkeiten tragen dazu bei, das Testteam an die Arbeitsbedingungen anzupassen oder, wenn möglich, die Arbeitsbedingungen an das Testteam anzupassen. Darüber hinaus sind Testteams dynamischen Teamentwicklungsprozessen unterworfen und erfordern daher situationsabhängige Kompetenzen (z. B. entsprechend den Phasen des Tuckman-Modells der Kleingruppenentwicklung) (Tuckman, 1965) (Bonebright, 2010):

- Die Bereitschaft, Mitglieder des Testteams beim Eintritt in das Testteam zu unterstützen (Forming).
- Die Fähigkeit, Konflikte innerhalb des Testteams zu lösen (Storming).
- Disziplin und zielorientierte Führung, um vereinbarte Werte und Regeln sicherzustellen (Norming).
- Die Fähigkeit zu delegieren, um dem Testteam persönliche Verantwortung zu übertragen (Performing).
- Die Fähigkeit, mit ausscheidenden Mitgliedern des Testteams wertschätzend und vertrauensvoll umzugehen (Adjourning).

3.1.6 Motivierende und demotivierende Faktoren für das Testteam in bestimmten Situationen

Zufriedene und motivierte Mitglieder des Testteams erhöhen die Produktivität und Leistungsfähigkeit und haben damit einen erheblichen Einfluss auf den Projekterfolg. Wenn dies erreicht ist, findet ein Cross-Training informell statt, die Mitglieder des Testteams können ihr eigenes Arbeitspensum bewältigen und das Testmanagement hat mehr Zeit, sich mit externen Themen zu beschäftigen. Die Zwei-Faktoren-Theorie (Herzberg, Mausner, & Bloch Snyderman, 1993) unterscheidet zwischen Motivatoren und Hygienefaktoren:

Motivatoren werden bewusst wahrgenommen und können zu Wachstum und Zufriedenheit führen. Dazu können folgende gehören:

- Anerkennung und Wertschätzung für die geleistete Arbeit (z. B. Incentives und jedes andere individuelle Ergebnis, das die Mitglieder des Testteams als wertvoll empfinden).
- Mehr Verantwortung und Autonomie (z. B. bei der Definition der Testprozesse im Testteam).
- Interessante, sinnvolle und herausfordernde Aufgaben, die von den Mitgliedern des Testteams als machbar und gleichzeitig erstrebenswert empfunden werden (z. B. die Auswahl und Einführung eines neuen Testautomatisierungswerkzeugs).
- Beruflicher Aufstieg und Weiterentwicklung (z. B. vom Tester zum Testmanager oder Testprozessverantwortlichen).

Die Hygienefaktoren werden in der Regel als selbstverständlich vorausgesetzt. Die Erfüllung dieser Faktoren führt nicht automatisch zu einer höheren Zufriedenheit. Wenn sie aber fehlen, können sie sich demotivierend auf die Mitglieder des Testteams auswirken:

- Angemessene Entlohnung (z. B. marktgerechtes Gehalt, bezahlte Überstunden, gute Sozialleistungen)
- Wertschätzende Personalpolitik und Führungsstil (z. B. Lean Management, realistische Zielvorgaben, Schutz vor externer Erreichbarkeit und Überlastung)
- Angenehme Arbeitsbedingungen (z. B. klare Spezifikationen, ausgereifte Testobjekte, adäquat behobene Fehlerzustände, adäquater Arbeitsplatz, stabile Testumgebung)
- Sicherheit als existenzielles Bedürfnis (z. B. sicherer Arbeitsplatz, Einhaltung von Vereinbarungen)
- Gute zwischenmenschliche Beziehungen (z. B. zu Kollegen, Vorgesetzten)

Das Testmanagement sollte daher kontinuierlich demotivierende Faktoren beseitigen und gleichzeitig motivierende Faktoren schaffen und stärken.

Weitere Informationen siehe (Belbin, 2010) (Marston, 1999) (Kahler, 2008),(Mohsen, 2022).

3.2 Stakeholder-Beziehungen

Einführung

Beim Testmanagement ist es wichtig, das Testen so zu optimieren, dass ein guter Geschäftswert erzielt wird. Übermäßiges Testen kann unangemessene Verzögerungen und Kosten verursachen, die den Nutzen überwiegen, während unzureichendes Testen dazu führen kann, dass den Anwendern ein Produkt von geringer Qualität geliefert wird. Der optimale Ansatz liegt zwischen diesen beiden Extremen. Es liegt in der Verantwortung des Testmanagers, den Beteiligten zu helfen, dieses Gleichgewicht zu finden und den Mehrwert des Testens bei der Erreichung dieses Gleichgewichts zu verstehen. Dabei muss der Testmanager beispielsweise auch die typischen zeitlichen Rahmenbedingungen eines Projekts im Auge behalten.

3.2.1 Qualitätskosten

Die Vorteile des Testens werden durch die Qualitätskosten ausgeglichen. Ein Mittel zur Quantifizierung der Gesamtkosten von qualitätsbezogenen Maßnahmen und von Fehlern wird als Qualitätskosten bezeichnet. Bei den Qualitätskosten werden die Projekt- und Betriebskosten in vier Kategorien eingeteilt, die mit den Kosten für die Fehler des Produkts zusammenhängen:

- **Fehlerpräventionskosten:** Die Kosten für alle Maßnahmen, die geplant und proaktiv durchgeführt werden, um schlechte Qualität zu verhindern, z. B.
  - Entwickler zur Erfüllung ihrer Aufgaben qualifizieren, beispielsweise mit Trainings zur Erstellung von wartbarem und sicherem Code.
  - Möglichst frühzeitiges Review der Testbasis und entsprechende Kommunikation im Team.
- **Überprüfungskosten:** Die Kosten aller Maßnahmen, die auf die Erkennung von Fehlern abzielen, z. B.
  - Durchführung statischer und dynamischer Tests.
  - Reviews von Arbeitsergebnissen.
- **Interne Fehlerkosten:** Die Kosten aller reaktiven Aktivitäten, z. B.
  - Behebung von Fehlerzuständen, die beim Testen gefunden wurden.
  - Bereitstellung von Workarounds.
- **Externe Fehlerkosten:** Die Kosten aller nicht wertschöpfenden und reaktiven Maßnahmen, z. B.
  - Verlust von Einnahmen, Vermögenswerten, Gesundheit oder Menschenleben
  - Schäden an der Umwelt
  - Rechtskosten im Zusammenhang mit der Behebung von Fehlern
  - Zusätzliches Testen
  - Zusätzliche Bereitstellung
  - Zusätzlicher Support, weil ein fehlerhaftes Produkt an den Kunden ausgeliefert wurde ("Post-Release")
  - Behebung von Fehlern, die von Kunden gemeldet wurden

Die Gesamtkosten für die Überprüfung und die internen Fehlerkosten sind zusammen in der Regel deutlich geringer als die Kosten für externe Fehlerwirkungen. Das macht das Testen also äußerst wertvoll. Durch die Ermittlung der Kosten in diesen vier Kategorien können Testmanager einen überzeugenden Business Case für das Testen erstellen.

Es gibt mehrere Ansätze, die für die Definition der Qualitätskosten in Frage kommen. Der ISTQB®- Lehrplan unterstützt zwei davon. Der vorliegende Lehrplan basiert auf dem Ansatz von Feigenbaum und der ISTQB®-Lehrplan Foundation Level V4.0 stellt den Ansatz von Boehm vor (siehe ISTQB®-Lehrplan Foundation Level V4.0, Abschnitt 1.3 „Grundsätze des Testens“). Diese beiden Ansätze wurden ausgewählt, um ein breiteres Verständnis der Qualitätskosten zu erreichen. Der Ansatz von Feigenbaum (Feigenbaum, Nov/Dec 1956) betrachtet Qualität als einen kundenorientierten und unternehmensweiten Prozess, während der Ansatz von Boehm (Boehm, 1981) sich auf die Abwägung zwischen den Präventionskosten und den Fehlerkosten bei der Softwareentwicklung konzentriert (Hadjicostas, 2004).

3.2.2 Kosten-Nutzen-Verhältnis beim Testen

Während die meisten Unternehmen das Testen in gewisser Weise als wertvoll ansehen, können nur wenige Manager, einschließlich Testmanager, diesen Wert quantifizieren, beschreiben oder artikulieren. Darüber hinaus konzentrieren sich viele Testmanager, Testleiter und Tester auf die operativen Details des Testens (d. h. auf die Aspekte, die sich auf die Testaufgabe oder Teststufe beziehen). Sie ignorieren dabei meist die größeren taktischen und strategischen (also übergeordneten) Fragen im Zusammenhang mit dem Testen. Also jene Fragen, die anderen Stakeholdern, insbesondere Managern, am Herzen liegen.

Testen erbringt sowohl quantitative als auch qualitative Vorteile für das Unternehmen, das Projekt und/oder den Betrieb:

- **Zu den qualitativen Vorteilen** gehören ein besserer Ruf für Qualität, reibungslosere und besser vorhersehbare Releases, größeres Vertrauen, Schutz davor, rechtlich belangt zu werden, und es mindert das Risiko, das Einsatzziel zu verfehlen oder sogar Verletzte bzw.Todesopfer zu beklagen.
- **Zu den quantitativen Vorteilen** gehören gefundene oder verhinderte bzw. vor der Freigabe behobene Fehler, Fehler, die bereits vor der Freigabe bekannt waren (d. h. nicht behoben, aber dokumentiert, vielleicht mit Workarounds umgangen), Kostenvorteile (Boehm, 1981) (Böhler, 2008), die Verringerung des Risikoausmaßes durch die Durchführung von Tests und die Bereitstellung von Informationen zum Projekt-, Prozess- und Produktstatus.

Ein weiterer Vorteil des Testens ist, dass alle Beteiligten mit genügend Informationen versorgt werden, um fundierte Entscheidungen darüber zu treffen, ob die Qualität des Produkts ausreicht, um in Betrieb zu gehen. Sei es nun mit oder ohne Fehler. Manchmal ist es besser, mit bekannten Fehlern in Betrieb zu gehen, als zu warten, bis die Fehler beseitigt sind. In jenen Fällen, in denen ein Fehler toleriert werden kann, kommt es sehr auf die Wahrscheinlichkeit seines Auftretens und den Fehlerschweregrad an.

Die Qualitätskosten pro Fehler beim Testen werden wie folgt berechnet:

Durchschnittliche Einsparungen pro Fehler = Durchschnitt der externen Fehlerkosten - (Durchschnitt der Überprüfungskosten + Durchschnitt der internen Fehlerkosten).

Gesamtkosten der Qualität = (Fehlerpräventionskosten + (Überprüfungskosten * Anzahl der vor der Freigabe gefundenen Fehler)) + ((Interne Fehlerkosten * Anzahl der vor der Freigabe

gefundenen Fehler) + (Externe Fehlerkosten * Anzahl der nach der Freigabe gefundenen Fehler)).

Nehmen Sie als Beispiel an, dass Sie die folgenden Qualitätskosten pro Fehlerzustand für ein Produkt berechnet haben:

- Kosten für die Fehlerprävention: 180 EUR
- Durchschnittliche Kosten der Überprüfung pro Fehler: 500 EUR
- Durchschnittliche Kosten pro internen Fehler: 200 EUR
- Durchschnittliche Kosten des externen Fehlers pro Fehler: 4.000 EUR

Die durchschnittlichen Kosten der Fehlerprävention, der Überprüfung und der internen Fehler werden anhand der Anzahl der vor der Freigabe gefundenen Fehler berechnet, während die durchschnittlichen Kosten der externen Fehler anhand der Anzahl der nach der Freigabe gefundenen Fehler ermittelt werden. Mit diesen Werten können wir die durchschnittlichen Einsparungen pro Fehlerzustand wie folgt berechnet werden:

Durchschnittliche Einsparungen pro Fehlerzustand = 4.000 EUR - (500 EUR+ 200 EUR) = 3.300 EUR

Die Böhm-Kurve ist eine grafische Darstellung der Kosten für die Behebung von Fehlerzuständen im Laufe des SDLC. Daraus ergibt sich, dass Testen früher im SDLC durchgeführt werden sollte, um die Kosten für die Behebung von Fehlerzuständen zu senken. Die Böhm-Kurve zeigt, dass interne Fehlerkosten, d. h. die Kosten für die Behebung eines Fehlers, steigen, je später ein Fehlerzustand im SDLC entdeckt wird. Anhand dieser Informationen sollte der Testmanager versuchen, das optimale Verhältnis zwischen Fehlerpräventionskosten und den internen und externen Fehlerkosten zu finden.

Der Testaufwand muss sich nach dem spezifischen Risiko des Projekts und des Produkts richten und nach dem Risiko, das ein Unternehmen bereit ist, einzugehen. Zu viel Testen kann höhere Kosten verursachen, als die Verringerung des Risikoniveaus an Nutzen bringt. Wird zu wenig getestet, können übersehene Fehler ein hohes Risiko darstellen und höhere Kosten verursachen, als die nicht ausgeführten Tests gekostet hätten. Risikobasiertes Testen (siehe Abschnitt 1.3 „Risikobasiertes Testen“) unterstützt ein gutes Kosten-Nutzen-Verhältnis im Testen, indem der Testaufwand proportional zu den Risikostufen aufgebracht wird und die Priorisierung der Tests auf der Grundlage ihrer Risikostufen erfolgt.

Testmanager sollten erkennen, welche dieser Vorteile und Kosten für ihre Organisation, ihr Projekt und/oder ihren Betrieb zutreffend sind und sie sollten in der Lage sein, den Mehrwert des Testens in Bezug auf diese Vorteile und die Qualitätskosten pro Fehler zu kommunizieren.

4 Literaturhinweise

4.1 Normen und Standards

- IEC 61508 (2010) Functional safety of electrical/electronic/programmable electronic safety-related systems – Parts 1 to 7
- **ISO/IEC/IEEE 29119-2** (2021) ISO/IEC/IEEE 29119-2 Software and systems Engineering –Software testing – Part 2 Test processes
- **ISO/IEC/IEEE 29119-3** (2021) ISO/IEC/IEEE 29119-3 Software and systems Engineering –Software testing – Part 3 Test documentation

4.2 ISTQB®-Dokumente

- ISTQB® Certified Tester Advanced Level Syllabus Testautomatisierungsentwickler (2019)
- ISTQB® Certified Tester Agile Test Leadership at Scale Syllabus v2.0 (2023)
- ISTQB® Certified Tester Expert Level Test Management Syllabus v1.0 (2011)
- ISTQB® Certified Tester Expert Level Improving the Test Process Syllabus v1.0.1 (2011)
- ISTQB® Certified Tester Lehrplan Foundation Level V4.0 (2023)

4.3 Fachliteratur

- Basili, V., Trendowicz, A., Kowalczyk, M., Heidrich, J., Seaman, C., Münch, J., Rombach, D. (2014). *Aligning Organizations Through Measurement – The GQM+ Strategies Approach.* Springer International.
- Bath, G., van Veenendaal, E. (2014). *Improving the Test Process – chapter 6: Process for Improvement.* Rocky Nook.
- Belbin, R. M. (2010). *Management Teams: Why They Succeed or Fail.* London: Routledge.
- Black, R. (2009). *Managing the Testing Process, 3rd Edition.* John Wiley & Sons.
- Boehm, B. (1979). *Software Engineering Economics.* Prentice-Hall.
- Bonebright, D. A. (2010). *40 years of storming: a historical review of Tuckman's model of small group development* (1 Ausg., Bd. 13). Human Resource Development International, 1, 2010, Vol. 13.
- Craig, R., Jaskiel, S. P. (2002). *Systematic Software Testing.* Artech House.
- Derby, E., Larsen, D. (2006). *Agile Retrospectives – Making Good Teams Great.* The Pragmatic Bookshelf.
- Erpenbeck, J., von Rosenstiel, L. (2017). *Handbuch Kompetenzmessung.* Stuttgart: Schäffer-Poeschel.
- Fowler, M. (2010). *Hybrid development processes.* IEEE Software, 27(2), 57-63.
- Herzberg, F., Mausner, B., Bloch Snyderman, B. (1993). *Motivation to Work.* London: Routledge.

- Kahler, T. (2008). *The Process Therapy Model: The Six Personality Types with Adaptations.* Taibi Kahler Associates, Inc.
- Marston, W. M. (1999). *Emotions Of Normal People.* London: Routledge.
- Sonntag, K., Schmidt-Rathjens, C. (2005). *Anforderungsanalyse und Kompetenzmodelle.* Wiesbaden: VS Verlag für Sozialwissenschaften.
- Tuckman, B. W. (1965). *Developmental sequence in small groups* (Bd. 63(6)). Psychological Bulletin.
- van Ewijk, A. (2013). *TPI NEXT – Business Driven Test Process Improvement.* Sogeti Nederland B.V.
- van Solingen, R. Berghout, E. (1999). *The Goal Question Metric Method – A Practical Guide for Quality Improvement of Software Development.* McGraw-Hill.
- van Veenendaal, E. (2012). *The PRISMA Approach: Practical Risk-Based Testing.* UTN Publishers.
- van Veenendaal, E. (2020). *TMMi in the Agile world, version 1.4.* TMMi Foundation.
- van Veenendaal, E., Cannegieter, J. J. (2011). *The Little TMMi – Objective-Driven Test Process Improvement.* UTN Publishers.

4.4 Artikel und Internetquellen

- Feigenbaum, Armand V. (Nov/Dez 1956). Harvard Business Review, Vol. 34 Issue 6, p93-101.
- Hadjicostas, Evsevios (2004). Total Quality Management and Cost of Quality, Springer https://link.springer.com/chapter/10.1007/978-3-662-09621-5_7.
- www.tmmi.org Test Maturity Model Integration (TMMi®); letzter Besuch 31. Januar 2024.
- www.tmap.net/book/tpir-next Test Process Improvement (TPI); letzter Besuch 31. Januar 2024.
- www.wikipedia.org/wiki/PDCA Plan-Do-Check-Act; letzter Besuch 31. Januar 2024.
- https://www.asqf.de/soft-vs-hard-skills-in-software-testing/ „Soft vs. Hard Skills in Software Testing"; letzter Besuch 18.01.2025.

Die vorstehenden Quellenangaben verweisen auf Informationen, die im Internet und anderswo verfügbar sind. Auch wenn diese Verweise zum Zeitpunkt der Veröffentlichung dieses Lehrplans geprüft wurden, kann das ISTQB® nicht dafür verantwortlich gemacht werden, wenn die Verweise nicht mehr verfügbar sind.

4.5 Deutschsprachige Literatur

- Kribernegg, Anja. SOFTWARE – TEST IT PROFESSION@LLY: Ausbildung für zertifizierte Softwaretester und Testmanager (CTAL-TM 3.0: S.390-618). Selfpublishing. AMAZON, Kindle-Version, 2024.
- Roßner, Th.; Winter, M.; Spillner, A.; Linz, T. (2025): Praxiswissen Softwaretest - Testmanagement. 5., völlig überarbeitete Auflage, dpunkt.verlag, Heidelberg, Juni 2025.

5 Anhang A – Lernziele/kognitive Stufen des Wissens

Die spezifischen Lernziele (Learning Objectives, LO), die für diesen Lehrplan gelten, sind am Anfang jedes Kapitels aufgeführt. Jedes Thema des Lehrplans wird entsprechend seinem Lernziel untersucht.

Die Lernziele beginnen mit einem Aktionsverb, das dem jeweiligen kognitiven Wissensstand entspricht, wie unten aufgeführt.

Wissensstufe 1: Sich erinnern (K1) – Der Lernende kann einen Begriff oder ein Konzept erkennen, sich erinnern oder abrufen.

Aktionswörter: Erinnern, erkennen.

Beispiele
Erinnern Sie sich an die Konzepte der Testpyramide.
Erkennen Sie die typischen Ziele des Testens.

In diesem Advanced-Level-Lehrplan wurden keine K1-Lernziele definiert.

Wissensstufe 2: Verstehen (K2) – Der Lernende kann die Gründe für, oder Erklärungen zu Aussagen zu einem Thema auswählen. Typische beobachtbare Leistungen zusammenfassen, vergleichen, einordnen und Beispiele für Konzepte des Testens nennen.

Aktionswörter: Klassifizieren, vergleichen, differenzieren, unterscheiden, erklären, Beispiele nennen, schließen, zusammenfassen

Beispiele	**Anmerkungen**
Klassifizieren Sie Testwerkzeuge nach ihrem Zweck und den Testaktivitäten, die sie unterstützen.	
Vergleichen Sie die verschiedenen Teststufen.	Kann verwendet werden, um nach Ähnlichkeiten, Unterschieden oder beidem zu suchen.
Differenzieren Sie zwischen Testen und Debugging.	Sucht nach Unterschieden zwischen Konzepten.
Unterscheiden Sie zwischen Projektrisiken und Produktrisiken.	Ermöglicht die separate Klassifizierung von zwei (oder mehr) Konzepten.
Erklären Sie, wie sich der Kontext auf den Testprozess auswirkt.	
Nennen Sie Beispiele dafür, warum Testen notwendig ist.	
Schließen Sie aus einem gegebenen Profil von Fehlerzuständen auf die Grundursache von Fehlern.	

Beispiele	Anmerkungen
Fassen Sie die Aktivitäten des Work-Product- Review-Prozesses zusammen.	

Wissensstufe 3: Anwenden (K3) – Der Lernende kann ein Verfahren anwenden, wenn er mit einer vertrauten Aufgabe konfrontiert wird, oder das richtige Verfahren auswählen und es auf einen bestimmten Kontext anwenden.

Aktionswörter: Anwenden, umsetzen, vorbereiten, nutzen

Beispiele	Anmerkungen
Wenden Sie die Grenzwertanalyse an, um Testfälle aus gegebenen Anforderungen abzuleiten.	Sollte sich auf ein Verfahren/Technik/Prozess etc. beziehen.
Setzen Sie Methoden zur Erfassung von Metriken um, um technische und Managementanforderungen zu unterstützen.	
Bereiten Sie Tests zur Installierbarkeit von mobilen Anwendungen vor.	
Nutzen Sie die Verfolgbarkeit zur Überwachung des Testfortschritts auf Vollständigkeit und Konsistenz mit den Testzielen, der Teststrategie und dem Testkonzept.	Könnte in einem LO verwendet werden, in dem der Kandidat in der Lage sein soll, eine Technik oder ein Verfahren zu nutzen. Ähnlich wie „anwenden“.

Wissensstufe 4: Analysieren (K4) – Der Kandidat kann Informationen, die sich auf ein Verfahren beziehen, zum besseren Verständnis in ihre Bestandteile zerlegen und zwischen Fakten und Schlussfolgerungen unterscheiden. Eine typische Anwendung ist die Analyse eines Dokuments, einer Software oder einer Projektsituation und das Vorschlagen geeigneter Maßnahmen, um ein Problem zu lösen oder eine Aufgabe zu erfüllen.

Aktionswörter: Analysieren, zerlegen, skizzieren, priorisieren, auswählen

Beispiele	Anmerkungen
Analysieren Sie eine gegebene Projektsituation, um festzustellen, welche Black-Box-Testverfahren oder erfahrungsbasierte Testverfahren angewendet werden sollten, um bestimmte Ziele zu erreichen.	Nur in Verbindung mit einem messbaren Ziel der Analyse prüfbar. Sollte von der Form „Analysieren Sie xxxx, um zu xxxx“ (oder ähnlich) sein.
Priorisieren Sie Testfälle in einer bestimmten Testsuite für die Ausführung auf der Grundlage der damit verbundenen Produktrisiken.	

Beispiele	Anmerkungen
Wählen Sie die geeigneten Teststufen und Testarten aus, um eine bestimmte Gruppe von Anforderungen zu überprüfen.	Wird benötigt, wenn die Auswahl eine Analyse erfordert.

Referenz für die kognitiven Ebenen der Lernziele):

- Anderson, L. W., Krathwohl, D. R. (Hrsg.) (2001). *A Taxonomy for Learning, Teaching, and Assessing*: Eine Überarbeitung von Blooms Taxonomie der Bildungsziele, Allyn & Bacon.

Die spezifischen Lernziele, die für diesen Lehrplan gelten, sind am Anfang jedes Kapitels aufgeführt.

6 Anhang B – Verfolgbarkeitsmatrix des geschäftlichen Nutzens (Business Outcomes) mit Lernzielen

In diesem Abschnitt wird die Verfolgbarkeit zwischen den geschäftlichen Nutzen und den Lernzielen des Lehrplans für Advanced Level Test Management aufgeführt.

Geschäftlicher Nutzen: Advanced Level Test Management			**BO1**	**BO2**	**BO3**	**BO4**	**BO5**	**BO6**	**BO7**	**BO8**	**BO9**	**BO10**	**BO11**
TM_01	Tests in verschiedenen Softwareentwicklungsprojekten durch Anwendung von Testmanagementprozessen managen, die für das Projektteam oder die Testorganisation festgelegt wurden.		12										
TM_02	Stakeholder und Softwareentwicklungslebenszyklus-Modelle identifizieren, die in einem bestimmten Kontext relevant sind.			4									
TM_03	Eine Risikoidentifizierung und Risikobewertung innerhalb eines Softwareentwicklungslebenszyklus organisieren und die Ergebnisse nutzen, um das Testen so zu steuern, dass die Testziele erreicht werden.				6								
TM_04	Eine Teststrategie für das Projekt im Einklang mit der organisationsweiten Teststrategie und dem Projektkontext definieren.					11							
TM_05	Tests zur Erreichung der Projektziele kontinuierlich überwachen und steuern.						4						
TM_06	Den Testfortschritt an die Projektbeteiligten bewerten und berichten.							3					

Geschäftlicher Nutzen: Advanced Level Test Management			BO1	BO2	BO3	BO4	BO5	BO6	BO7	BO8	BO9	BO10	BO11
TM_07	Die notwendigen Kompetenzen identifizieren und diese Kompetenzen im Team entwickeln.								6				
TM_08	Einen Business Case für das Testen in verschiedenen Kontexten erstellen und präsentieren, der die Kosten und den erwarteten Nutzen darlegt.									5			
TM_09	Testprozessverbesserungsaktivitäten in Projekten oder Softwareentwicklungsprodukt-Streams leiten und zu organisatorischen Initiativen der Testprozessverbesserung beitragen.										5		
TM_10	Die Testaktivitäten einschließlich der erforderlichen Testinfrastruktur planen und den für das Testen erforderlichen Aufwand schätzen.											9	
TM_11	Fehlerberichte erstellen und einen an den Softwareentwicklungslebenszyklus angepassten Fehlerworkflow definieren.												6
Prüfbare Lernziele	Lernziel – Der Lernende kann …	K-Level											
1	Die Testaktivitäten managen												
1.1	Der Testprozess												
TM-1.1.1	… die Aktivitäten der Testplanung zusammenfassen	K2	X			X							
TM-1.1.2	… die Aktivitäten der Testüberwachung und Teststeuerung zusammenfassen	K2	X				X						

Geschäftlicher Nutzen: Advanced Level Test Management			BO1	BO2	BO3	BO4	BO5	BO6	BO7	BO8	BO9	BO10	BO11
TM-1.1.3	… die Aktivitäten des Testabschlusses zusammenfassen	K2	X					X					
1.2	Der Kontext des Testens												
TM-1.2.1	… vergleichen, warum verschiedene Stakeholder am Testen interessiert sind	K2		X		X							
TM-1.2.2	… erklären, warum das Wissen der Stakeholder im Testmanagement relevant ist	K2		X		X							
TM-1.2.3	… das Testen in einem hybriden Softwareentwicklungsmodell erklären	K2		X		X							
TM-1.2.4	… die Aktivitäten des Testmanagements für verschiedene Softwareentwicklungslebenszyklen zusammenfassen	K2	X	X		X							
TM-1.2.5	… die Aktivitäten des Testmanagements auf verschiedenen Teststufen miteinander vergleichen	K2	X			X							
TM-1.2.6	… die Aktivitäten des Testmanagements für verschiedene Testarten miteinander vergleichen	K2	X			X							
TM-1.2.7	… ein vorgegebenes Projekt analysieren und dazu Testmanagementaktivitäten, insbesondere Testplanung, Testüberwachung und Teststeuerung, festlegen	K4	X			X							
1.3	Risikobasiertes Testen												

Geschäftlicher Nutzen: Advanced Level Test Management			BO1	BO2	BO3	BO4	BO5	BO6	BO7	BO8	BO9	BO10	BO11
TM-1.3.1	… die verschiedenen Maßnahmen erklären, die bei risikobasiertem Testen ergriffen werden müssen, um auf Risiken zu reagieren	K2			X								
TM-1.3.2	… Beispiele für verschiedene Verfahren nennen, die ein Testmanager zur Identifizierung von Risiken im Zusammenhang mit der Produktqualität verwenden kann	K2			X								
TM-1.3.3	… die Faktoren zusammenfassen, die die Risikostufen in Bezug auf die Produktqualität beeinflussen	K2			X								
TM-1.3.4	… geeignete Testaktivitäten zur Minderung von Risiken gemäß ihrer Risikostufe in einem bestimmten Kontext auswählen	K4			X								
TM-1.3.5	… Beispiele von schwergewichtigen und leichtgewichtigen risikobasierten Testverfahren unterscheiden	K2			X								
TM-1.3.6	… Beispiele für Erfolgsmetriken und Schwierigkeiten im Zusammenhang mit risikobasierten Tests nennen	K2			X								
1.4	Die Teststrategie für das Projekt												
TM-1.4.1	… die typische Auswahl an Testansätzen erklären	K2				X							
TM-1.4.2	… eine organisationsweite Teststrategie und den Projektkontext analysieren und dann damit einen geeigneten Testansatz auswählen	K4				X							

Geschäftlicher Nutzen: Advanced Level Test Management			BO1	BO2	BO3	BO4	BO5	BO6	BO7	BO8	BO9	BO10	BO11
TM-1.4.3	… die S.M.A.R.T.-Zieldefinitionsmethode verwenden, um messbare Testziele und Endekriterien zu definieren	K3				X							
1.5	Verbesserung des Testprozesses												
TM-1.5.1	… erklären, wie das IDEAL-Modell zur Testprozessverbesserung bei einem bestimmten Projekt einzusetzen ist	K2									X		
TM-1.5.2	… den modellbasierten Ansatz zur Verbesserung des Testprozesses zusammenfassen und verstehen, wie man ihn im Projektkontext anwendet	K2									X		
TM-1.5.3	… den analytisch basierten Ansatz zur Verbesserung des Testprozesses zusammenfassen und verstehen, wie man ihn im Projektkontext anwendet	K2									X		
TM-1.5.4	… eine Projekt- oder Iterationsretrospektive durchführen, um Testprozesse zu bewerten und verbesserungswürdige Testbereiche zu ermitteln	K3									X		
1.6	Testwerkzeuge												
TM-1.6.1	… die Best Practices für die Einführung von Werkzeugen zusammenfassen	K2										X	
TM-1.6.2	… die Auswirkungen der verschiedenen technischen und fachlichen Aspekte bei der Entscheidung für eine Werkzeugart erklären	K2										X	
TM-1.6.3	… eine gegebene Situation analysieren und dann damit einen Plan für die Auswahl von	K4										X	

Geschäftlicher Nutzen: Advanced Level Test Management			BO1	BO2	BO3	BO4	BO5	BO6	BO7	BO8	BO9	BO10	BO11
	Werkzeugen erstellen, der Risiken, Kosten und Nutzen berücksichtigt												
TM-1.6.4	… die Phasen des Lebenszyklus eines Werkzeugs voneinander unterscheiden	K2										X	
TM-1.6.5	… Beispiele für die Erfassung und Bewertung von Metriken mithilfe von Testwerkzeugen nennen	K2									X	X	
2	Das Produkt managen												
2.1	Testmetriken												
TM-2.1.1	… Beispiele für Metriken zur Erreichung der Testziele nennen	K2					X						
TM-2.1.2	… erklären, wie man den Testfortschritt mithilfe von Testmetriken steuern kann	K2					X						
TM-2.1.3	… Testergebnisse analysieren, um Testberichte zu erstellen, die es Stakeholdern ermöglichen, Entscheidungen zu treffen	K4					X	X					
2.2	Testschätzung												
TM-2.2.1	… die Hauptmerkmale erklären, die bei der Testschätzung berücksichtigt werden müssen	K2	X							X		X	
TM-2.2.2	… Beispiele für Faktoren nennen, die Testschätzungen beeinflussen können	K2	X							X		X	
TM-2.2.3	… für einen vorgegebenen Kontext ein geeignetes Verfahren oder einen geeigneten Ansatz für die Testschätzung auswählen	K4	X							X		X	

Geschäftlicher Nutzen: Advanced Level Test Management			BO1	BO2	BO3	BO4	BO5	BO6	BO7	BO8	BO9	BO10	BO11
2.3	Fehlermanagement												
TM-2.3.1	… einen Fehlermanagementprozess samt Fehlerworkflow umsetzen, der zur Überwachung und Steuerung von Fehlerzuständen verwendet werden kann	K3											X
TM-2.3.2	… den Prozess und die erforderlichen Teilnehmer für ein effektives Fehlermanagement erklären	K2											X
TM-2.3.3	… die Besonderheiten des Fehlermanagements in der agilen Softwareentwicklung erklären	K2	X										X
TM-2.3.4	… die Herausforderungen für das Fehlermanagement bei der hybriden Entwicklung von Software erklären	K2	X										X
TM-2.3.5	… die Daten und Klassifizierungen verwenden, die während des Fehlermanagements gesammelt werden sollten	K3											X
TM-2.3.6	… erklären, wie Statistiken aus Fehlerberichten verwendet werden können, um daraus Prozessverbesserungen abzuleiten	K2										X	X
3	Das Team managen												
3.1	Das Testteam												
TM-3.1.1	… Beispiele für typische Kompetenzen nennen, die von Mitgliedern eines Testteams	K2							X				

Geschäftlicher Nutzen: Advanced Level Test Management			BO1	BO2	BO3	BO4	BO5	BO6	BO7	BO8	BO9	BO10	BO11
	in den vier Kompetenzbereichen benötigt werden												
TM-3.1.2	… einen gegebenen Projektkontext analysieren, um daraus die geforderten Kompetenzen der Mitglieder des Testteams abzuleiten	K4							X				
TM-3.1.3	… typische Verfahren zur Bewertung der Kompetenz von Mitgliedern eines Testteams erklären	K2							X				
TM-3.1.4	… die typischen Ansätze zur Entwicklung der Kompetenzen von Mitgliedern eines Testteams voneinander unterscheiden	K2							X				
TM-3.1.5	… die für die Leitung eines Testteams erforderlichen Managementkompetenzen erklären	K2							X				
TM-3.1.6	… Beispiele für motivierende Faktoren und Hygienefaktoren für Mitglieder eines Testteams erklären	K2							X				
3.2	Stakeholder-Beziehungen												
TM-3.2.1	… Beispiele für jede der vier Kategorien nennen, aus denen sich die Qualitätskosten ermitteln lassen	K2								X			
TM-3.2.2	… eine Berechnung des Kosten-Nutzen-Verhältnisses zur Abschätzung des Mehrwerts des Testens durch die Stakeholder vornehmen	K3						X		X			

7 Anhang C – Release Notes

Der ISTQB®-Lehrplan Advanced Level Test Management V3.0 ist ein größeres Update, das auf dem Advanced Level Syllabus Testmanager 2012 basiert. Aus diesem Grund gibt es keine detaillierten Versionshinweise pro Kapitel und Abschnitt. Im Folgenden finden Sie jedoch eine Zusammenfassung der wichtigsten Änderungen.

In dieser Version wurden alle Lernziele (Learning Objectives, LOs) überarbeitet, um sie atomar zu machen und eine Eins-zu-eins-Verfolgbarkeit zwischen LOs und Lehrplanabschnitten zu schaffen, so dass es keinen Inhalt gibt, der nicht auch ein LO hat. Ziel ist es, diese Version leichter lesbar, verständlich, lernbar und übersetzbar zu machen, wobei der Schwerpunkt auf der Erhöhung des praktischen Nutzens und der Ausgewogenheit zwischen Wissen und Fähigkeiten liegt.

In dieser Hauptversion wurden die folgenden Änderungen vorgenommen:

- Reduzierung des gesamten Lehrplans. Ein Lehrplan ist kein Lehrbuch, sondern ein Dokument, das dazu dient, die grundlegenden Elemente eines Aufbaukurses zum Thema Softwaretesten zu definieren, unter Berücksichtigung welche Themen auf welcher kognitiven Ebene behandelt werden sollen. Daher gilt insbesondere:
  - In den meisten Fällen sind die Beispiele nicht im Text enthalten. Es ist die Aufgabe eines Schulungsanbieters, die Beispiele sowie die Übungen während der Schulung bereitzustellen.
  - Die "Syllabus writing checklist" wurde befolgt, die die maximale Textgröße für LOs auf jeder K-Stufe vorschlägt (K2 = maximal 1.500 nicht leere Zeichen, K3 = maximal 2.500 nicht leere Zeichen, K4 = maximal 3.000 nicht leere Zeichen, +/- 20 %).
- Verringerung der Anzahl der LOs im Vergleich zum ISTQB® CTAL TM Lehrplan 2012:
  - 36 K2-LOs im Vergleich zu 39 LOs in CTAL TM 2012
  - 5 K3-LOs im Vergleich zu 12 LOs in CTAL TM 2012
  - 7 K4-LOs im Vergleich zu 10 LOs in CTAL TM 2012
- Die gesamte Struktur des Lehrplans wurde überarbeitet.
- Die Angleichung an den ISTQB®-Lehrplan Foundation Level V4.0 ist abgeschlossen.
- Wesentliche Änderungen im früheren Kapitel 1 (Testprozess) der Ausgabe 2012:
  - Beschränkt sich auf die Verwaltung der Testaktivitäten (Testplanung, Testüberwachung, Teststeuerung und Testabschluss).
  - Integriert als Abschnitt in das neue Kapitel "Die Testaktivitäten managen".
- Neues Kapitel **Die Testaktivitäten managen**
  - Abschnitt 1.1 – Der Testprozess: siehe oben.
  - Abschnitt 1.2 – Der Kontext des Testens: Erweitert, um nicht sequenzielle Entwicklungsmodelle für Software zu überdecken.

- Abschnitt 1.3 – Risikobasiertes Testen: Vollständig umgeschrieben, um es auf Projektebene besser anwendbar zu machen.
- Abschnitt 1.4 – Die Teststrategie für das Projekt: Da das Testkonzept bereits im ISTQB®-Lehrplan Foundation Level V4.0 definiert ist, liegt der Schwerpunkt auf der Auswahl des geeigneten Testansatzes und der Definition messbarer Testziele.
- Abschnitt 1.5 – Verbesserung des Testprozesses: Integration in das Management der Testaktivitäten, Darstellung der Anwendung im Projektkontext und Umsetzung durch Retrospektiven innerhalb einer Iteration oder eines Projekts.
- Abschnitt 1.6 – Testwerkzeuge: Einführung von Werkzeugen wurde aus dem ISTQB®-Lehrplan Foundation Level V3.1D übernommen (ist im ISTQB®-Lehrplan Foundation Level V4.0 nicht enthalten).

- Neues Kapitel **Das Produkt managen**
  - Abschnitt 2.1 – Testmetriken: Frühere Abschnitte, die Metriken und die Verwendung von Metriken definieren, Testmetriken.
  - Abschnitt 2.2 – Testschätzung: Der ISTQB®-Lehrplan Foundation Level V4.0 behandelt bereits die Berechnung der Testschätzung. Der Abschnitt wurde um die Auswahl geeigneter Testschätzungsverfahren auf K4-Niveau und die Verwendung von Testschätzungen in den SDLC-Modellen erweitert.
  - Abschnitt 2.3 – Fehlermanagement: Angepasst an die neuesten Ausgaben der Standards und erweitert für die Verwendung in der agilen und hybriden Softwareentwicklung.
- Neues Kapitel **Das Team managen**
  - Abschnitt 3.1 – Das Testteam: Die Hauptthemen sind dieselben wie im ISTQB® TM-2012-Lehrplan: Identifizieren Sie die einzelnen Fähigkeiten und stellen Sie die Testteams zusammen.
  - Abschnitt 3.2 – Stakeholder-Beziehungen: Es handelt sich um den ehemaligen Abschnitt des TM-2012-Lehrplans mit dem Titel "Mehrwert des Testens".
- Wichtige Änderungen und entfernte Abschnitte/Kapitel im ISTQB® CTAL TM Lehrplan 2012:
  - Abschnitt über verteiltes, Outsourcing- und Insourced-Testen entfernt.
  - Abschnitt über das Management der Anwendung von Industriestandards entfernt.
  - Kapitel über Reviews entfernt.
  - Unterabschnitte zur Verbesserung des Testprozesses CTP und STEP entfernt.
  - Unterabschnitte zu Testanalyse, Testentwurf, Testrealisierung und Testdurchführung entfernt.

8 Anhang D – Domänenspezifische Schlüsselbegriffe

Begriff Name	Definition
Breitband-Delphi	Ein auf Experten basierendes Testschätzungsverfahren, das darauf abzielt, mithilfe der kollektiven Weisheit der Teammitglieder eine genaue Schätzung vorzunehmen.
Drei-Punkt-Schätzung	Bei diesem expertenbasierten Verfahren werden drei Schätzungen von den Experten vorgenommen: die optimistischste Schätzung (a), die wahrscheinlichste Schätzung (m) und die pessimistischste Schätzung (b). Die endgültige Schätzung (E) ist ihr gewichtetes arithmetisches Mittel.
IDEAL	Ein Modell zur Organisationsverbesserung, das als Fahrplan für die Initiierung, Planung und Umsetzung von Verbesserungsmaßnahmen dient.
Indikator	Eine Messung, die eine Schätzung oder Bewertung bestimmter, aus einem Modell abgeleiteter Attribute im Hinblick auf einen definierten Informationsbedarf liefert.
Messung	Die Zahl oder Kategorie, die einem Attribut einer Entität durch eine Messung zugewiesen wird.
Metrik	Eine Messskala und die für die Messung verwendete Methode.
Planungspoker	Ein konsensbasiertes Schätzverfahren, das meist zur Schätzung des Aufwands oder der relativen Größe von User Stories in der agilen Softwareentwicklung verwendet wird. Es handelt sich dabei um eine Variante der Breitband-Delphi-Methode, bei der ein Kartenspiel mit Werten verwendet wird, die die Einheiten darstellen, in denen das Team schätzt.
Ziel-Frage-Metrik (GQM)	Ein Ansatz zur Messung von Software anhand eines Drei-Ebenen-Modells, das aus einer konzeptionellen Ebene (Ziel), einer operativen Ebene (Frage) und einer quantitativen Ebene (Metrik) besteht.

9 Anhang E – Eingetragene Marken

- CMMI® ist eine eingetragene Marke der Carnegie Mellon University beim U.S. Patent and Trademark Office.
- ISTQB® ist eine eingetragene Marke des International Software Testing Qualifications Board.
- TMMi® ist eine eingetragene Marke der TMMi Foundation.
- TPI NEXT® ist eine eingetragene Marke von Sogeti, Niederlande.

10 Index

Alle Testbegriffe sind im ISTQB®-Glossar definiert (http://glossary.istqb.org/).

U