

TMMi in der agilen Welt

Version 1.4

Produziert von der TMMi Foundation

Herausgeber: Erik van Veenendaal

Copyright-Hinweis
Unbeschränkte Verbreitung vorbehaltlich
des Urheberrechts Copyright © TMMi
Foundation, Irland.

Dieses Material der TMMi Foundation wird auf der Grundlage des Ist-Zustandes bereitgestellt.

Die TMMi Foundation gibt keinerlei ausdrückliche oder stillschweigende Garantien in Bezug auf irgendeine Angelegenheit, einschließlich, aber nicht beschränkt auf Garantien für die Eignung für einen bestimmten Zweck oder die Marktgängigkeit, die Exklusivität oder die durch die Verwendung des Materials erzielten Ergebnisse. Die TMMi Foundation übernimmt keine Garantie hinsichtlich der Freiheit von Patent-, Marken- oder Urheberrechtsverletzungen.

Die Verwendung von Marken in diesem Dokument soll in keiner Weise die Rechte der Markeninhaber verletzen.

Die Erlaubnis, dieses Dokument zu vervielfältigen und davon abgeleitete Arbeiten für den internen Gebrauch anzufertigen, wird erteilt, vorausgesetzt, dass alle Vervielfältigungen und abgeleiteten Arbeiten den Hinweis auf das Urheberrecht und "Keine Gewährleistung" enthalten.

Anträge auf Genehmigung zur Vervielfältigung dieses Dokuments oder zur Erstellung abgeleiteter Werke dieses Dokuments für externe und kommerzielle Zwecke sind an die TMMi Foundation zu richten.

TMMi® ist eine eingetragene Marke der TMMi Foundation.

Mitwirkende

Asim Ali	(Vereinigte Arabische Emirate)
Katalin Balla	(Ungarn)
Clive Bates	(UK)
Jan Jaap Cannegieter	(Niederlande)
Vahid Garousi	(Niederlande)
Alon Linetzki	(Israel)
Fran O'Hara	(Irland)
Jurian van de Laar	(Niederlande)
Leanne Howard	(Australien)
Poonam Jain	(Indien)
Tim Moore	(UK)
Alfonsina Morgavi	(Argentinien)
Meile Posthuma	(Niederlande)
Matthias Rasking	(Deutschland)
Chaobo Shang	(China)
Erik van Veenendaal	(Bonaire- Karibik Niederlande)
Blaine Webb	(UK)
Karolina Zmitrowicz	(Polen)

Versionsgeschichte

Dieser Abschnitt dient nur zu Informationszwecken.

Version	Datum	Kommentar
1.0	30/06/2017	Einführungskapitel und alle TMMi Level 2 Prozessbereiche angesprochen
1.1	16/05/2018	Die Interpretation in einem agilen Kontext aller TMMi Level 3 Prozessbereiche hinzugefügt.
1.2	10/12/2018	Die Interpretation in einem agilen Kontext aller TMMi Level 4 Prozessbereiche hinzugefügt.
1.3	03/07/2019	Die Interpretation in einem agilen Kontext aller TMMi Level 5 Prozessbereiche hinzugefügt.
1.4	24/12/2019	Die Abschnitte 6.4 und 6.5 wurden hinzugefügt, um ausdrücklich darauf hinzuweisen, dass alle TMMi-Stufen 4 und 5 anwendbar sind.

Inhalt

Inhalt

1	Einführung.....	6
1.1	Zielsetzungen.....	6
1.2	TMMi und Agilität.....	6
1.3	Test Maturity Model integrated (TMMi).....	7
1.4	Agilität.....	8
1.5	Verbesserung des Testprozesses in einem agilen Kontext.....	9
2	TMMi-Stufe 2 Gelenkt.....	11
2.1	Prozessbereich 2.1 Testtrichtlinie und Teststrategie	11
2.2	Prozessbereich 2.2 Testplanung	13
2.3	Prozessbereich 2.3 Testüberwachung und Teststeuerung.....	17
2.4	Prozessbereich 2.4 Testentwurf und Testdurchführung	20
2.5	Prozessbereich 2.5 Testumgebung	24
3	TMMi-Stufe 3 Definiert.....	26
3.1	Prozessbereich 3.1 Organisation der Tests	26
3.2	Prozessbereich 3.2 Testschulungsprogramm	29
3.3	Prozessbereich 3.3 Testlebenszyklus und Integration.....	30
3.4	Prozessbereich 3.4 Nicht-funktionale Tests	32
3.5	Prozessbereich 3.5 Peer-Reviews	35
4	TMMi-Stufe 4 Gemessen.....	38
4.1	Prozessbereich 4.1 Messung von Tests	38
4.2	Prozessbereich 4.2 Bewertung der Produktqualität	39
4.3	Prozessbereich 4.3 Fortgeschrittene Reviews	40
5	TMMi-Stufe 5 Optimierung.....	42
5.1	Prozessbereich 5.1 Fehlerprävention	42
5.2	Prozessbereich 5.2 Qualitätssteuerung.....	43
5.3	Prozessbereich 5.3 Testprozessoptimierung.....	44
6	Überblick Anwendbarkeit TMMi-spezifische Ziele und Praktiken.....	46
6.1	TMMi Assessments.....	46
6.2	TMMi-Stufe 2 Gelenkt.....	46
6.3	TMMi-Stufe 3 Definiert	46
6.4	TMMi-Stufe 4 Gemessen	47
6.5	TMMi-Stufe 5 Optimierung	47
	Referenzen.....	48

1 Einführung

1.1 Zielsetzungen

In diesem Dokument wird erläutert, wie die Kombination eines agilen Ansatzes für die Softwareentwicklung mit dem TMMi-Modell zur Verbesserung des Testprozesses ein möglicher Weg zur Erreichung Ihrer Unternehmensziele ist. Es wird erläutert, wie TMMi in einem agilen Kontext vorteilhaft genutzt und angewendet werden kann. Es werden bewährte Alternativen zu traditionellen Testansätzen vorgestellt, die TMMi-Praktiken implementieren und dabei die Agilität beibehalten und vielleicht sogar erhöhen. Dieses Dokument zeigt auf, wie TMMi agilen Organisationen helfen kann, unabhängig davon, wie weit die Organisation auf ihrem Weg zu mehr Agilität fortgeschritten ist. Indem es an kritische Testpraktiken erinnert, die häufig aus dem Blickfeld geraten, unterstützt dieses Dokument selbst wenn Organisationen wachsen und der Projektdruck steigt. Jeder TMMi-Prozessbereich und seine spezifischen Ziele werden einzeln besprochen. Es wird dargelegt, wie sich diese auf Agilität beziehen und wie die Praktiken typischerweise aussehen werden. Wenn eine Praxis nicht in einem agilen Kontext durchgeführt werden soll, weil sie keinen zusätzlichen Nutzen bringt, wird dies ebenfalls ausdrücklich erwähnt.

Viele (kleine) agile Organisationen sind erfolgreich und wachsen, aber sie haben vielleicht nur wenige dokumentierte Prozesse und wenig formale Schulungsprogramme für die beteiligten Personen. Um den Erfolg aufrechtzuerhalten, wenn die Organisation wächst, brauchen sie mehr Prozessdisziplin. Die Organisationen haben dann in der Regel Angst, die agile Kultur zu verlieren, die zu ihrem derzeitigen Erfolg geführt hat. Dies ist eine der Herausforderungen, die vor uns liegen, wenn wir eine TMMi-Initiative zur Verbesserung der Testprozesse in einer agilen Umgebung starten.

Dieses Dokument richtet sich an viele Zielgruppen, wobei die beiden wichtigsten Gruppen sind:

- Ausgereifte traditionelle Organisationen, die auf Agilität umstellen und dabei ihre Prozessreife beibehalten wollen.
- Agile Organisationen, die erfolgreich sind und wachsen. Daher benötigen sie eine gewisse Prozessreife, während sie gleichzeitig die Vorteile der Agilität bewahren wollen.

Dieses Dokument ist nicht dazu gedacht, unabhängig vom ursprünglichen TMMi-Modell verwendet zu werden. Es soll denjenigen, die Testprozessverbesserungen in einer agilen Umgebung durchführen, eine Anleitung zur Interpretation und Bedeutung der verschiedenen TMMi-Ziele und -Praktiken geben. Dieses Dokument ergänzt das ursprüngliche TMMi-Modell und sollte als komplementäres Dokument verwendet werden.

1.2 TMMi und Agilität

Es besteht der Irrglaube, dass TMMi und agile Ansätze nicht miteinander vereinbar sind. Agile Ansätze und TMMi können nicht nur nebeneinander bestehen, sondern bringen bei erfolgreicher Integration auch erhebliche Vorteile. Eine weitere Herausforderung besteht darin, das Testen anders zu betrachten, es vollständig in die agile Entwicklung zu integrieren und zu verstehen, was das im Zusammenhang mit einem Verbesserungsprogramm "Test" bedeutet. Man beachte, dass das "i" von TMMi sich auf die Tatsache bezieht, dass Testen ein integrierter Teil der Softwareentwicklung sein sollte und nicht als etwas völlig Separates behandelt wird. In der Literatur und in Präsentationen über das Testen in agilen Projekten liegt der Schwerpunkt in der Regel auf Unit-Tests, Testautomatisierung und explorativen Tests, aber es gibt noch mehr! Die Verwendung des TMMi-Modells in einem agilen Kontext erinnert an kritische Testpraktiken, die oft "vergessen" werden. In diesem Dokument wird anhand von Beispielen gezeigt, dass TMMi und agile Methoden effektiv zusammenarbeiten können. Die Herausforderung besteht darin, Lean-Prinzipien anzuwenden, um agile Praktiken zu stärken und TMMi-Praktiken zu erleichtern.

Bei der Implementierung von TMMi ist zu berücksichtigen, dass das TMMi-Modell nicht darauf abzielt, einer Organisation eine Reihe von Praktiken "aufzuzwingen", und auch nicht als Standard angewendet werden soll, dessen "Einhaltung" man beweisen muss. Bei richtiger Anwendung kann TMMi Ihnen helfen, die spezifischen Testbereiche zu finden, in denen Änderungen angesichts der Geschäftsziele von Nutzen sein können. Dies gilt unabhängig von dem angewandten Lebenszyklusmodell. Es ist wichtig, immer daran zu denken, dass TMMi-Praktiken eine erwartete Komponente sind, aber auch durch so genannte "alternative" Praktiken in Bezug auf eine definierte TMMi-Praxis erreicht werden können. Denken Sie immer daran, was die Intention der Anwendung ist, mit welcher Begründung

und wie sie dem Unternehmen einen Mehrwert bringt. In einer agilen Kultur wird die Absicht oft schon erreicht, allerdings durch eine alternative Vorgehensweise. In der Regel ist "jede" Lösung zulässig, solange sie sich an den geschäftlichen Anforderungen orientiert! Beim Einsatz von TMMi sollten Sie nicht zu viele Vorschriften machen, denn so war TMMi von Anfang an nicht gedacht. Interpretieren Sie die TMMi-Ziele und -Praktiken immer im Zusammenhang und dem Kontext mit Ihrer Situation. Indem Sie zunächst den Prozessbedarf in Ihrem spezifischen Geschäftskontext ermitteln, können Sie entscheiden, wie Sie die Prioritäten für die Prozessverbesserung setzen und vorantreiben.

Die meisten Konflikte, die zwischen dem TMMi und Agilität entstehen, beruhen entweder auf einer historischen TMMi-Sichtweise, wie eine "gute Praxis" aussehen sollte, wenn sie umgesetzt wird, oder auf einem Missverständnis des Grundgedankens für agile Praktiken, wie sie die agilen Werte unterstützen sollten. TMMi-Experten, einschließlich der (federführenden) Assessoren, müssen Botschaften, die versehentlich weitergegeben werden und die sich darauf beziehen, wie eine "gute TMMi-konforme" Praxis aussehen sollte, wenn sie umgesetzt wird, möglicherweise neu überdenken. Wenn agile Ansätze in geeigneter Weise zusammen mit TMMi-Prozessen implementiert werden, wird dies zu einer effektiven Umsetzung von Testpraktiken führen und nicht zu deren Abschaffung. Beachten Sie, dass es neben den spezifischen (Test-)Praktiken innerhalb des TMMi auch generische Praktiken gibt. Der Zweck der allgemeinen Praktiken besteht darin, die Institutionalisierung eines Prozessbereichs zu unterstützen, d. h. sicherzustellen, dass die Organisation über eine Infrastruktur verfügt, die den Prozessbereich unterstützt, wenn neue Mitarbeiter hinzukommen oder andere Veränderungen innerhalb der Organisation stattfinden.

Die Umstellung von traditioneller Softwareentwicklung auf Agilität kann auch die Initiative zum Beschneiden und Verschlinken der Prozesse, wie sie heute definiert sind, hervorbringen. Auf diese Weise werden TMMi-basierte Organisationen von der agilen Denkweise profitieren. Es besteht die Tendenz, Dinge in das TMMi-Modell hineinzulesen, die nicht vorhanden sind, und so unnötige, nicht wertschöpfende Prozesse und Arbeitsprodukte zu schaffen. Wenn man sich auf die Wurzeln und die Verbesserungsziele besinnt und das TMMi-Modell so einsetzt, wie es gedacht ist, unterstützt man die Ausrichtung von Prozessen mit echtem Mehrwert auf echte Prozessanforderungen und -ziele. Das Beschneiden und Verschlinken von Prozessen mit einer agilen Denkweise führt zu Prozessen, die das widerspiegeln, was die Menschen wirklich tun, und stellt sicher, dass nur Daten gesammelt werden, die auch genutzt werden. Die agile Denkweise wird auch dazu führen, dass man sich darauf konzentriert, die Dinge so einfach wie möglich zu halten, was in der Regel nicht einfach ist, aber für diejenigen, die eine TMMi-Implementierung durchführen, von Vorteil sein wird. Verbesserungen im Rahmen der agilen Denkweise erfolgen in der Regel durch kleine, befähigte Teams, die schnell handeln können, was eine weitere Möglichkeit ist, wie TMMi von der agilen Denkweise profitieren kann. Denken Sie auch daran, dass es einen natürlichen Bruch zwischen TMMi-Stufe 3 und TMMi-Stufe 4 und 5 gibt. Besonders agilen Organisationen wird empfohlen, die Praktiken der TMMi-Stufen 4 und 5 kritisch auszuwählen und durchzuführen, die von Bedeutung sind und einen Mehrwert bieten. Das TMMi ist zwar umfassend, aber um erfolgreich zu sein, müssen Organisationen die wichtigsten Testverfahren und Verbesserungen identifizieren, auf die sie sich konzentrieren müssen.

1.3 Test Maturity Model integrated (TMMi)

Das TMMi-Framework wurde von der TMMi Foundation als Leitfaden und Referenzrahmen für die Verbesserung von Testprozessen entwickelt und befasst sich mit Themen, die für Testmanager, Testingenieure, Entwickler und Softwarequalitätsexperten wichtig sind. Testen wird im Rahmen von TMMi im weitesten Sinne definiert und umfasst alle Aktivitäten, die mit der Qualität von Softwareprodukten zusammenhängen.

TMMi verwendet das Konzept der Reifegrade für die Prozessbewertung und -verbesserung. Darüber hinaus werden Prozessbereiche, Ziele und Praktiken identifiziert. Die Anwendung der TMMi-Reifekriterien verbessert den Testprozess und wirkt sich nachweislich positiv auf die Produktqualität, die Produktivität der Testingenieure und den Zeitaufwand aus. TMMi wurde entwickelt, um Organisationen bei der Evaluierung und Verbesserung ihrer Testprozesse zu unterstützen.

TMMi hat eine stufenweise Architektur für die Prozessverbesserung. Sie enthält Stufen oder Ebenen, die ein Unternehmen durchläuft, wenn sich sein Testprozess von einem ad hoc und nicht verwalteten zu einem verwalteten, definierten, gemessenen und optimierten Prozess entwickelt. Durch das Erreichen jeder Stufe wird sichergestellt, dass alle Ziele der jeweiligen Stufe erreicht wurden und die Verbesserungen die Grundlage für die nächste Stufe bilden.

Die interne Struktur von TMMi ist reich an Testpraktiken, die erlernt und systematisch angewendet werden können, um einen qualitativ hochwertigen Testprozess zu unterstützen, der sich in kleinen Schritten verbessert. TMMi besteht aus fünf Stufen, die die Reifegradhierarchie und den evolutionären Weg zur Verbesserung des Testprozesses vorgeben. Jede Stufe hat eine Reihe von Prozessbereichen, die eine Organisation implementieren muss, um den Reifegrad auf dieser Stufe zu erreichen. Die Prozessbereiche für jede Reifegradstufe des TMMi sind in Abbildung 1 dargestellt.

Ein Hauptgrundsatz des TMMi ist, dass es sich um ein generisches Modell handelt, das auf verschiedene Lebenszyklusmodelle und Umgebungen anwendbar ist. Die meisten Ziele und Praktiken, wie sie im TMMi definiert sind, haben sich sowohl bei sequentiellen als auch bei iterativen Lebenszyklusmodellen, einschließlich agilen, als anwendbar erwiesen. Auf der untersten Stufe des Modells sind jedoch viele der Unterpraktiken und Beispiele (sehr) unterschiedlich, je nachdem, welches Lebenszyklusmodell angewendet wird. Beachten Sie, dass innerhalb des TMMi nur die Ziele verbindlich sind, die Praktiken jedoch nicht.

TMMi ist auf der Website der TMMi Foundation frei verfügbar. Das Modell wurde bereits in verschiedene Sprache übersetzt. TMMi ist auch in Buchform erhältlich.

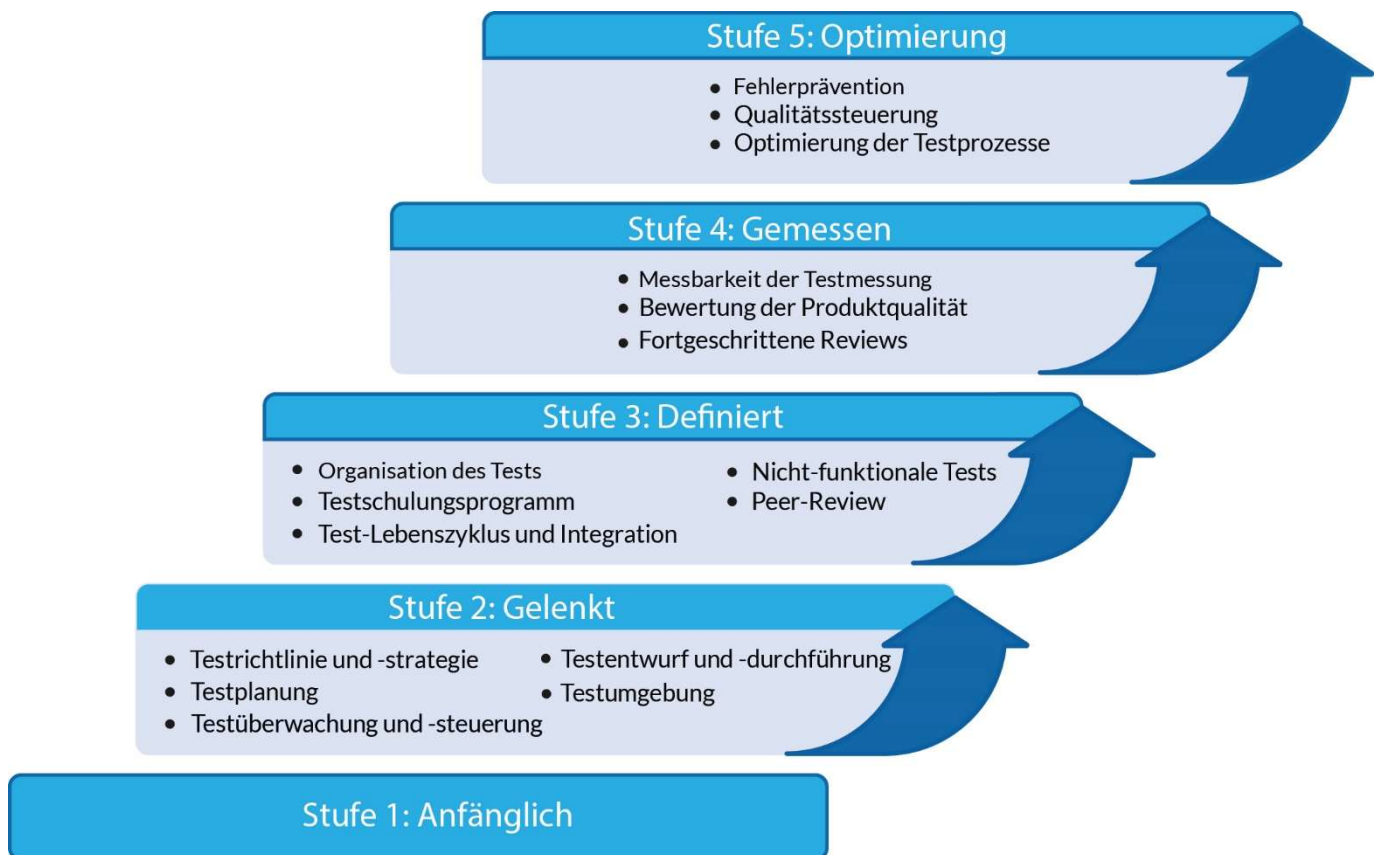


Abbildung 1: TMMi-Reifegrade und Prozessbereiche

1.4 Agilität

Im Jahr 2001 einigte sich eine Gruppe von Personen, die die am weitesten verbreiteten leichtgewichtigen Softwareentwicklungsmethoden repräsentieren, auf eine gemeinsame Reihe von Werten und Prinzipien, die als Manifest für agile Softwareentwicklung oder Agiles Manifest bekannt wurden. Das agile Manifest enthält vier wertende Kernaussagen:

- Individuen und Interaktionen *vor* Prozessen und Werkzeugen
- Funktionierende Software *über* umfassende Dokumentation
- Zusammenarbeit mit dem Kunden *statt* Vertragsverhandlungen
- Reagieren auf Veränderungen *statt* Befolgen eines Plans

Im Agilen Manifest wird argumentiert, dass die Konzepte auf der rechten Seite zwar einen Wert haben, die Konzepte

auf der linken Seite jedoch von größerem Wert sind. Agilität selbst ist keine Methode, aber es wurden inzwischen verschiedene Methoden entwickelt, die Praktiken zur Umsetzung dieser Werte bieten. Drei wichtige Vertreter von Methodologien, die Agilität unterstützen, sind Extreme Programming (XP), Scrum und Kanban.

1.5 Verbesserung des Testprozesses in einem agilen Kontext

Bei der agilen Methode liegt der Schwerpunkt darauf, dass die Teams ihre eigenen kontinuierlichen Verbesserungen vor Ort durchführen, auch durch häufige Retrospektiven. Einige dieser Verbesserungen können testorientiert sein. Diese können durch ein TMMi-basiertes Verbesserungsprojekt auf höherer Ebene gesteuert werden, können aber auch auf lokale Prozessprobleme des Teams in Bezug auf das Testen abzielen. Es ist sehr wichtig, dass die Maßnahmen im Kontext der Testprozessverbesserung nicht so interpretiert und/oder verwendet werden, dass sie den agilen Teams die lokale Verantwortung abnehmen.

Einige der wichtigsten Aspekte, die bei der Betrachtung des Einflusses von Agilität auf den Verbesserungskontext zu berücksichtigen sind, sind:

- Häufigkeit des Verbesserungszyklus
- Organisatorische Aspekte
- Umfang der Verbesserungen
- Quelle der Verbesserungen
- Niveau der (Test-)Dokumentation
- Methoden zur Verbesserung.

In agilen Projekten finden Verbesserungen in der Regel in häufigen Feedbackschleifen statt, die es ermöglichen, Verbesserungen des Testprozesses häufig in Erwägung zu ziehen (z. B. am Ende eines Sprints, wenn Scrum verwendet wird). Da der Anwendungsbereich oft auf den vorangegangenen Zyklus oder die vorangegangene Iteration beschränkt ist, werden kleine, aber häufige Verbesserungen vorgenommen, die sich hauptsächlich auf die Lösung spezifischer Projektprobleme konzentrieren. Der Schwerpunkt dieser Verbesserungen liegt oft nicht auf dem projektübergreifenden Lernen und der Institutionalisierung von Verbesserungen.

Wenn man sich ansieht, wie die Verbesserung des Testprozesses organisiert und verwaltet wird, liegt der Schwerpunkt wahrscheinlich weniger auf einer Testprozessgruppe auf organisatorischer Ebene als vielmehr auf der Selbstverwaltung von Teams innerhalb des Projekts. Diese Teams haben in der Regel das Mandat, den Testprozess im Rahmen des Projekts nach ihren Bedürfnissen zu ändern, was zu sehr maßgeschneiderten Prozessen führt. Einige Organisationen nutzen jedoch auch wöchentliche Test-Standup-Meetings, um die Dinge auf eine höhere und projektübergreifende Ebene zu bringen.

Da die Verbesserung des (Test-)Prozesses eher projektspezifisch ist, wird wahrscheinlich weniger Wert auf allgemeinere Fragen gelegt, die das Testen in der gesamten Organisation betreffen. Dies könnte zum Beispiel bedeuten, dass grundlegende Testprobleme nicht in vollem Umfang angegangen werden, weil sie außerhalb des projektspezifischen Kontexts liegen. Ein typisches Beispiel hierfür ist der Ansatz für das Testen bestimmter Qualitätsattribute, wie Performanz und Zuverlässigkeit. Diese Themen können von Iteration zu Iteration verschoben werden, weil sie oft mehr Kompetenzen und Ressourcen erfordern, als dem Projektteam zur Verfügung stehen. In diesen Bereichen ist es schwierig, ohne größere Investitionen einen wesentlichen nächsten Schritt zu machen. Probleme nur auf Projektebene zu lösen, kann auch leicht zu einer Suboptimierung führen und zum Verlust des Blicks für das große Ganze.

Im agilen Kontext kann die Bandbreite und Anzahl der zu berücksichtigenden alternativen Verbesserungsideen deutlich größer sein als bei nicht-agilen Lebenszyklusmodellen. Da alle Mitglieder einen Teil der Tests innerhalb des Projekts durchführen, können diese Ideen von jedem Teammitglied kommen. Dies legt einen stärkeren Schwerpunkt auf die Bewertung und Priorisierung von Verbesserungsvorschlägen, was eher eine Teamleistung als eine Aufgabe für eine einzelne Testprozessverbesserungs-Rolle sein kann. Das spezifische Testwissen eines Testprozessverbesserers kann auf Wunsch auch als Berater für das Team herangezogen werden.

Bei Projekten, die eine agile Methodik verwenden, sollten Sie nicht den Umfang der Testdokumentation erwarten, den Sie bei Projekten mit einem sequenziellen Lebenszyklus erwarten würden. Es kann ein einziges kombiniertes "Testdokument" geben, das die wesentlichen Elemente einer Testrichtlinie, einer Teststrategie und sogar eines übergeordneten Testkonzepts enthält. Wer den Testprozess verbessern will, sollte es vermeiden,

"Verbesserungsvorschläge" zu machen, die eine strengere und gründlichere Testdokumentation erfordern. Einer der wichtigsten agilen Grundsätze ist natürlich, dass eine Dokumentation nur dann erstellt wird, wenn ein klarer und eindeutiger Bedarf dafür besteht. Nicht nur die Testdokumentation wird weniger detailliert sein, sondern auch die Prozessdokumentation. Oft wird in agilen Umgebungen erfolgreich eine Strategie angewandt, die als "Formalisierung der Informalität" bezeichnet wird. Wenn etwas gut funktioniert, gibt es keinen Grund, dies um des TMMi willen zu ändern. Wenn es jedoch dokumentiert ist, kann es gelehrt und mit anderen geteilt werden, was oft von Vorteil ist. Mit der Strategie der "Formalisierung der Informalität" ist gemeint, dass man einen Prozess, der funktioniert, aber in gewisser Hinsicht informell ist, so lehren und dokumentieren kann, wie er durchgeführt wird. Da es sich um einen leichtgewichtigen Prozess handelt, wird in seiner Beschreibung nicht jedes mögliche Anwendungsszenario berücksichtigt. Diese Prozesse müssen daher durch Mentoring und On-the-Job-Hilfe unterstützt werden, insbesondere in der Phase der Ersteinführung. Das Ergebnis ist, dass die Testprozessreife in eine agile Organisation gebracht wird, während die agile Kultur erhalten bleibt, braucht man auch mehr - nicht weniger - Ausbildung. Ebenso wird sich bei einem Assessment der Schwerpunkt bei der Sammlung von Nachweisen von der Untersuchung von Artefakten auf die Durchführung von mehr Interviews verlagern.

Die Methoden, die bei der Anwendung von Agilität eingesetzt werden, um Verbesserungen des Testprozesses vorzuschlagen, konzentrieren sich in der Regel auf analytische Methoden zur Bewertung der Ursachen von Problemen, wie beispielsweise Ursache-Wirkungs-Diagramme. Dies sind besonders nützliche Methoden für die problemlösende Denkweise, die am Ende einer Iteration wichtig ist.

2 TMMi-Stufe 2 Gelenkt

2.1 Prozessbereich 2.1 Testrichtlinie und Teststrategie

Zweck des Prozessbereichs Testrichtlinie und Teststrategie ist die Entwicklung und Festlegung einer Testrichtlinie und einer organisations- oder programmweiten Teststrategie, in der die Testaktivitäten, z.B. Testarten und Testquadranten, eindeutig definiert sind. Um die Testleistung und den Wert der Testaktivitäten zu messen und Bereiche mit Verbesserungsbedarf aufzudecken, werden Testleistungsindikatoren eingeführt.

2.1.1 ¹SG1 Festlegung einer Testrichtlinie

Jede Organisation, die ein Projekt zur Testverbesserung in Angriff nimmt, sollte zunächst eine Testrichtlinie festlegen. In der Testrichtlinie werden die allgemeinen Testziele und die strategischen Ansichten des Unternehmens in Bezug auf Tests und Testexperten festgelegt. Es ist wichtig, dass die Testrichtlinie mit der allgemeinen Geschäfts-(Qualitäts-)politik des Unternehmens in Einklang steht. Testverbesserungen sollten sich an klaren Geschäftszielen orientieren, die wiederum in der Test(verbesserungs)richtlinie dokumentiert werden sollten. Eine Testrichtlinie ist notwendig, um eine gemeinsame Sichtweise auf das Testen und seine Ziele zwischen allen Beteiligten innerhalb einer Organisation zu erreichen. Diese gemeinsame Sichtweise ist erforderlich, um die Testaktivitäten (Prozessverbesserung) im gesamten Unternehmen aufeinander abzustimmen. Beachten Sie, dass Testziele nie ein eigenständiges Ziel sein sollten, sondern sich aus dem übergeordneten Ziel ableiten, funktionierende Software- und Produktqualität zu schaffen.

Dies gilt auch für Unternehmen, die agile Softwareentwicklung praktizieren. In vielen Unternehmen gibt es viele Diskussionen über die sich verändernde Rolle des Testens, die Unabhängigkeit des Testens, die Testautomatisierung und professionelle Tester in der agilen Softwareentwicklung. Diese und andere Punkte sind typischerweise Themen, die in einer Diskussion mit der Geschäftsleitung und anderen Beteiligten angesprochen und in einer Testrichtlinie dokumentiert werden sollten. Jedes Unternehmen, das ein Projekt zur Testverbesserung starten möchte, einschließlich derjenigen, die agil arbeiten, muss die geschäftlichen Treiber und Bedürfnisse für eine solche Initiative identifizieren und definieren. Warum sonst sollte man ein Verbesserungsprojekt starten? Wenn man sich die Zeit nimmt, die wahren Geschäftsbedürfnisse zu erfassen, kann man einen Kontext schaffen, um zu entscheiden, wo die Prioritäten für die (Test-)Verbesserung liegen sollen, z. B. auf welchen Prozessbereich. Beachten Sie, dass eine Testrichtlinie in der Regel ein einseitiges, schlankes Dokument, eine Webseite oder ein Wanddiagramm auf Organisationsebene ist und kein Dokument auf Projektebene.

Das spezifische Ziel des TMMi, die Erstellung einer Testrichtlinie einschließlich ihrer spezifischen Praktiken, ist in vollem Umfang auf Organisationen anwendbar, die agile Software-Entwicklung betreiben. Natürlich können die Elemente einer Testrichtlinie auch in eine Entwicklungsrichtlinie aufgenommen werden. Es gibt keine spezifische TMMi-Anforderung, dass es sich um ein separates Dokument handeln muss. Die Entwicklungspolitik in einer Organisation, die agile Softwareentwicklung anwendet, könnte beispielsweise Scrum als Managementrahmen, XP als Hauptmethode und die Einhaltung der agilen Werte als Hauptprinzip angeben. Ein weiterer wichtiger Grundsatz, der erwähnt werden könnte, ist, dass jeder im Team für alles verantwortlich ist; auch die Produktqualität liegt in der Verantwortung des Teams.

Es gibt jedoch eine wichtige Überlegung für Agilität in Bezug auf die Testrichtlinie und insbesondere definierte Test(verbesserungs)ziele. Auch wenn es Gesamtziele gibt, die sich auf die Verbesserung des Testprozesses in der Organisation beziehen, muss dies mit den einzelnen Projekten und agilen Teams, die für die Verbesserung ihres eigenen Prozesses verantwortlich sind, in Einklang gebracht werden. Die Herausforderung bei der agilen Prozessverbesserung besteht darin, die Verbesserung auf Organisationsebene zu lenken und zu gestalten, ohne dabei die Eigenverantwortung der einzelnen agilen Teams für ihren eigenen Prozess einzuschränken.

2.1.2 SG2 Festlegung einer Teststrategie

¹ SG = specific goal. Die englischen Abkürzungen des TMMi-Modells werden im Deutschen beibehalten.

Die Teststrategie folgt der Testrichtlinie und dient als Ausgangspunkt für die Testaktivitäten innerhalb von Projekten. Eine Teststrategie wird in der Regel entweder organisationsweit oder programmweit definiert. Eine typische Teststrategie basiert auf einer Produktrisikobewertung auf hoher Ebene und enthält eine Beschreibung der Testarten, Testquadranten und Teststufen, die durchgeführt werden sollen, z. B. Unit-, Abnahme-, Regressions- und Performanztests. Es reicht nicht aus, nur zu sagen, dass innerhalb einer Iteration Unittests und Abnahmetests durchgeführt werden. Es muss definiert werden, was unter Unit- und Abnahmetests zu verstehen ist, wie diese typischerweise durchgeführt werden und was ihre Hauptziele sind. Die Erfahrung zeigt, dass es weniger Überschneidungen zwischen den verschiedenen Testaktivitäten gibt, wenn eine Teststrategie definiert und befolgt wird, was wiederum zu einem effizienteren Testprozess führt.

Da die Testziele und Testansätze der verschiedenen Testarten und Teststufen aufeinander abgestimmt sind, verbleiben wahrscheinlich auch weniger Lücken, was zu einem effektiveren Testprozess und somit zu einer höheren Produktqualität führt. Um diese Angleichung zu erreichen, ist es sehr empfehlenswert, sowohl das agile Testen als auch alle nicht-agilen Tests, die im selben Projekt stattfinden, in einer übergreifenden Teststrategie zusammenzufassen. Gibt es jedoch getrennte agile und nicht-agile Projekte, gibt es normalerweise zwei Teststrategien (oder eine Teststrategie mit Hinweisen für die unterschiedliche Ausgestaltung der Tests je nach Lebenszyklus).

Eine Teststrategie ist ein wichtiges Dokument in einer agilen Umgebung. Sie definiert auf einer hohen Ebene die Tests, die in den agilen Teams (Iterationsteams) durchgeführt werden; welche Testarten, Testquadranten und Teststufen ausgeführt werden und auf einer hohen Ebene deren Ansatz. Das Dokument beschreibt, wie das Testen organisiert ist, z.B. welches Testen innerhalb der agilen Teams und welches Testen außerhalb stattfindet. Es definiert die Beziehung von agilen Teams zu den Teststufen, die außerhalb ihres Bereichs durchgeführt werden, z. B. Hardware-/Software-Integrationstests, Systemintegrationstests oder Betatests. Eine Teststrategie stellt sicher, dass alle am Testen Beteiligten das Gesamtbild des Testens verstehen.

Ein schlankes Teststrategie-Dokument ist oft eine gute Lösung für eine agile Organisation und ein Ausweg aus detaillierten (Projekt-)Testplänen. Während der Releaseplanung wird die vorhandene organisationsweite oder programmweite Teststrategie diskutiert und bestätigt oder eine abgeleitete Teststrategie speziell für das Projekt erstellt. Die bestätigte oder erstellte Teststrategie bietet einen Testrahmen, der alle Iterationen umfasst.

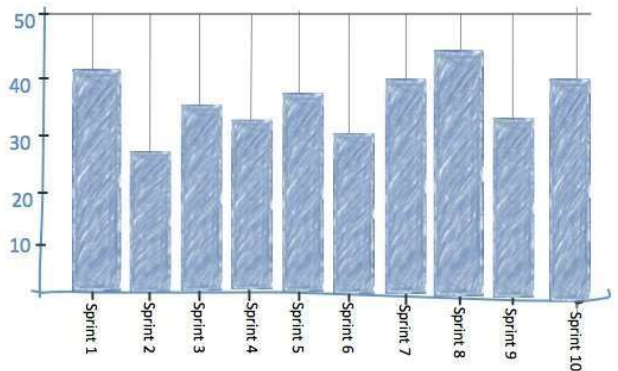
Das spezifische TMMi-Ziel Erstellen einer Teststrategie, einschließlich seiner spezifischen Praktiken, ist vollständig auf Organisationen anwendbar, die agile Softwareentwicklung anwenden. Beachten Sie, dass aktive "Verteilungs"-Aktivitäten, z. B. für das Teststrategiedokument, in der agilen Entwicklung weniger relevant sein können. Diejenigen, die ein Stakeholder sind, sollten bereits in frühere Diskussionen als Teil des Gesamtteamansatzes einbezogen worden sein. Eine Teststrategie ist jedoch nicht auf Teamebene, sondern auf einer höheren Ebene angesiedelt. Das gesamte Team arbeitet auf Teamebene, so dass ein Organisations- oder Programmdokument immer noch an die Teams und auch an andere Beteiligte verteilt werden muss, z. B. an diejenigen, die Testaktivitäten außerhalb des Teams durchführen, wenn ein hybrides Modell existiert.

2.1.3 SG3 Festlegung von Testleistungsindikatoren

Die in der Testrichtlinie festgelegten Geschäftsziele für die Testverbesserung müssen in eine Reihe von Schlüsselindikatoren für die Testleistung umgesetzt werden. Die Testrichtlinie und die zugehörigen Leistungsindikatoren geben eine klare Richtung vor und sind ein Mittel, um das erwartete und erreichte Niveau der Testleistung zu kommunizieren. Die Leistungsindikatoren müssen den Wert des Testens und der Testprozessverbesserung für die Beteiligten aufzeigen. Da Investitionen in die Prozessverbesserung eine langfristige Unterstützung durch das Management erfordern, ist es von entscheidender Bedeutung, den Nutzen eines Verbesserungsprogramms quantitativ zu messen, um die Motivation zu erhalten. Beachten Sie, dass es bei diesem TMMi-spezifischen Ziel um die Definition einer begrenzten Anzahl (z.B. 2 oder 3) von Testleistungsindikatoren geht. Es geht nicht darum, ein komplettes Messprogramm aufzustellen und zu implementieren, sondern vielmehr darum, einen Kernsatz von Indikatoren zu definieren, die Aufschluss darüber geben, wie sich der Wert des Testens im Laufe der Zeit und in verschiedenen Lieferumgebungen verändert.

Im Rahmen von Agilität wird der Schwerpunkt mehr auf Teamarbeit und Systemdenken liegen. Dies kann zu einer entsprechenden Ausweitung der Indikatoren auf das Team und das Gesamtsystem führen, anstatt sich nur auf die

Besonderheiten des Testens selbst zu beschränken. Auch die Indikatoren auf TMMi-Stufe 2 beziehen sich hauptsächlich auf die Endergebnisse der Iterationen. Beispiele hierfür wären: entgangene Fehler, Geschwindigkeit, Kundenzufriedenheit, Aufwand/Verschwendung, Prozentsatz der Testautomatisierung usw. Die Herausforderung besteht darin, eine geeignete Mischung von Indikatoren zu definieren, die sich auf den teambasierten Ansatz und das Systemdenken beziehen und gleichzeitig einen guten Hinweis auf die Leistungsergebnisse eines TMMi-basierten Testverbesserungsprogramms geben.



Das spezifische Ziel des TMMi, die Festlegung von Leistungsindikatoren für Tests, einschließlich der spezifischen Verfahren, ist uneingeschränkt anwendbar, auch wenn die ausgewählten und angewandten Leistungsindikatoren möglicherweise einen größeren Anwendungsbereich haben und breiter angelegt sind, als wenn sie sich nur auf Tests beziehen. Letzteres macht natürlich die Analyse und Interpretation der Leistungsindikatoren schwieriger. Die verwendeten Leistungsindikatoren werden möglicherweise nicht als Testleistungsindikatoren bezeichnet, sondern eher als Team- oder Systemleistungsindikatoren. Dies ist im Kontext von TMMi immer noch in Ordnung, solange sie testbezogene Elemente enthalten und zur Bewertung der Fortschritte bei der Testverbesserung verwendet werden.

2.2 Prozessbereich 2.2 Testplanung

Zweck der Testplanung ist es, auf der Grundlage der identifizierten Risiken und der festgelegten Teststrategie einen Testansatz zu definieren und fundierte Pläne für die Durchführung und Verwaltung der Testaktivitäten zu erstellen und zu pflegen.

Beachten Sie, dass der Schlüssel zu einer erfolgreichen Testplanung in der Vorüberlegung ("die Aktivität") liegt, nicht in der Definition des zugehörigen Testkonzepts ("das Dokument").

In agilen Lebenszyklen gibt es in der Regel zwei Arten von Planung, die Releaseplanung und die Iterationsplanung. Der Prozessbereich Testplanung auf TMMi-Stufe 2 konzentriert sich auf die testbezogenen Aktivitäten sowohl bei der Release- als auch bei der Iterationsplanung. Die Releaseplanung sieht die Veröffentlichung eines Produkts zu Beginn eines Projekts vor. Die Releaseplanung erfordert ein definiertes Product Backlog und kann die Verfeinerung größerer User Stories in eine Sammlung kleinerer Stories beinhalten. Die Releaseplanung bildet die Grundlage für einen Testansatz und ein Testkonzept, der alle Iterationen umfasst. Releasepläne sind sehr allgemein gehalten. Nachdem die Releaseplanung abgeschlossen ist, beginnt die Iterationsplanung für die erste Iteration. Die Iterationsplanung sieht bis zum Ende einer einzelnen Iteration voraus und befasst sich mit dem Iterations-Backlog.

Beachten Sie, dass der Prozessbereich Testplanung eine Reihe von spezifischen Praktiken umfasst. Das TMMi legt nicht fest, wann oder wie diese Praktiken durchzuführen sind. Sie besagt auch nicht, dass man nicht inkrementell planen kann. Der traditionelle Ansatz besteht darin, so viele Entscheidungen wie möglich im Voraus zu treffen, damit auch die damit verbundenen Kosten und der Zeitplan festgelegt werden können. Der Grund für diesen Ansatz war, dass man die Arbeit besser einschätzen und das Risiko einer schleichenden Ausweitung des Projektumfangs verringern konnte. Bei agilen Ansätzen wird in der Regel davon ausgegangen, dass eine kontinuierliche Verfeinerung des Plans auf der Grundlage der neuesten Informationen und der fortlaufenden Zusammenarbeit mit dem Kunden einen größeren Wert darstellt.

2.2.1 SG1 Risikobewertung durchführen

Ausführliche Tests sind unmöglich, und es müssen immer Entscheidungen getroffen und Prioritäten gesetzt werden. Dieses TMMi-Ziel ist daher auch für agile Projekte anwendbar. Bei agilen Projekten wird bei der Releaseplanung eine Bewertung des Produktrisikos auf hoher Ebene auf der Grundlage eines Dokuments zur Produktvision oder einer Reihe von User Stories auf hoher Ebene durchgeführt. Für jede Iteration wird eine detailliertere Produktrisikosituation auf der Grundlage der User Stories oder anderer Anforderungen für diese Iteration als Teil der Iterationsplanungssitzung durchgeführt. Der Produktrisikobewertungsprozess in einem agilen Projekt hat im Vergleich zu den Verfahren, die in einem traditionellen Projekt mit einem sequenziellen Lebenszyklusmodell

angewandt werden, ein sehr viel einfacheres Format. Ein Beispiel für eine leichtgewichtige Technik zur Bewertung von Produktrisiken ist der Risikopoker [Van Veenendaal].

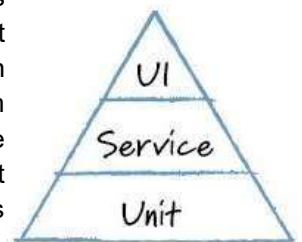
Bei der Releaseplanung geben Unternehmensvertreter, die die Funktionen der neuen Version kennen, einen Überblick über die zu entwickelnden Funktionen, und das gesamte Team, einschließlich der Tester, hilft bei der Ermittlung und Bewertung der Risiken.

Während der Iterationsplanung identifiziert und analysiert das agile Team die Produktrisiken auf der Grundlage der User Stories, die in der kommenden Iteration implementiert werden sollen. Vorzugsweise nehmen alle Mitglieder des agilen Teams und möglicherweise einige andere Stakeholder an der Sitzung zu den Produktrisiken teil. Das Ergebnis ist eine nach Prioritäten geordnete Liste von Produktrisiken, in der die kritischen Bereiche für das Testen aufgeführt sind. Dies wiederum hilft bei der Bestimmung des angemessenen Testaufwands, um jedes Risiko mit genügend Tests abzudecken, und bei der Anordnung dieser Tests in einer Weise, die die Effektivität und Effizienz der zu leistenden Testarbeit optimiert. Die geschätzten Aufgaben auf der Aufgabentafel können zum Teil nach dem Grad des mit ihnen verbundenen Produktrisikos priorisiert werden. Aufgaben, die mit höheren Risiken verbunden sind, sollten früher beginnen und einen höheren Testaufwand erfordern. Aufgaben, die mit geringeren Risiken verbunden sind, sollten später beginnen und weniger Testaufwand erfordern.

2.2.2 SG2 Festlegung eines Testansatzes

Es wird ein Testansatz definiert, um die identifizierten und priorisierten Produktrisiken zu mindern. Für eine bestimmte Iteration werden die zu testende Elemente und Funktionen während der Iterationsplanung festgelegt. Diese Aktivität basiert ebenfalls auf dem Ergebnis der Produktrisikositzung. Die priorisierte Liste der zu testenden Elemente bezieht sich in der Regel auf die User Stories, die in dieser Iteration getestet werden sollen. Die Merkmale beziehen sich typischerweise unter anderem auf die verschiedenen zu prüfenden Softwarequalitätsmerkmale. Während der Iteration können neue Produktrisiken auftauchen, die zusätzliche Tests erfordern. Themen wie neue Produktrisiken, die zusätzliche Tests erfordern, werden in der Regel in täglichen Standup-Meetings besprochen. Der Testansatz, der auf Iterationsebene definiert wird, um die Risiken zu mindern, kann z. B. eine zusätzliche Überprüfung der User Stories und der Akzeptanzkriterien, einen zum Risikoniveau proportionalen Testaufwand und die Auswahl geeigneter Testtechniken je nach Risikoniveau und -art umfassen. Der Testansatz auf Release-Ebene wird auf einer viel höheren Ebene angesiedelt sein und muss auf einer definierten Teststrategie auf Programm- oder Organisationsebene basieren. Häufig wird ein Testansatz im Team-/Projekt-Wiki festgehalten oder angezeigt.

Ein wichtiges Risiko, das bei der iterativen Entwicklung immer auftritt, ist das Regressionsrisiko. Der Testansatz muss festlegen, wie das Regressionsrisiko gehandhabt wird. Typischerweise wird dies durch die Erstellung eines spezifischen Regressionstestsatzes geschehen, der vorzugsweise automatisiert wird. In diesem Zusammenhang ist die Pyramide der Testautomatisierung [Cohn] hilfreich. Sie zeigt, wie der Wert der Automatisierung von Regressionstests maximiert werden kann, beginnend mit Unit-Tests auf der untersten Ebene der Pyramide bis hin zu Tests auf Service-Ebene. Das Testen der Benutzeroberfläche steht an der Spitze. Unit-Tests sind schnell und zuverlässig. Die Dienstebene ermöglicht das Testen der Geschäftslogik auf API- oder Dienstebene, wo Sie nicht durch die folgenden Faktoren beeinträchtigt werden die Benutzeroberfläche (UI). Je höher die Stufe, desto langsamer und brüchiger werden die Tests.



Eingangskriterien, die normalerweise Teil eines definierten Testansatzes sind (spezifische Praxis 2.3), sind für die agile Entwicklung wahrscheinlich nicht relevant. In der agilen Softwareentwicklung ist das Testen ein integraler Bestandteil des Teamprozesses und eine nahezu kontinuierliche Tätigkeit. Folglich gibt es keinen Bedarf für eine spezifische Checkliste oder einen Zugang, um festzustellen, ob mit dem Testen begonnen werden kann oder nicht. Dies gilt auch für eine Komponente, die innerhalb eines agilen Teams von einer Testphase (z. B. Unit-Tests) zu einer anderen (z. B. Abnahmetests) übergeht.

Testabschlusskriterien (spezifische Praxis 2.4) sind Teil der sogenannten Definition of Done (DoD). Es ist wichtig, dass die DoD spezifische testbezogene Kriterien enthält, z. B. für die Testabdeckung und die Produktqualität (Fehler). Die Iteration sollte zur Implementierung des vereinbarten Satzes von User Stories führen und die in der DoD definierten (Test-)Endekriterien erfüllen. Typischerweise werden Geschichten, die die Endekriterien nicht erfüllen, in

das Backlog gestellt und können in der nächsten Iteration bearbeitet werden. Natürlich gibt es in der Agilität keine Endekriterien für eine Komponente, die von einer Testphase in eine andere übergeht. Es gibt nicht nur eine Definition of Done auf Iterationsebene, sondern oft auch eine DoD auf Release-Ebene, die mehrere Iterationen umfasst. Die DoD auf Release-Ebene enthält wiederum typischerweise Kriterien zur Abdeckung und Produktqualität.

Die spezifische Praxis 2.5 Definition von Kriterien für die Unterbrechung und Wiederaufnahme ist für agile Lebenszyklen wahrscheinlich nicht relevant. Da das Testen ein integraler Bestandteil des agilen Softwareentwicklungsprozesses ist, wird es natürlich nicht als separate und unabhängige Aktivität von anderen Iterationsaktivitäten behandelt. Wenn es blockierende Probleme gibt, die als potenzielle oder tatsächliche Bedrohung für den Fortschritt des Testens angesehen werden können, werden diese im täglichen Standup-Meeting besprochen. In diesem Forum entscheidet das Team, welche Maßnahmen gegebenenfalls zu ergreifen sind, um die Probleme zu lösen. Daher sind formale Kriterien für die Unterbrechung und Wiederaufnahme nicht erforderlich und werden auch nicht definiert; diesbezügliche Probleme werden im Rahmen der normalen agilen Routine behandelt. Die agile Routine dient somit als alternative Praxis für diese spezifische Praxis.

2.2.3 SG3 Erstellung von Testschätzungen

Bei agilen Teams werden detaillierte Schätzungen für Tests während der Iterationsplanung vorgenommen. High-Level-(Test-)Schätzungen werden während der Releaseplanung und möglicherweise auch bei Backlog-Verfeinerungssitzungen vorgenommen. Alle Schätzungen werden natürlich als Teamschätzungen vorgenommen, die den gesamten Aufwand für die Lieferung jeder Story enthalten. Es muss sichergestellt werden, dass die Testaktivitäten in den Schätzungssitzungen tatsächlich berücksichtigt werden. Dies kann dadurch geschehen, dass die Testaktivitäten als separate Aufgaben identifiziert werden und anschließend jede einzelne Testaufgabe geschätzt wird, oder durch die Schätzung jeder User Story, wobei die durchzuführenden Tests explizit gemacht und somit berücksichtigt werden. Ein Tester, der Teil des agilen Teams ist, muss an den Schätzungssitzungen teilnehmen. Planning Poker oder Shirt-Size sind typische Schätztechniken, die bei der agilen Softwareentwicklung eingesetzt werden.



Die für die nächste Iteration zu erledigende Arbeit wird in der Regel durch die User Storys definiert. Die User Storys müssen klein sein, damit sie schätzbar sind. Schätzbar ist eines der von INVEST [Wake] definierten Kriterien für User Storys und gilt für User Storys, die Teil einer Iteration sein sollen. Während der Iterationsplanung werden agile Teams typischerweise testbezogene Aufgaben identifizieren und definieren. Die Testaufgaben werden zusammen mit den anderen Entwicklungsaufgaben auf einem Task-Board festgehalten. Dieser Aufgabensatz bildet die Grundlage für die Schätzung der Iteration. Der Satz der definierten Aufgaben für die kommende Iteration dient als eine Art Arbeitsunterbrechungsstruktur (wie in sequenziellen Projekten).

Bei der Releaseplanung sind die User Stories oder Epics in der Regel eher auf hoher Ebene definiert und noch nicht in spezifische Aufgaben unterteilt. Dies macht natürlich die Schätzungen schwieriger und ungenauer. Wie bereits erwähnt, wird bei agilen Projekten keine Projektstruktur als Grundlage für Schätzungen erstellt, aber auf Release-Ebene kann ein einfaches Diagramm hilfreich sein, um das zu entwickelnde Produkt zu visualisieren und damit den Arbeitsumfang angemessen festzulegen..

Obwohl ein agiles Team seine Schätzungen relativ informell vornimmt, sollten die Gründe für die Schätzungen klar sein (d. h. welche Faktoren berücksichtigt werden). Diskussionen auf der Grundlage von Begründungen tragen zu einer höheren Genauigkeit der Schätzungen bei. In der Regel konzentriert sich die Schätzung in agilen Projekten auf den Umfang (unter Verwendung von Story Points) oder den Aufwand (unter Verwendung von idealen Manntagen als Schätzeinheit). Die Kosten werden in agilen Projekten normalerweise nicht als Teil der Schätzungssitzungen behandelt. Dieses spezifische TMMi-Ziel und seine spezifischen Praktiken sind daher vollständig anwendbar, mit Ausnahme der spezifischen Praxis 3.2 Definition des Testlebenszyklus. Eine der Grundlagen der iterativen und agilen Softwareentwicklung besteht darin, in kleinen Stücken zu arbeiten. Daher sind die identifizierten Aufgaben in der Regel detailliert genug, um als Grundlage für die (Test-)Schätzung zu dienen. In der agilen Softwareentwicklung

ist es daher nicht erforderlich, auch einen Lebenszyklus für Testaktivitäten zu definieren, der als zusätzliche Grundlage für die Schätzung dient.

2.2.4 SG4 Entwicklung eines Testkonzepts

Wiederholung der Bemerkung, dass es bei der Testplanung um das Vordenken ("die Aktivität") und nicht um die Definition des zugehörigen Testkonzepts ("das Dokument") geht. Bei Agilität werden die meisten Testplanungsaktivitäten, wie sie in diesem spezifischen TMMi-Ziel definiert sind, während der Release- und Iterationsplanung durchgeführt werden. Die Ergebnisse dieser Aktivitäten werden jedoch typischerweise nicht in einem Testkonzept dokumentiert, insbesondere nicht bei der Iterationsplanung, wo sie auf dem Task Board festgehalten werden könnten.

Da das Testen als integraler Bestandteil der Release- und Iterationsplanung durchgeführt wird, enthalten die daraus resultierenden "Zeitpläne" auch Testaktivitäten. Anstelle eines detaillierten Zeitplans, wie er bei sequenziellen Lebenszyklen entwickelt wird, ist der Zeitplan in einem agilen Projekt eher eine Anordnung von User Stories (Backlog-Elementen) und Aufgaben, die die Prioritäten der Releases und Iterationen widerspiegelt, z. B. auf der Grundlage der gewünschten Lieferung des Geschäftswerts. Mindmaps werden hier oft als unterstützende Technik eingesetzt. Die Aufgabentafel spiegelt die Iterationsprioritäten wider. Es wird also kein expliziter (Test-)Zeitplan aufgestellt; es wird erwartet, dass klare Release- und Iterationsprioritäten für die User Stories bzw. die durchzuführenden Aufgaben, einschließlich der Testaufgaben, definiert sind.

Auf Projektebene ist die Besetzung von Testressourcen Teil des Teamaufbaus; der Bedarf an Testressourcen oder Ressourcen mit mehreren Qualifikationen wird im Vorfeld ermittelt. Wenn sich Projekte ändern oder wachsen, kann man leicht die Gründe für die ursprüngliche Auswahl einer Person für ein bestimmtes Team/Projekt vergessen, zu denen oft ein spezifischer Bedarf an Kompetenzen oder Erfahrung in Bezug auf das Team/Projekt gehört. Dies ist eine gute Grundlage, um die Anforderungen an die Kompetenzen der Mitarbeiter aufzuschreiben und zu begründen, warum sie für das Team/Projekt ausgewählt wurden. Sobald die agilen Teams definiert sind, steht die Testbesetzung mehr oder weniger fest. Während der Iterationsplanung kann die Identifizierung der (zusätzlichen) Ressourcen und Kompetenzen, z. B. für nicht-funktionale Tests, die für die Durchführung der Tests in einer Iteration benötigt werden, bei Bedarf diskutiert werden, um sicherzustellen, dass das Team über ausreichende Testressourcen, Kenntnisse und Kompetenzen verfügt, um die erforderlichen Tests durchzuführen.

Der Scrum Master sollte sicherstellen, dass der Product Owner bei Bedarf Input zum Testen liefert, z.B. durch Beantwortung von Fragen und Gespräche über User Stories. Der Product Owner sollte ausreichend verfügbar sein, was wiederum im Vorfeld organisiert werden sollte.

Eine erste Projektrisikositung sollte Teil der Release- und Iterationsplanung sein. Die Identifizierung (und das Management) weiterer Projektrisiken während der Iteration ist Teil der täglichen Standup-Sitzungen und wird in der Regel mit Hilfe eines Hindernisprotokolls dokumentiert. Es ist wichtig, dass auch Testprobleme im Hindernisprotokoll vermerkt werden. Die Hindernisse sollten bei den täglichen Standup-Meetings besprochen werden, bis sie behoben sind.

Obwohl meistens kein spezifisches und detailliertes Testkonzept entwickelt und dokumentiert wird, ist die spezifische Praxis 4.5 Erstellung des Testkonzepts in einem agilen Kontext immer noch relevant. Das Ergebnis der Diskussionen, die im Rahmen der Testplanung stattfinden, wird jedoch wahrscheinlich in einer leichtgewichtigen Form, möglicherweise einer Mindmap, festgehalten.

2.2.5 SG5 Einholung der Zustimmung zum Testkonzept

Im Rahmen von Agilität ist der Prozess zur Entwicklung und Festlegung eines Testansatzes und Testkonzepts eine teambasierte Übung, die möglicherweise von einem Testexperten (einem der Teammitglieder) geleitet wird. Die Produktqualität liegt in der Verantwortung des Teams. Vorausgesetzt, das Team folgt dem richtigen Prozess, ist das Engagement für den (Test-)Ansatz und den (Test-)Plan bereits ein implizites Ergebnis der Release- und Iterationsplanung, da es sich um eine Teamleistung handelt. Dies ist natürlich ein großer Unterschied zur Arbeitsweise in einer traditionellen Umgebung, in der der Tester typischerweise das Testkonzept erstellt und danach ein explizites Commitment einholen muss.

In agilen Projekten muss das agile Team (einschließlich des Product Owners) die priorisierte Liste der Produktrisiken

und die durchzuführenden Testmaßnahmen zur Risikominderung verstehen und sich darauf einigen. Das Verständnis und die Zustimmung können zum Beispiel durch eine kurze Präsentation mit anschließender Diskussion während der Release- oder Iterationsplanung erreicht werden, in der dem Team die Produktrisiken, der Testansatz und seine Gründe erläutert werden.

Während der Schätzungssitzung (siehe SG 3 Erstellen von Testschätzungen) wird der Arbeitsaufwand geschätzt. Die (Test-)Ressourcen für ein Team werden durch das Setup des agilen Teams festgelegt. Die User Stories, die geschätzt und zur Entwicklung ausgewählt werden, nehmen die verfügbaren Ressourcen als Ausgangspunkt (Constraint). Daher ist auch hier der Abgleich von Arbeits- und Ressourcenniveau keine sinnvolle Aktivität. Die spezifische Praxis 3.2 Abgleich von Arbeit und Ressourcen ist daher eine Praxis, die typischerweise in einem agilen Kontext nicht relevant ist. Tägliche Stand-up-Meetings werden genutzt, um Ressourcenprobleme während der Iteration anzusprechen und die entsprechenden Ressourcen sofort (neu) zuzuweisen oder ein Deliverable aus einer Iteration zu entfernen, um es in einer zukünftigen Iteration oder Releaseplanung abzugleichen.

2.3 Prozessbereich 2.3 Testüberwachung und Teststeuerung

Zweck der Testüberwachung und Teststeuerung ist es, einen Überblick über den Testfortschritt und die Produktqualität zu geben, damit geeignete Korrekturmaßnahmen ergriffen werden können, wenn der Testfortschritt erheblich vom Plan und die Produktqualität erheblich von den Erwartungen abweicht.

Nach dem Agilen Manifest und den dazugehörigen Grundsätzen gibt es einige Dinge, die sich bei der Überwachung und Steuerung in Agilen Projekten grundlegend von denen in traditionellen Projekten unterscheiden. Obwohl Überwachung und Steuerung wesentliche Elemente eines agilen Projekts sind, bedeutet dies nicht, dass das Festhalten an einem starren Plan das Ziel ist, im Gegenteil, sowohl das Manifest als auch die Prinzipien sprechen davon, Veränderungen willkommen zu heißen. Die Testüberwachung und -steuerung könnte in einem agilen Kontext als Bereitstellung von Best Practices zur kontinuierlichen Anpassung des Plans interpretiert werden, um ihn auf dem neuesten Stand zu halten, wie es die agilen Ansätze empfehlen.

Aus der Perspektive der Testüberwachung und Teststeuerung bedeutet dies, dass wir nicht planorientiert vorgehen, sondern unseren Fortschritt und die Testergebnisse ständig überprüfen und unseren Plan und Ansatz gegebenenfalls anpassen, wenn neue Produktrisiken erkennbar werden. Der Testkonzept, wie auch immer er erfasst wird, ist ein lebendiges Gebilde und muss ständig überprüft und aktualisiert werden, wenn neue Informationen oder Rückmeldungen eintreffen. Agile Projekte müssen auch das "große Ganze" im Auge behalten und sowohl auf höherer (Release-)Ebene als auch auf Iterationsebene überwachen und steuern.

Da das Testen ein Prozess ist, der vollständig in den Gesamtprozess des agilen Teams integriert ist, ist auch die Testüberwachung und -steuerung ein integraler Bestandteil der allgemeinen Überwachungs- und Kontrollmechanismen des agilen Teams. Infolgedessen sind die Tester nicht wie bei traditionellen Projekten einem Testmanager unterstellt, sondern dem Team.

Da sich der Prozessbereich Testplanung auf TMMi-Stufe 2 sowohl auf die Release- als auch auf die Iterationsplanung konzentriert, gilt dies auch für die Testüberwachung und -steuerung. Es wird erwartet, dass die Testüberwachung und -steuerung sowohl auf der Planungs- als auch auf der Freigabeebene durchgeführt wird.

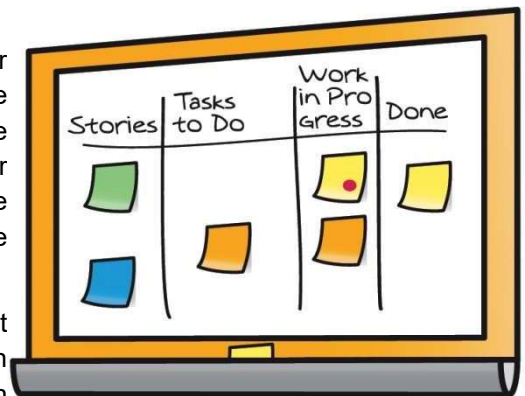
2.3.1 SG1 Überwachung des Testfortschritts gegenüber dem Plan

Tester in agilen Teams verwenden verschiedene Methoden, um den Testfortschritt zu überwachen und aufzuzeichnen, z. B. den Verlauf von Testaufgaben und Stories auf dem agilen Task Board und auf Burndown-Diagrammen. Diese können dann mit Hilfe von Medien wie Wiki-Dashboards und E-Mails im Dashboard-Stil sowie mündlich bei Stand-up-Meetings an den Rest des Teams weitergegeben werden. Teams können Burndown-Diagramme sowohl zur Verfolgung des Fortschritts über die gesamte Version als auch innerhalb jeder Iteration verwenden. Ein Burndown-Diagramm zeigt in der Regel den Fortschritt im Vergleich zur erwarteten Geschwindigkeit und zum Backlog der zu implementierenden Funktionen (Leistung des Teams). Wenn die Ressourcen der Testumgebung knapp und wichtig sind, z. B. für nicht-funktionale Tests, werden manchmal spezielle Burndown-Diagramme verwendet, um die Nutzung der Ressourcen der Testumgebung im Vergleich zu den bei der Iterationsplanung vereinbarten Werten zu überwachen. Eine weitere gängige Praxis ist die Verfolgung von Testumgebungsproblemen über die Aufgabentafel, z. B. durch die Verwendung von Aufklebern mit der Aufschrift

"blockierte Testumgebung" auf den Story-Karten oder durch die Erstellung einer separaten Spalte auf der Tafel, in der alle durch Umgebungen blockierten Storys warten, bis sie freigegeben werden. Es geht darum, die Auswirkungen einer Testumgebung, die den Fortschritt blockiert, sichtbar zu machen.

Um eine sofortige, detaillierte visuelle Darstellung des aktuellen Status des gesamten Teams, einschließlich des Teststatus, zu erhalten, können Teams agile Task Boards verwenden. Die Story Cards, Entwicklungsaufgaben, Testaufgaben und andere Aufgaben, die während der Iterationsplanung erstellt wurden, werden auf dem Task Board festgehalten, wobei häufig farblich abgestimmte Karten verwendet werden, um den Aufgabentyp zu bestimmen. Während der Iteration wird der Fortschritt durch das Verschieben dieser Aufgaben auf dem Task Board in Spalten wie "zu tun", "in Arbeit" und "erledigt" verwaltet. Agile Teams können Tools verwenden, um ihre Story Cards auf agilen Task Boards zu verwalten, wodurch Dashboards und Statusübersichten automatisiert werden können. Einige Teams erstellen jedoch keine spezifischen Aufgaben für die einzelnen Aktivitäten, sondern verwenden einfach die Story Card und vermerken

Kommentare in dieser Karte für Tests mit Verweis auf agile Tools oder Wikis, in denen die Tests dokumentiert werden können. Die Testaufgaben auf der Aufgabentafel beziehen sich in der Regel auf die für die User Stories definierten Abnahmekriterien. Wenn Skripte zur Testautomatisierung, manuelle Skripttests und explorative Tests für eine Testaufgabe den Status "bestanden" erreichen, wird die Aufgabe in die Spalte "erledigt" der Aufgabentafel verschoben.



Die Definition of Done dient als Ausstiegs Kriterium, an dem der Fortschritt gemessen wird. Die DoD sollte sich auch auf die Testaktivitäten beziehen und zeigt alle Kriterien auf, die erfüllt sein müssen, bevor das Testen einer User Story als "erledigt" bezeichnet werden kann. Beachten Sie, dass die Testkriterien im Zusammenhang mit den Testaufgaben nur einen Teil dessen ausmachen, was das Team zu erledigen vereinbart als Definition of Done. Die Definition of Done-Kriterien werden in der Regel auf mehreren Ebenen angewandt, z. B. auf Iterations- und Release-Ebene. Das gesamte Team überprüft den Status der Aufgabentafel regelmäßig, oft während der täglichen Stand-up-Meetings, um sicherzustellen, dass die Aufgaben in einem akzeptablen Tempo über die Tafel laufen. Wenn eine Aufgabe (einschließlich Testaufgaben) nicht oder zu langsam vorankommt, löst dies eine Diskussion im Team aus, in der die Probleme analysiert werden, die den Fortschritt dieser Aufgaben behindern.

Am täglichen Stand-up-Meeting nehmen alle Mitglieder des agilen Teams teil, auch die Tester. Bei diesem Treffen teilen sie dem Rest des Teams ihren aktuellen Status und den tatsächlichen Fortschritt mit. Alle Probleme, die den Testfortschritt blockieren könnten, werden in den täglichen Stand-up-Meetings mitgeteilt, so dass das gesamte Team über die Probleme informiert ist und entsprechend handeln kann. Auf diese Weise wird auch das Risikomanagement in diese täglichen Besprechungen integriert. Jedes Projektrisiko, einschließlich der Risiken für das Testen, z. B. die mangelnde Verfügbarkeit von Testumgebungen, kann während des täglichen Stand-ups kommuniziert und angesprochen werden. (Hinweis: Die Überwachung von Produktrisiken ist Teil der Überwachung der Produktqualität und wird daher im Folgenden unter dem spezifischen Ziel SG 2 Überwachung der Produktqualität anhand von Plan und Erwartungen behandelt). Tägliche Standup-Meetings für das tägliche Aufgabenmanagement und agile Teampraktiken im Zusammenhang mit Aufgabenkurskorrekturen sind bewährte Techniken, die der Absicht der TMMi-spezifischen Praktiken im Prozessbereich Testüberwachung und -steuerung entsprechen.

Der Meilenstein-Review im Rahmen von Agilität findet am Ende einer Iteration statt. Die Ergebnisse der Tests, z. B. anhand der Definition of Done, sind Teil des Iterationsreviews. Es werden Demos mit den Stakeholdern organisiert, um den Geschäftswert und die Qualität des gelieferten Produkts zu diskutieren.

Der Product Owner vertritt die Stakeholder bei der Iterationsplanung, den Iterationsbesprechungen (Demos) und Retrospektiven. Der Product Owner wird durch Diskussionen über das Product Backlog einbezogen und gibt während der gesamten Iteration Feedback und Input für die Ausarbeitung von User Stories und den Entwurf von Tests. Andere Stakeholder werden am Ende jeder Iteration im Rahmen des Iterationsreviews einbezogen. Eine spezifische Überwachung der Stakeholder-Beteiligung ist nicht erforderlich, da die Vertretung der Stakeholder durch einen Product Owner in der agilen Arbeitsweise verankert ist. Beachten Sie, dass die aktive Einbindung des Product Owners in agilen Projekten ein entscheidender Erfolgsfaktor ist, aber in der Praxis manchmal recht schwierig zu realisieren ist. Wenn letzteres der Fall ist, wird dies beim täglichen Stand-up gemeldet und diskutiert und als

Projektrisiko (siehe oben) behandelt, das die Effektivität und Effizienz des gesamten Teams beeinträchtigt.

2.3.2 SG2 Überwachung der Produktqualität anhand von Plan und Erwartungen

Für die Überwachung der Produktqualität werden weitgehend die gleichen Mechanismen wie für die Fortschrittsüberwachung verwendet (siehe SG1 [Überwachung des Testfortschritts gegenüber dem Plan](#)). Bei Agilität liegt der Schwerpunkt bei der Überwachung der Produktrisiken auf der Überprüfung der Liste der Produktrisiken in regelmäßigen Sitzungen und nicht auf der Überprüfung der detaillierten Dokumentation der Risiken. Neu identifizierte Produktrisiken oder geänderte Produktrisiken, z. B. als Ergebnis von Sondierungstests, werden besprochen und die erforderlichen Testmaßnahmen werden vereinbart. Der Status der verschiedenen Produktrisiken wird in der Regel in Form von Diagrammen auf dem Dashboard dargestellt.

Ein bewährtes Verfahren besagt, dass eine Funktion erst dann als fertiggestellt gilt, wenn sie erfolgreich in das System integriert und getestet wurde. In einigen Fällen werden in regelmäßigen Abständen Härtings- oder Stabilisierungsiterationen durchgeführt, um verbleibende Fehler und andere Formen von technischen Schulden zu beseitigen. Agile Teams verwenden defektbasierte Metriken, die denen ähneln, die in traditionellen Entwicklungsmethoden erfasst werden, wie z. B. Test-Pass/Fail-Raten, Fehlerentdeckungsraten, gefundene und behobene Fehler, um die Produktqualität zu überwachen und zu verbessern. Die Anzahl der während einer Iteration gefundenen und behobenen Fehler sowie die Anzahl der ungelösten Fehler, die möglicherweise Teil des Backlogs für die nächste Iteration werden, sollten während der täglichen Stand-up-Meetings überwacht werden. Um die allgemeine Produktqualität zu überwachen und zu verbessern, verwenden viele agile Teams auch Umfragen zur Kundenzufriedenheit, um Rückmeldungen darüber zu erhalten, ob das Produkt die Erwartungen der Kunden erfüllt.

Testabschlusskriterien, z. B. für die Testabdeckung und die Produktqualität (Fehler), sind Teil der Definition of Done (DoD). Die Einhaltung der vereinbarten Endekriterien wird in der Regel durch das Task Board überwacht, wobei eine Story nur dann als "erledigt" bezeichnet werden kann, wenn sie die DoD-Kriterien erfüllt.

Das tägliche Stand-up-Meeting ist der Mechanismus, mit dem fast kontinuierlich Produktqualitätsprüfungen durchgeführt werden. Der Meilenstein für die Überprüfung der Produktqualität ist bei Agilität der Abschluss einer Iteration. Es werden Demos mit den Beteiligten organisiert, um den Geschäftswert und die Qualität des gelieferten Produkts zu besprechen. Die Produktqualität wird anhand der definierten DoD-Qualitätskriterien überprüft und validiert.

Im Anschluss an den Prozessbereich Testplanung sind spezifische Praktiken zur Überwachung der Kriterien für Eintritt, Aussetzung und Wiederaufnahme wahrscheinlich nicht relevant. Weitere Informationen finden Sie unter dem spezifischen Ziel 2 Festlegen eines Testansatzes im Prozessbereich Testplanung, wo erläutert wird, warum diese Kriterien in einer agilen Umgebung in der Regel nicht relevant sein dürften.

2.3.3 SG3 Verwaltung von Korrekturmaßnahmen bis zur Schließung

Agile Teams würden sehr schnell Probleme wie Abweichungen von den Erwartungen in einem Burn-Down-Diagramm und/oder mangelnden Fortschritt von (Test-)Aufgaben und Geschichten auf dem agilen Task Board bemerken. Diese und andere Probleme, z. B. Probleme, die den Testfortschritt blockieren, werden während der täglichen Stand-up-Meetings (siehe oben) kommuniziert, so dass das gesamte Team über die Probleme informiert ist. Dies löst dann eine teambasierte Diskussion aus, in der die Probleme analysiert werden. Das Ergebnis könnte sein, dass die Baseline aktualisiert werden muss (z. B. Streichung einer oder mehrerer User Stories aus dem Iterations-Backlog), die Definition of Done vielleicht zu streng ist oder Änderungen an der Arbeitsweise vorgenommen werden sollten. Das Team entscheidet gemeinsam mit dem Product Owner über die zu treffenden Korrekturmaßnahmen. In Fällen, in denen das Iterations-Backlog geändert wird, legt der Product Owner dem Team ein aktualisiertes Iterations-Backlog vor, und die aus dem Iterations-Backlog entfernten Elemente werden in die nächste Iteration mitgenommen und während der Iterationsplanung diskutiert.

Das Management von Korrekturmaßnahmen in agilen Projekten liegt in erster Linie in der Verantwortung des selbstorganisierenden Teams. Das Team kann geeignete Korrekturmaßnahmen festlegen und umsetzen oder Probleme als "Hindernisse" an den Scrum Master eskalieren. Der Scrum Master hat in der Regel die Aufgabe, das Team dabei zu unterstützen, Probleme bis zum Abschluss zu managen. Typische Veranstaltungen zur Besprechung und Verwaltung von Korrekturmaßnahmen sind tägliche Stand-ups und retrospektive Meetings.

Korrekturmaßnahmen, die vereinbart wurden, können als "Aufgaben" oder "Backlog Items" über das Product Backlog oder (innerhalb der Iteration) über das Task Board bis zur Schließung verwaltet werden.

2.4 Prozessbereich 2.4 Testentwurf und Testdurchführung

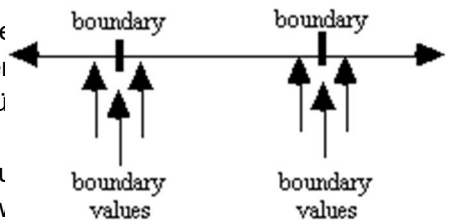
Der Zweck von Testentwurf und Testdurchführung ist die Verbesserung der Testprozessfähigkeit während des Testentwurfs und der Testdurchführung durch die Erstellung von Testentwurfsspezifikationen, die Anwendung von Testentwurfstechniken, die Durchführung eines strukturierten Testdurchführungsprozesses und das Management von Testzwischenfällen bis zum Abschluss.

Obwohl das zugrundeliegende Ziel ("Risikominderung und Testen der Software") für ein agiles und ein traditionelles sequenzielles Projekt dasselbe ist, ist der Ansatz, wie getestet wird, in der Regel sehr unterschiedlich. Bei Agilität sind Flexibilität und die Fähigkeit, auf Veränderungen zu reagieren, wichtige Ausgangspunkte. Außerdem sind Testanalyse und -design, Testrealisierung und Testausführung keine aufeinander folgenden Testphasen, sondern werden parallel, überlappend und iterativ durchgeführt. Der Detaillierungsgrad der Testdokumentation ist ein weiterer wichtiger Unterschied. Typischerweise werden in agilen Projekten häufiger erfahrungsbasierte und defektbasierte Techniken verwendet, obwohl auch spezifikationsbasierte Testentwurfstechniken weiterhin anwendbar sind und verwendet werden können. Durch die stärkere Betonung von Unit- und Integrationstests sind White-Box-Techniken wie Statement- und Entscheidungstests ebenfalls sehr viel beliebter. Ein letzter großer Unterschied ist die Höhe des Regressionsrisikos, das mehr Regressionstests auf den verschiedenen Teststufen erforderlich macht. Idealerweise sind Regressionstests weitgehend automatisiert. Es gibt viele Unterschiede, aber letztlich geht es im Kontext des Prozessbereichs Testdesign und -durchführung immer um die Erstellung von Tests, die Minderung von Produktrisiken, die Durchführung von Tests und das Auffinden von Fehlern.

2.4.1 SG1 Durchführung von Testanalyse und Testentwurf unter Verwendung von Testentwurfstechniken

Bei Agilität sind Testanalyse und -design sowie die Testdurchführung sich gegenseitig, die in der Regel während einer Iteration parallel laufen. In sequentieller Testanalyse, indem die Tester die Testbasis, z. B. die Anforderungen, überprüfen nach deren Erstellung bewerten. In agilen Projekten sind die Tester Teil eines erstellt und verfeinert. Während der Entwicklung der Anforderungen werden hat einschließlich der Akzeptanzkriterien für jede User Story. Diese Kriterien werden von Geschäftsvertretern, Entwicklern und Testern definiert. In der Regel wird die einzigartige Perspektive des Testers genutzt, um die User Story verbessern, indem er fehlende Details identifiziert und sie testbar macht. Ein Tester kann einen Beitrag leisten, indem er den Geschäftsvertretern offene Fragen über die User Story und ihre Akzeptanzkriterien stellt und Möglichkeiten zum Testen der User Story vorschlägt. Die Testanalyse ist somit keine explizite separate Aktivität, sondern eine implizite Aktivität, die Tester als Teil ihrer Rolle innerhalb der kollaborativen User-Story-Entwicklung durchführen. Auf der Grundlage der Analyse der User Storys werden die Testbedingungen ermittelt. Aus der Testperspektive wird die Testbasis analysiert, um zu sehen, was getestet werden könnte - dies sind die Testbedingungen. Testbedingungen (manchmal auch als Testsituationen bezeichnet) sind im Grunde eine Identifizierung von "Dingen", die getestet/abgedeckt werden müssen [Black, Van Veenendaal]. Sofern die definierten Akzeptanzkriterien detailliert und klar genug sind, können sie durchaus die Rolle der traditionellen Testbedingungen übernehmen. Die Testbedingungen werden dann in Tests umgesetzt, die die definierten und vereinbarten Akzeptanzkriterien abdecken müssen. Oft ist es auch von Vorteil, die Testanalyse auf einer höheren Ebene als nur auf der Ebene der User Stories durchzuführen. Zum Beispiel die Analyse eines Features oder eines Epos oder einer Sammlung von Stories, um Testbedingungen zu identifizieren, die abstrakter sind als die auf User-Story-Ebene und auch mehrere User-Stories umfassen. Spezifikationsbasierte Testentwurfstechniken sind in der Regel hilfreich bei der Ableitung von Testbedingungen aus User Stories und Akzeptanzkriterien.

Bei Agilität werden diese Testentwurfstechniken jedoch meist eher implizit als explizit verwendet, wobei die Tester aufgrund ihrer Erfahrung die Techniken beherrschen und in der Lage sind, sie im jeweiligen Kontext flexibel einzusetzen. Die Testbedingungen werden in einem leichtgewichtigen Format dokumentiert, im Gegensatz zu der eher traditionellen Arbeitsweise, bei der sie als Teil eines Testentwurfs-Spezifikationsdokuments dokumentiert



werden. Manchmal, insbesondere in Projekten, in denen exploratives Testen weit verbreitet ist, werden die Testbedingungen nur durch Brainstorming ermittelt und als Testideen dokumentiert (als Teil von Testchartas). Die Definition von Testbedingungen ist auch die Grundlage für die Herstellung der horizontalen Rückverfolgbarkeit (siehe unten) und die Verwaltung der Abdeckung des (automatisierten) Regressionstestsets, wobei automatisierte Tests entwickelt werden, um bestimmte Testbedingungen abzudecken.

Bei der Anwendung des Test-First-Prinzips im Rahmen von Agilität werden die Tests, die die Testbedingungen abdecken, vor oder zumindest parallel zur Entwicklung des Codes ermittelt (und möglicherweise automatisiert). Für automatisierte Unit-Tests kann ein Ansatz wie Test-Driven Development in Betracht gezogen werden. Für höhere Teststufen sind Behavior-Driven Development (BDD) und Acceptance Test-Driven Development (ATDD) beliebte agile Entwicklungsansätze, die auch das Testen unterstützen. Sowohl BDD als auch ATDD sind eng mit der Testautomatisierung verbunden.

Bei den meisten manuellen Tests werden die Tests im Laufe der Testdurchführung durch das Team identifiziert und verfeinert. Die Tests werden in den meisten Fällen nicht so detailliert dokumentiert wie in traditionellen Projekten, sondern eher in Form von Testideen unter Verwendung von Sondierungstests. Für komplexe und kritische Bereiche kann ein traditionellerer Ansatz für den Testentwurf und die Entwicklung von Testfällen unter Verwendung formaler Testentwurfstechniken der beste Weg sein, um die Risiken abzudecken. Allerdings ist auch bei diesem Ansatz der Umfang der Dokumentation im Vergleich zum Testen in einer traditionellen sequenziellen Lebenszyklsumgebung begrenzt. Die Priorisierung der Tests folgt der Priorisierung der User Story, die sie abdecken. Die Priorisierung der User Stories basiert auf dem Geschäftswert; die höchste Priorität bezieht sich auf den höchsten Geschäftswert. Es ist wichtig, dass Tests erstellt werden, um funktionale und nicht-funktionale Risiken abzudecken, aber insbesondere bei Agilität auch speziell, um das Regressionsrisiko abzudecken.

Es werden spezifische Testdaten identifiziert, die zur Unterstützung der Testbedingungen und der Testdurchführung erforderlich sind. Im Gegensatz zu traditionellen Umgebungen werden die benötigten Testdaten bei Agilität jedoch in der Regel nicht zuerst als Teil eines Testspezifikationsdokuments festgelegt. Vielmehr werden, sofern die notwendigen Werkzeuge und/oder Funktionalitäten vorhanden sind, die Testdaten sofort erstellt, so dass mit der Ausführung manueller Tests fast sofort begonnen werden kann. Für automatisierte Tests müssen die Daten jedoch in der Regel im Voraus spezifiziert werden.

Die Nachvollziehbarkeit zwischen den Anforderungen, den Testbedingungen und den Tests muss hergestellt und aufrechterhalten werden. Die Teams müssen deutlich machen, dass sie die verschiedenen User Stories und Akzeptanzkriterien im Rahmen ihrer Tests abgedeckt haben. Ein formales Anforderungsmanagement-Tool ist zwar nicht unbedingt erforderlich, aber das Team braucht Unterstützung, um seine Anforderungen so zu organisieren und zu verwalten, dass jeder Anforderung Identifikatoren zugewiesen werden können, damit diese Identifikatoren verwendet werden können, um sicherzustellen, dass die Tests gemäß den vereinbarten Kriterien durchgeführt werden. In vielen agilen Organisationen bilden die User Stories die Grundlage für die Entwicklung einer Reihe von zugehörigen Akzeptanzkriterien und anschließend von Tests. Sobald diese Tests festgelegt sind, werden die Tests selbst oft zu den detaillierten vereinbarten Anforderungen. Dieser Ansatz kann ausreichen, um die Absicht der TMMi-Praxis zum Anforderungsmanagement in Bezug auf die horizontale Rückverfolgbarkeit zu erreichen.

2.4.2 SG2 Testrealisierung durchführen

Bei der Testrealisierung geht es darum, alles bereitzustellen, was für die Durchführung von Tests erforderlich ist. In der Regel wird die Entwicklung von Testdokumentation, z. B. von Testverfahren, zur Unterstützung der Testdurchführung auf ein Minimum reduziert. Stattdessen werden automatisierte (Regressions-)Testskripte entwickelt und priorisiert. Auch die Vorbereitung von Regressionstests beginnt so früh wie möglich, parallel zu anderen Testaktivitäten.

In einer agilen Umgebung sind detaillierte Testverfahren keine gängige Praxis. Bei der Arbeit in einem sachkundigen Team werden die Tests höchstwahrscheinlich auf einer viel höheren Abstraktionsebene dokumentiert. Dies reicht denjenigen, die die Tests durchführen, aus, da erwartet wird, dass sie über das erforderliche Maß an Fach- und technischem Wissen verfügen, um die Tests durchzuführen. Diejenigen, die die Tests durchführen, arbeiten innerhalb des Teams, so dass sie ein besseres Verständnis dafür haben, was und wie es kodiert ist, und wie die Funktionalität für den Zweck geeignet ist. Da das Niveau der Änderungen innerhalb und zwischen den Iterationen in der Regel hoch ist, würde die Entwicklung detaillierter Testverfahren auch viel Wartungsarbeit verursachen. In der

Regel werden viele Tests mit Hilfe von Sondierungstests durchgeführt. Beim explorativen Testen werden keine detaillierten Testverfahren entwickelt, sondern eher einseitige Testchartas auf hohem Niveau, die Testideen und Anleitungen für den Tester beschreiben. Die für die Durchführung von Tests erforderlichen spezifischen Testdaten müssen natürlich im Vorfeld erstellt werden.

Viele agile Teams verwenden automatisierte Tests. Typische Ansätze sind Test-Driven Development (TDD), Behavior-Driven Development (BDD) und Acceptance Test-Driven Development (ATDD). Mit einem oder mehreren dieser Ansätze werden im Rahmen der Testrealisierung automatisierte Testskripte erstellt.

Die kontinuierliche Integration ist eine wichtige Praxis für agile Projekte. Das Testen ist eine parallele und vollständig integrierte Aktivität, die vom Team durchgeführt wird. Es handelt sich nicht um eine unabhängige, separate Aktivität oder Phase. Die spezifische Praxis 2.3 Spezifizieren Sie den Smoketest wird daher in einem agilen Team irrelevant. Es besteht keine Notwendigkeit und es macht auch keinen Sinn, einen speziellen formalen Eingangstest zu spezifizieren und durchzuführen. Falls jedoch einige Tests, z. B. Systemintegrationstests oder Hardware-/Software-Integrationstests, außerhalb des Bereichs des agilen Teams durchgeführt werden, kann ein Eingangstest festgelegt werden, um die Qualität des Outputs des agilen Teams zu bewerten und festzustellen, ob das gelieferte Produkt für ihre Tests bereit ist.

In agilen Umgebungen wird kein spezifischer Zeitplan für die Testdurchführung erstellt und daher ist die spezifische Praxis 2.4 Testausführungsplan entwickeln wahrscheinlich nicht relevant. Es besteht ein hohes Maß an Flexibilität hinsichtlich der Reihenfolge, in der die Tests während einer Iteration ausgeführt werden, obwohl dies durch die Priorität der verschiedenen User Stories bestimmt wird. Die verschiedenen auszuführenden Tests werden während der Iterationsplanung identifiziert und als Testaufgaben über das agile Task Board verwaltet.

2.4.3 SG3 Testdurchführung ausführen

Die in der Iteration zu entwickelnden Softwareelemente müssen natürlich getestet werden. Gemäß dem agilen Manifest wird dies, wie auch bei den anderen Zielen in diesem Prozessbereich, in der Regel mit weitaus geringerer Dokumentationsintensität geschehen als in einem traditionellen sequenziellen Projekt. Oft werden keine detaillierten Testverfahren und Testprotokolle erstellt. Die Tests, insbesondere die Regressionstests, sollten so weit wie möglich und sinnvoll automatisiert werden.

In einem agilen Projekt wird die Software mindestens täglich geliefert, und das Testen ist ein integraler Bestandteil des agilen Entwicklungsprozesses, bei dem die Arbeit in gegenseitiger Zusammenarbeit erfolgt, um ein Qualitätsprodukt zu liefern. Es gibt kein dediziertes und separates Entwicklungs- und Testteam. Daher ist ein spezifischer Eingangstest, wie er in der spezifischen Praxis 3.1 Durchführung eines Smoketests definiert ist, in einer reinen Agile-Umgebung wahrscheinlich nicht von Bedeutung. Wenn jedoch einige Tests außerhalb des agilen Teams durchgeführt werden (was häufig der Fall ist), wird ein Eingangstest typischerweise von dem Testteam durchgeführt, das für die Tests außerhalb des agilen Teams verantwortlich ist. Beachten Sie, dass es Aktivitäten gibt, die abgeschlossen sein müssen, bevor die Abnahmetests für eine bestimmte User Story innerhalb des agilen Teams beginnen. Diese Aktivitäten sind typischerweise Teil der Definition of Done, z. B. wenn der vereinbarte Umfang der Unit-Tests abgeschlossen und bestanden wurde.

Die Testdurchführung erfolgt in Übereinstimmung mit den bei der Iterationsplanung festgelegten Prioritäten. Einige Tests können unter Verwendung eines dokumentierten Testverfahrens durchgeführt werden, aber in der Regel werden viele Tests unter Verwendung von explorativen und sitzungsbasierten Tests als Rahmen durchgeführt. Beim explorativen Testen finden Testentwurf und Testdurchführung gleichzeitig statt und werden von einer vorbereiteten Testcharta geleitet. Eine Testcharta enthält Testziele und Testbedingungen, die während einer zeitlich begrenzten Testsitzung abzudecken sind. Beim explorativen Testen dienen die Ergebnisse der letzten Tests als Leitfaden für die nächsten Tests.

Mit der iterativen Entwicklung steigt auch der Bedarf an organisierten und strukturierten Regressionstests. Dies geschieht manchmal manuell, wird aber vorzugsweise mit automatisierten Regressionstests und unterstützenden Tools durchgeführt. Regressionstests, insbesondere Unit- und Integrationstests, sind oft Teil eines kontinuierlichen Integrationsprozesses. Kontinuierliche Integration ist eine wichtige Praxis innerhalb der agilen Softwareentwicklung. Dabei handelt es sich im Grunde um einen automatisierten Build- und Testprozess, der mindestens täglich stattfindet und frühzeitig und schnell erkannt wird. Die kontinuierliche Integration ermöglicht es, regelmäßig automatisierte

Regressionstests durchzuführen und dem Team eine schnelle Rückmeldung über die Qualität des Codes und die erreichte Codeabdeckung zu geben.

Die beim Testen gefundenen Vorfälle können protokolliert und gemeldet werden. In der Regel wird in agilen Projekten darüber diskutiert, ob tatsächlich alle gefundenen Vorfälle protokolliert werden sollten. Insbesondere in Fällen, in denen das Team an einem anderen Standort arbeitet, müssen die Tester mit den Entwicklern sprechen, und Vorfälle/Fehler, die sofort behoben und in den nächsten Build aufgenommen werden können, müssen möglicherweise nicht protokolliert, sondern nur behoben werden. In der agilen Arbeitsweise werden im Allgemeinen nicht alle gefundenen Vorfälle protokolliert und entsprechend verwaltet. Wie bereits erwähnt, werden viele einfach informell von einem Teammitglied (z. B. einem Tester) gemeldet und sofort von dem Entwickler behoben, der den Fehler verursacht hat. Dabei gehen zwar einige Daten verloren, aber gleichzeitig wird der Overhead reduziert. Manche Teams protokollieren nur Vorfälle, die Iterationen überstehen, manche protokollieren sie, wenn sie nicht sofort behoben werden können, manche protokollieren nur Vorfälle mit hoher Priorität. Wenn nicht alle gefundenen Vorfälle protokolliert werden, müssen Kriterien vorhanden sein, um zu bestimmen, welche Vorfälle protokolliert werden sollen und welche nicht. Denken Sie daran, dass die Absicht darin besteht, den Fehler korrekt zu beheben, und nicht darin, einen Fehler zu protokollieren.

Ein Beispiel für ein solches Kriterium: "Wenn ein Fehler vor dem nächsten Daily Standup und Daily Build behoben werden kann, muss er nicht dokumentiert werden. Jeder aufgetretene Fehler, der nicht innerhalb eines Tages behoben werden kann oder einen Stakeholder außerhalb des Teams betrifft, sollte dokumentiert und erfasst werden."

Manchmal werden Vorfälle auf dem agilen Task Board protokolliert, entweder als Post-It auf einer Aufgabe oder als separate Aufgabe, um zu verdeutlichen, dass sie die Fertigstellung einer Story und ihrer Aufgaben blockieren. Es ist akzeptabel, Post-it-Notizen auf einem Task-Board zu verwenden, um die Defekte zu verwalten, und dies ist oft die gewählte Methode für co-located Teams. Das agile Team wird anschließend an den Vorfällen arbeiten und sie bis zum Abschluss verwalten (siehe spezifisches Ziel 4 dieses Prozessbereichs). Einige entscheiden sich dafür, die gefundenen Vorfälle in einem Tool zu protokollieren und zu dokumentieren, z. B. in einem Defektmanagement- oder Defektverfolgungs-Tool. Ein solches Tool sollte dann so schlank wie möglich eingesetzt werden und keine Elemente von Testvorfallberichten erzwingen, die keinen oder einen zu geringen Mehrwert haben.

Auch in agilen Teams gilt es als gute Praxis, während der Testdurchführung Daten zu protokollieren, um festzustellen, ob die getestete(n) Aufgabe(n) die definierten Abnahmekriterien erfüllen und tatsächlich als "erledigt" bezeichnet werden können. Dies gilt auch bei der Anwendung von erfahrungsbasierten Techniken wie dem explorativen Testen. Beispiele für Informationen, die dokumentiert werden sollten, sind die Testabdeckung (wie viel wurde abgedeckt und wie viel muss noch getestet werden), Beobachtungen während des Testens, z. B. ob das System und die zu testende Benutzergeschichte stabil zu sein scheinen, eine Risikoliste (welche Risiken wurden abgedeckt und welche gehören noch zu den wichtigsten), gefundene Fehler und andere Probleme und mögliche offene Fragen.

Die protokollierten Informationen sollten in einer Form von Statusmanagement-Tools (z. B. Testmanagement-Tools, Aufgabenmanagement-Tools, Task Board) erfasst und/oder zusammengefasst werden, so dass das Team und die Stakeholder den aktuellen Status aller durchgeführten Tests leicht nachvollziehen können. Natürlich muss das Team nicht nur den Status der einzelnen Testaufgaben kennen, sondern auch den Gesamtstatus der User Story. Nichts ist aus Sicht der Story vollständig, wenn nicht alle damit verbundenen Aktivitäten abgeschlossen sind, nicht nur die Tests.

2.4.4 SG4 Verwaltung von Testvorfällen bis zur Schließung

Wie bereits erwähnt, werden in agilen Umgebungen oft nicht alle gefundenen Vorfälle protokolliert. Dieses spezifische Ziel gilt nur für die Vorfälle, die protokolliert werden und somit bis zum Abschluss verwaltet werden müssen. Im Prinzip ist die Verwaltung von Vorfällen bei Agilität einfach. Wenn der Vorfall auf dem agilen Task Board protokolliert wird, entweder als Aufkleber auf einer Aufgabe oder als separate Aufgabe, wird er als Defekt visualisiert, der die Fertigstellung einer Story und ihrer Aufgaben blockiert. Die standardmäßige agile Arbeitsweise wird wie bei jedem anderen Hindernis oder jeder anderen Aufgabe, die den Fortschritt blockiert, angewandt. Er wird in den Stand-up-Meetings besprochen und dem Team zur Lösung zugewiesen, woraufhin er von der Aufgabentafel verschwindet.

Wenn die Entscheidung getroffen wird, die gefundenen Vorfälle auf eine weitere Iteration zu verschieben, werden

sie lediglich zu einer weiteren gewünschten Option (oder Änderung) für das Produkt. Fügen Sie sie dem Backlog hinzu und priorisieren Sie sie entsprechend. Wenn die Priorität hoch genug angesetzt ist, werden sie vom Team in der nächsten Iteration aufgegriffen. Beachten Sie, dass die Priorität in der Regel durch das Geschäft definiert wird, aber manchmal muss Vorfällen, die sich auf sogenannte technische Schulden beziehen, eine gewisse Priorität eingeräumt werden. Einige Teams haben spezielle Iterationen für die Beseitigung von Vorfällen/Fehler im Zusammenhang mit technischen Schulden. Dies ist jedoch keine von Agilität empfohlene Standardpraxis, da Schulden regelmäßig zurückgezahlt werden sollten.

Als Beispiel dafür, wie Vorfälle in einer agilen Umgebung gehandhabt werden können, stellen Sie sich zunächst vor, dass sich das Team in der Mitte einer Iteration befindet. Während der Arbeit an einer der User Stories stellt das Team fest, dass es diese nicht fertigstellen kann, weil eines der Abnahmekriterien nicht erfüllt ist, z. B. aufgrund eines während der Entwicklung eingeführten Fehlers. Agile Teams neigen dazu, diese Art von Defekten so schnell wie möglich in der aktuellen Iteration zu beheben, da sie sonst unerledigte Arbeit weiterführen würden (gleichbedeutend mit einer schlanken Inventur). Diese Art von Vorfällen wird auf dem Task Board visualisiert und verwaltet.

Desweiteren: Nehmen wir an, dass alle Stories für die Iteration bis zum Ende der Iteration gemäß der Definition of Done abgeschlossen sind. Wenn jedoch während des Review-Meetings bei der Demonstration einer Story etwas auffällt, das sich nicht so verhält, wie der Kunde es sich wünscht. Bedenken Sie, dass das unerwartete Verhalten nicht mit dem Unternehmen besprochen wurde, bevor die Iteration begann. In diesem Fall funktioniert die erzeugte Softwarefunktionalität, und was wahrscheinlich getan werden muss, ist, den Vorfall in eine zusätzliche Story zu übersetzen und sie dem Produkt-Backlog hinzuzufügen.

Es ist eine gute Praxis, die Anzahl der während einer Iteration gefundenen und behobenen Fehler sowie die Anzahl der ungelösten Fehler, die möglicherweise Teil des Backlogs für die nächste Iteration werden, während des täglichen (Stand-up) Meetings und auch bei retrospektiven Meetings zu überwachen.

In einer traditionellen, sequenziellen Lebenszyklusumgebung gibt es oft ein spezielles Konfigurationskontrollgremium (CCB), das zusammenkommt, den Vorfall prüft und darüber entscheidet, wie er zu behandeln ist. Bei Agilität wird diese Rolle vom befähigten Team übernommen. Natürlich spielt auch der Vertreter des Unternehmens, z. B. der Product Owner, eine wichtige Rolle, wenn es darum geht, über die Behandlung eines Vorfalls zu entscheiden. Die Bearbeitung von Vorfällen ist ein Overhead, der den Fokus und den Aufwand von anderen Dingen abzieht. Der Overhead wird umso größer, je mehr wir versuchen, einen Prozess zu formalisieren, um sie aufzulisten, zu qualifizieren, zu ordnen und schließlich zu planen. Ein so komplexer Prozess für das Management von Zwischenfällen ist reine Verschwendung. Besser ist es, einige Grundprinzipien für die Behandlung und das Management von Vorfällen in agilen Teams zu umreißen und entsprechend zu handeln.

2.5 Prozessbereich 2.5 Testumgebung

Der Zweck der Testumgebung ist es, eine angemessene Umgebung, einschließlich Testdaten, zu schaffen und zu erhalten, in der es möglich ist, die Tests in einer handhabbaren und wiederholbaren Weise durchzuführen.

Mit dem Prozessbereich Testumgebung wird eine adäquate Testumgebung inklusive generischer Testdaten aufgebaut und gepflegt, in der es möglich ist, die Tests überschaubar und wiederholbar durchzuführen. Natürlich ist auch in einem Softwareentwicklungsprojekt, das den Agilen Lebenszyklus nutzt, eine adäquate Testumgebung unabdingbar. Aufgrund der kurzen Zykluszeiten einer Iteration muss die Testumgebung hoch stabil und verfügbar sein. Probleme in der Testumgebung haben immer sofort Auswirkungen auf den Fortschritt und die Ergebnisse der Iteration. Die ordnungsgemäße Verwaltung der Konfiguration und der Änderungen an der Testumgebung und den Testdaten ist daher ebenfalls von größter Bedeutung.

Testumgebungen gibt es in vielen Varianten, die unter anderem vom zu testenden System abhängen, aber typischerweise erfolgt die Spezifikation, Implementierung und das Management der Testumgebung parallel zum agilen Softwareentwicklungsprojekt. Die Implementierung einer Testumgebung nimmt oft viel Zeit in Anspruch und kann sehr komplex sein, was es fast unmöglich macht, diese Aufgabe innerhalb der begrenzten Zykluszeit einer Iteration durchzuführen [Van der Aalst und Davis]. Außerdem sind verschiedene Interessengruppen und Ingenieure beteiligt, die normalerweise ein agiles Team bilden. Die agilen Methoden wurden zur Unterstützung der Softwareentwicklung entwickelt, nicht für die Entwicklung und Verwaltung von Testumgebungen oder Infrastrukturen. Sie sind einfach weniger geeignet, und die Art und Weise, wie die Testumgebung oder die Infrastrukturentwicklung

und -verwaltung durchgeführt wird, bleibt weitgehend dieselbe wie in einer traditionellen Umgebung, auch wenn es bei der Anwendung von Agilität einige Änderungen im Prozess geben kann.

2.5.1 SG1 Entwicklung von Anforderungen an die Testumgebung

Die Spezifikation der Anforderungen an die Testumgebung erfolgt in einem frühen Stadium des Projekts. Die Anforderungsspezifikation wird überprüft, um ihre Korrektheit, Eignung, Durchführbarkeit und genaue Darstellung einer "realen" Betriebsumgebung sicherzustellen. Eine frühe Anforderungsspezifikation hat den Vorteil, dass mehr Zeit für den Erwerb und/oder die Entwicklung der erforderlichen Testumgebung und Komponenten wie Simulatoren, Stubs oder Treiber zur Verfügung steht.

In einigen agilen Entwicklungsprojekten wird eine sogenannte erste Iteration (Iteration 0) durchgeführt, in der die Anforderungen an die Testumgebung erhoben und spezifiziert werden.



2.5.2 SG2 Anforderungen an die Testumgebung umsetzen

Manchmal findet (ein Teil) der Implementierung in der ersten Iteration statt, zusätzlich zu anderen Aktivitäten, z. B. Schulung, Risikobewertung des Produkts auf hoher Ebene und Festlegung der Definition of Done. Auf diese Weise soll erreicht werden, dass bereits in den ersten Iterationen der Softwareentwicklung eine (teilweise) funktionierende Testumgebung vorhanden ist. Technische Umgebungsmaßnahmen oder -themen können Teil des Product Backlogs sein.

Die Definition der Testumgebung (einschließlich der generischen Testdaten) könnte wie bei einem sequentiellen Lebenszyklus mit einem vollständigen Plan beginnen, doch ist es oft besser, so bald wie möglich mit der Implementierung zu beginnen und bei Beginn der ersten Iteration eine erste Version der Testumgebung zur Verfügung zu haben.

In der Tat ist es schwierig, in Testumgebungen generisch zu sein. In einigen Bereichen werden Stubs und Treiber schnell durch Service-Virtualisierung ersetzt, was in der Regel eine viel schnellere Implementierung der Testumgebung ermöglicht. In einigen agilen Projekten ist die Bereitstellung von Testumgebungen weitgehend automatisiert. Testumgebungen können innerhalb von Minuten oder Stunden in Betrieb genommen werden, statt in Tagen. Auch die Virtualisierung von Maschinen, Servern und Diensten wird heutzutage immer häufiger eingesetzt. Auch die Cloud kann bei der Bereitstellung der benötigten Umgebungen helfen, was wiederum den Zugriff beschleunigt.

2.5.3 SG3 Management und Steuerung von Testumgebungen

Das Management und die Steuerung der Testumgebung finden während eines agilen Projekts statt. Manchmal muss das agile Team die Testumgebung selbst einrichten und pflegen, aber meistens wird dies von einer separaten Einheit außerhalb des agilen Teams erledigt. Wenn die Aktivitäten vom agilen Team selbst durchgeführt werden, setzt dies natürlich voraus, dass das Team über ausreichende technische Kenntnisse verfügt, um diese Aufgaben ausführen zu können. Es bedeutet auch, dass die Aufgaben Teil des agilen Prozesses werden, d. h. sie müssen in einer Planungssitzung behandelt, auf das Task Board gesetzt und möglicherweise während der täglichen Stand-up-Meetings besprochen werden, falls es Probleme gibt.

Als Hauptschlussfolgerung für den Prozessbereich Testumgebung sind die spezifischen Ziele und Praktiken weiterhin anwendbar und ändern sich im Wesentlichen nicht. Das Einzige, was sich ändert, ist der Zeitpunkt im Lebenszyklus.

3 TMMi-Stufe 3 Definiert

3.1 Prozessbereich 3.1 Organisation der Tests

Der Zweck des Prozessbereichs Organisation der Tests besteht darin, eine Gruppe hochqualifizierter Personen zu identifizieren und zu organisieren, die für das Testen verantwortlich ist. Neben dem Testen kümmert sich die Testgruppe auch um die Verbesserung des Testprozesses und der Testprozess-Assets der Organisation, basierend auf einem gründlichen Verständnis der Stärken und Schwächen des aktuellen Testprozesses und der Testprozess-Assets der Organisation.

3.1.1 SG 1 Aufbau einer Testorganisation

Die Organisation der Tests ist ein oft missverständlicher Prozessbereich. Viele lesen dies so, dass das TMMi eine unabhängige Testgruppe oder sogar eine Abteilung verlangt, die unabhängige Tests durchführt. Obwohl dies auch eine Möglichkeit ist, gibt es auch andere Organisationsmodelle, die den TMMi-Anforderungen entsprechen. Neben der unabhängigen Testorganisation, die Tests vorbereitet und durchführt, ist auch das Testkompetenzzentrum eine Option. Typische Aufgaben und Verantwortlichkeiten eines so genannten Testkompetenzzentrums sind die Einrichtung, das Management und die Verbesserung von Testprozessen sowie die Bereitstellung von Ressourcen für Projekte mit Testerfahrung, Wissen und Kompetenzen. Ein Testkompetenzzentrum hat in der Regel die Verantwortung für Testmethoden und -prozesse. Beachten Sie, dass ein Tester im Tagesgeschäft typischerweise Teil des agilen Teams ist, aber darüber hinaus ist es üblich, dass ein Tester einer Testorganisation angehört, wie dem Testkompetenzzentrum oder der Testgilde.

Eine Testorganisation erfordert eine Führung in den Bereichen, die sich auf Test- und Qualitätsfragen beziehen. Die Mitarbeiter einer solchen Gruppe werden als Testspezialisten bezeichnet. Die Option des Testkompetenzzentrums passt gut zur Agilität und stellt sicher, dass das Testen als Disziplin erhalten bleibt und als solche ernst genommen wird. Natürlich gibt es agile Vordenker, die nicht für die Beibehaltung spezieller Disziplinen sind. Während dies für einige eine bevorzugte Option sein mag, erfordert die Durchführung von TMMi, dass Testen als Disziplin behandelt und beibehalten wird. Natürlich sollte dies nur dann geschehen, wenn es aus geschäftlicher Sicht sinnvoll ist, und wenn dies nicht der Fall ist, ist TMMi wahrscheinlich nicht der bevorzugte Weg zur Verbesserung.

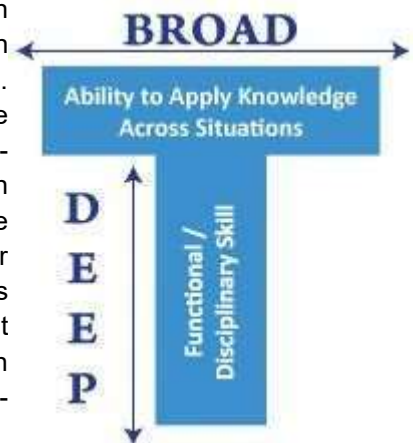
Bei Agilität kann die Testorganisation die Form einer Testgilde annehmen. Eine Testgilde ist typischerweise eine informelle, sich selbst verwaltende Gruppe von Testern, die Mitglied in verschiedenen agilen Teams sind. Wichtige Aktivitäten einer Testgilde können Wissensaustausch, gruppenweises Lernen, Trendbeobachtung usw. sein. Die Mitgliedschaft in einer Gilde erfolgt auf freiwilliger Basis. In der Praxis trifft sich eine Testgilde oft alle zwei Wochen. Eine Testgilde kann auch mit formelleren Aufgaben betraut werden, z. B. mit der Festlegung von Testkarrieren, der Organisation von Schulungen und der Ermittlung, Planung und Umsetzung von Testprozessverbesserungen. Es ist wichtig, dass die Testgilde weitgehend informell bleibt und sich selbst verwaltet. Eine wichtige Voraussetzung für den Erfolg einer Testgilde ist, dass ihren Mitgliedern Zeit zugewiesen wird, die sie für die verschiedenen Aktivitäten der Testgilde aufwenden müssen.

Unabhängige Tester sind oft effektiver bei der Fehlersuche. Es ist eine Herausforderung, Unabhängigkeit und Agilität in Einklang zu bringen. Einige agile Teams behalten völlig unabhängige, separate Testteams bei und setzen in den letzten Tagen jedes Sprints Tester nach Bedarf ein. Dadurch kann die Unabhängigkeit gewahrt werden, und diese Tester können eine objektive, unvoreingenommene Bewertung der Software vornehmen. Allerdings führen Zeitdruck, mangelndes Verständnis für die neuen Funktionen des Produkts und Beziehungsprobleme mit Geschäftsinteressenten und Entwicklern oft zu Problemen bei diesem Ansatz. Eine andere Möglichkeit ist eine unabhängige, separate Testorganisation, bei der die Tester zu Beginn des Projekts langfristig agilen Teams zugewiesen werden, so dass sie ihre Unabhängigkeit bewahren und gleichzeitig ein gutes Verständnis für das Produkt und enge Beziehungen zu den anderen Teammitgliedern aufbauen können. Darüber hinaus kann die unabhängige Testorganisation über spezialisierte Tester außerhalb der agilen Teams verfügen, die an langfristigen und/oder Iterations-unabhängigen Aktivitäten arbeiten, wie z. B. der Entwicklung von automatisierten Testtools, der Durchführung von nicht-funktionalen Tests, der Erstellung und Unterstützung von Testumgebungen und -daten sowie der Durchführung von Teststufen, die möglicherweise nicht gut in einen Sprint passen (z. B. Systemintegrationstests). Beachten Sie, dass es auch Möglichkeiten gibt, innerhalb eines agilen Teams unabhängige Tests durchzuführen.

Solange eine andere Person als der Ersteller der Assets oder des Codes das Produkt ebenfalls testet, handelt es sich immer noch um Unabhängigkeit. Daher können Entwickler den Code der anderen über Code-Reviews und/oder Unit-Tests testen, oder Business-Analysten, die zum Testen aus der Benutzerperspektive beitragen, indem sie zum Beispiel verschiedene Personas verwenden. Zu guter Letzt kann die Unabhängigkeit durch eine Organisations- und Verantwortungsstruktur organisiert werden, aber wahrscheinlich ist es vor allem die richtige kritische Einstellung.

3.1.2 SG 2 Festlegung von Testfunktionen für Testspezialisten

Testen wird als geschätzter Beruf und Disziplin angesehen. Das agile Team benötigt Testkenntnisse und -Kompetenzen. Der Testspezialist wird diese bereitstellen und auch andere Teammitglieder bei der Durchführung von Testaktivitäten anleiten. Testfunktionen und Testkarrierepfade sind definiert und werden durch ein Testschulungsprogramm unterstützt. Die erforderlichen Kenntnisse und Kompetenzen eines typischen agilen Testers unterscheiden sich von denen in einer traditionellen Organisation. Der Tester benötigt natürlich nach wie vor "traditionelle" Testkenntnisse und -Kompetenzen, aber darüber hinaus auch Kenntnisse und Kompetenzen zur Testautomatisierung und zur Durchführung von Aufgaben außerhalb des Testens, z. B. Skripting oder Requirements Engineering. Da ein agiler Tester in einem Team arbeitet, sind Soft Skills ebenso wichtig wie zunehmende technische Kompetenzen. Der agile Tester ist ein sogenannter T-Shape-Tester, was sich in der Beschreibung der Testfunktionen widerspiegeln soll. Die Testorganisation besteht aus Personen, die die Kompetenzen und die Motivation haben, gute Tester in einem agilen Kontext zu sein. Sie werden einer bestimmten Testfunktion zugewiesen. Besonders in einem agilen Kontext ist es wichtig, da das gesamte Team Testaktivitäten durchführen wird, andere, nicht testspezifische Funktionsbeschreibungen auszuarbeiten mit Erwartungen hinsichtlich Testaktivitäten, Rollen und erforderlichen Testkenntnissen und -kompetenzen.



3.1.3 SG 3 Einrichtung von Testkarrierepfaden

Testkarrierepfade werden eingerichtet, um Testfachleuten die Möglichkeit zu geben, ihr Wissen, ihre Kompetenzen, ihren Status und ihre Vergütung zu verbessern und so ihre Karriere voranzutreiben. Auf der Grundlage der definierten Testkarrierepfade werden persönliche Testkarriereentwicklungspläne für jedes Mitglied der Testorganisation entwickelt und gepflegt. Dieses spezifische Ziel und die dazugehörigen spezifischen Praktiken sind auch in einem agilen Kontext voll anwendbar. Da bei Agilität der Schwerpunkt auf dem menschlichen Aspekt liegt, ist dies eindeutig ein wichtiges spezifisches Ziel. Das agile Team braucht Tester, die ihre anspruchsvolle Aufgabe verstehen, andere Teammitglieder coachen können und motiviert sind. Viele dieser Aspekte werden durch Testkarrierepfade angesprochen.

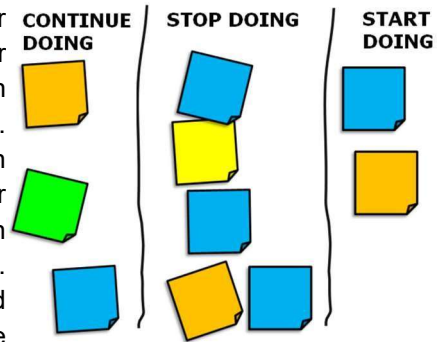
Die tatsächliche Definition von Testkarrierepfaden unterscheidet sich jedoch typischerweise in einer agilen Organisation von denen in einer Organisation, die einem sequenziellen Lebenszyklus folgt. Während in einer traditionellen Organisation die Testkarrierepfade oft auf vertikales Wachstum ausgerichtet sind (vom Tester zum Testmanager), konzentrieren sich die Testkarrierepfade in einer agilen Organisation eher auf horizontales Wachstum (vom Junior-Tester zum Senior-Tester und vielleicht zum Testcoach). Es besteht in der Regel ein geringerer Bedarf an verschiedenen hierarchischen Funktionen und Rollen, wie z. B. Testmanagement, und ein höherer Bedarf, sich an den Rollen eines Product Owner oder Scrum Master zu orientieren. Beachten Sie, dass viele der traditionellen Testmanagementaufgaben weiterhin relevant sind, aber von anderen übernommen werden (z. B. Scrum Master, agiles Team).

3.1.4 SG 4 Festlegen, Planen und Implementieren von Testprozessverbesserungen

Der Verbesserungsprozess, wie er in diesem speziellen TMMi-Ziel beschrieben ist, und seine spezifischen Praktiken werden weitgehend durch die Retrospektive abgedeckt, die ein agiles Team durchführt. Die Iterationsretrospektive ist eine Gelegenheit für das agile Team, sich selbst zu überprüfen und Maßnahmen für Verbesserungen zu definieren, die während der nächsten Iteration umgesetzt werden sollen. Eine Retrospektive findet in der Regel nach dem Iterationsreview und vor der Planung der nächsten Iteration statt.

In der Retrospektive bespricht das Team, was in der Iteration gut gelaufen ist, was verbessert werden konnte und

was in der nächsten Iteration verbessert werden soll. Während jeder Retrospektive identifiziert und plant das agile Team Möglichkeiten zur Steigerung der Produktqualität, z. B. durch die Verbesserung von Arbeitsprozessen oder die Anpassung der Definition of Done, falls erforderlich. Retrospektiven können zu testbezogenen Verbesserungsentscheidungen führen, die sich auf die Testeffektivität, die Testproduktivität und die Qualität der Tests konzentrieren. Sie können sich auch mit der Testbarkeit von Anwendungen, User Stories, Funktionen oder Systemschnittstellen befassen. Die Ursachenanalyse von Fehlern kann zu Verbesserungen bei Tests und Entwicklung führen. Tester spielen eine wichtige Rolle bei Retrospektiven. Sie sind Teil des Teams und bringen ihre einzigartige Perspektive ein. Alle Teammitglieder, Tester und Nicht-Tester, können sowohl zu Test- als auch zu Nicht-Test-Aktivitäten Beiträge leisten.



Am Ende der Retrospektive sollte das agile Team (Test-)Verbesserungen identifiziert haben, die es in der nächsten Iteration umsetzen wird. Die Umsetzung dieser Verbesserungen in der nächsten Iteration ist die Anpassung an die Inspektion des agilen Teams selbst. Das Team stimmt über die wichtigsten Verbesserungen für die nächste Iteration ab, so dass die (Test-)Verbesserungen möglicherweise länger als eine Iteration auf dem Backlog verbleiben, aber sie werden schließlich implementiert. Es ist wichtig, dass das Team nur Verbesserungen übernimmt, die es in die nächste Iteration einbauen kann, also wahrscheinlich maximal zwei oder drei pro Iteration. Die Tester sollten die Verantwortung für die Testverbesserungen übernehmen, um sicherzustellen, dass sie im Team umgesetzt werden. Die Retrospektive ist ein wichtiger Mechanismus, der es einem Team ermöglicht, sich während der gesamten Projektlaufzeit kontinuierlich weiterzuentwickeln und zu verbessern.

Da der Geltungsbereich oft auf den vorangegangenen Zyklus oder die vorangegangene Iteration beschränkt ist, werden kleine, aber häufige Verbesserungen vorgenommen, die sich hauptsächlich auf die Lösung spezifischer Projektprobleme konzentrieren. Der Schwerpunkt dieser Verbesserungen liegt häufig nicht auf dem projektübergreifenden Lernen und der Institutionalisierung von Verbesserungen. Wenn man sich ansieht, wie die (Test-)Prozessverbesserung organisiert und gemanagt wird, liegt der Schwerpunkt wahrscheinlich weniger auf einer (Test-)Prozessgruppe auf organisatorischer Ebene als vielmehr auf dem Selbstmanagement der Teams innerhalb des Projekts. Das ist zwar nicht schlecht, aber es ist wichtig, einen Mechanismus zu schaffen, mit dem erfolgreiche lokale Testverbesserungen mit dem Rest der Organisation geteilt werden und auch die Verbesserungsbereiche, die nicht in der Macht des agilen Teams liegen, angesprochen werden können. Eine mögliche Lösung wäre die Teilnahme eines Vertreters der Testorganisation an lokalen retrospektiven Meetings, um sicherzustellen, dass Verbesserungen in der gesamten Organisation geteilt und institutionalisiert werden, wenn dies angebracht ist. Die Rolle der Testprozessverbesserung kann auch allgemeinere oder grundlegende Probleme, die das Testen in der gesamten Organisation betreffen, abdecken.

3.1.5 SG 5 Einführung des organisationsweiten Testprozesses und Einarbeitung der gewonnenen Erkenntnisse

Auch für ein agiles Projekt ist es wichtig, Zugang zu den organisatorischen Standard-Testprozessen und Testprozess-Assets zu haben und diese wiederzuverwenden, soweit dies erforderlich ist und einen Mehrwert bietet. Die Testorganisation stellt sicher, dass diese in allen Projekten eingesetzt werden. (Weitere Informationen über den organisatorischen Standardtestprozess und die Testprozess-Assets und wie sich dies auf agile Projekte bezieht, finden Sie im Prozessbereich Testlebenszyklus und Integration).

Die Implementierung des Standardtestprozesses der Organisation und die Nutzung von Testprozess-Assets in agilen Projekten kann von den selbstorganisierenden Teams selbst überwacht werden, z.B. indem dies in Retrospektiven angesprochen wird. Wenn ein Vertreter der Testorganisation, z.B. ein Testprozessverbesserer, bei diesen Treffen anwesend ist, kann diese Person den Implementierungsstatus verfolgen und projektübergreifende Probleme ansprechen. Diese Person kann auch wertvolle Erkenntnisse und Testverbesserungen aus dem agilen Team zurück in die Testorganisation bringen. Die gewonnenen Erkenntnisse und Testverbesserungen können anschließend als Vorschläge zur Verbesserung des Testprozesses eingereicht und entsprechend verwaltet werden. Die aus dem Testen mit dem agilen Team gewonnenen Erkenntnisse werden so gegebenenfalls in den organisatorischen Standard-(Agile) (Test-)Prozess und die Testprozess-Assets aufgenommen.

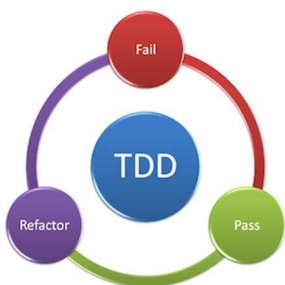
3.2 Prozessbereich 3.2 Testschulungsprogramm

Der Zweck des Prozessgebiets Testschulungsprogramm besteht darin, ein Trainingsprogramm zu entwickeln, das die Entwicklung der Kenntnisse und Kompetenzen der Mitarbeiter fördert, damit die Testaufgaben und -rollen effektiv und effizient durchgeführt werden können.

Das Hauptziel des Testschulungsprogramms besteht darin, den Testern und anderen Teammitgliedern die notwendige (Test-)Ausbildung zukommen zu lassen. Infolge der Aussage des Agilen Manifests "Individuen und Interaktionen vor Prozessen und Werkzeugen" sind die Prozesse in einer agilen Umgebung in der Regel leichtgewichtig dokumentiert und decken nicht alle möglichen Szenarien ab. Die Schulung der Mitarbeiter wird dann jedoch zu einem kritischen Erfolgsfaktor, auf den man sich konzentrieren muss und der fast wie eine Gegenmaßnahme zu den leichtgewichtigen Prozessen ist. Ein Qualitätsschulungsprogramm stellt sicher, dass die am Testen beteiligten Personen ihre Testkenntnisse und -kompetenzen kontinuierlich verbessern und sich das erforderliche Fachwissen sowie andere Kenntnisse und Kompetenzen im Zusammenhang mit dem Testen aneignen. Da Qualität und Testen nun eine Teamaufgabe sind, wird erwartet, dass testbezogene Schulungen nicht nur für Testexperten, sondern für das gesamte agile Team angeboten werden.

3.2.1 SG 1 Aufbau von Kompetenz zu organisationsweiten Testweiterbildung

Der Prozess beginnt mit der Ermittlung des strategischen Testschulungsbedarfs der Organisation. Neben dem traditionellen Schulungsbedarf für Tester, z. B. zu Testentwurfstechniken und Überdeckungskennzahlen, besteht nun auch die Notwendigkeit, Kenntnisse und Kompetenzen zu den Besonderheiten des agilen Testens zu erwerben. Beispiele hierfür sind das agile Manifest und seine Prinzipien, Frameworks wie Scrum, Kanban und XP, agile Testkonzepte (z. B. Testpyramide und Testquadranten), Test Driven Development (TDD), Behavior Driven Development (BDD), Acceptance Test Driven Development (ATDD), User-Story-Entwicklung einschließlich der



Unterstützung bei der Definition von Akzeptanzkriterien und deren Überprüfung auf Testbarkeit, Testautomatisierung und explorative Tests. Auch einige typische Wissens- und Kompetenzbereiche des Testmanagements wie Planung, Schätzung (z. B. Planungspoker), Überwachung und Risikoanalyse (z. B. Risikopoker) werden nun zu Bereichen, die zum Wissen und den Kompetenzen der meisten Tester gehören müssen, die in einem agilen Team arbeiten, und nicht nur zum Wissen des Testmanagers. Die Ermittlung des strategischen Schulungsbedarfs für das Testen beschränkt sich nicht nur auf die Tester, sondern es ist von größter Bedeutung, auch den testbezogenen Schulungsbedarf für andere Mitglieder agiler Teams zu ermitteln.

Wie in jeder anderen Organisation wird der Bedarf an Testschulungen in einen (leichtgewichtigen) Schulungsplan umgesetzt, der möglicherweise auch einige spezifische Projektschulungen für Tests umfasst. Der Schulungsplan muss überprüft werden und es müssen Verpflichtungen festgelegt werden. Beachten Sie, dass insbesondere im agilen Bereich einige Kompetenzen effektiv und effizient durch informelle Mittel vermittelt werden (z.B. Training-on-the-Job, Schulungen zum Mittagessen, Coaching und Mentoring), während andere Kompetenzen eine formelle Schulung erfordern. Der geeignete Ansatz zur Deckung des spezifischen Schulungsbedarfs im Testbereich sollte für jeden Wissens- und Kompetenzbereich festgelegt werden, wobei z. B. das Training-on-the-Job für exploratives Testen typischerweise ein wichtiger Aspekt ist.

3.2.2 SG 2 Erforderliche Testweiterbildung anbieten

Im Großen und Ganzen sind die spezifischen Praktiken für SG2 in einer traditionellen und einer agilen Organisation die gleichen. Auf der Grundlage des organisatorischen Schulungsplans werden die Teammitglieder identifiziert, die geschult werden müssen, um ihre Testrolle effektiv ausüben zu können. Die Schulung wird anschließend geplant, einschließlich der erforderlichen Ressourcen, und durchgeführt.

Dies scheint eine offensichtliche Aussage zu sein, aber einige agile Organisationen haben Schwierigkeiten, Zeit für Schulungen zu finden. Agile Teams sind immer voll ausgelastet und versuchen, die Frist für das Ende der Iteration einzuhalten. Natürlich muss die Zeit für die Schulung bei der Iterationsplanung berücksichtigt werden, wobei der Teilnehmer für die nächste Iteration weniger zur Verfügung stehen wird.

Wie bereits erwähnt, werden in einer agilen Organisation häufig informelle Schulungsinstrumente eingesetzt. So werden beispielsweise zusätzlich zur formalen Schulung Mentoring und Coaching am Arbeitsplatz in der Regel kontinuierlich gefördert und eingesetzt. Mentoring liegt in der Tat in der Verantwortung eines jeden und wird auf allen Ebenen einer Organisation erwartet. Die Arbeit in Zweiergruppen ist ein weiteres, von agilen Teams häufig genutztes Mittel zur Weiterbildung und zum Wissenstransfer. Bei der Anwendung von Scrum ist der Scrum-Master dafür verantwortlich, das Team in den Scrum-Praktiken zu coachen.

Es werden Aufzeichnungen über die durchgeführten Testschulungen, ob informell oder formell, erstellt und die Wirksamkeit der Testschulungen bewertet. Ein besuchtes (Test-)Training kann auch im Rahmen eines Retrospektivgesprächs ausgewertet werden, bei dem diskutiert wird, ob die Kenntnisse und Kompetenzen der Teilnehmer nun besser für die Durchführung der Testaufgaben geeignet sind.

3.3 Prozessbereich 3.3 Testlebenszyklus und Integration

Der Zweck von Testlebenszyklus und Integration besteht darin, einen brauchbaren Satz von organisatorischen Testprozess-Assets (z.B. einen Standard-Testlebenszyklus) und Arbeitsumgebungsstandards zu etablieren und zu pflegen und den Testlebenszyklus mit dem Entwicklungslebenszyklus zu integrieren und zu synchronisieren. Der integrierte Lebenszyklus gewährleistet eine frühzeitige Einbindung des Testens in ein Projekt. Der Zweck des Testlebenszyklus und der Integration besteht auch darin, einen kohärenten Testansatz über mehrere Teststufen hinweg zu definieren, der auf den identifizierten Risiken und der definierten Teststrategie basiert, und ein Gesamtestkonzept zu erstellen, das auf dem definierten Testlebenszyklus basiert.

3.3.1 SG 1 Etablierung organisationsweiter Testprozess-Assets

Eine wichtige Aufgabe der Testorganisation besteht darin, einen Standardtestprozess zu definieren, zu dokumentieren und aufrechtzuerhalten, der mit der Testrichtlinie und den Zielen der Organisation übereinstimmt. Er enthält unter anderem eine Beschreibung der verschiedenen Testarten und die auszuführenden Stufen. Während TMMi keine detaillierten, definierten und dokumentierten Testprozesse vorschreibt, sind häufig fokussierte Vorlagen, die mit Beispielen versehen sind, ein guter Ansatz, um eine gemeinsame Arbeitsweise zu etablieren. Siehe auch Bob Galens drei Säulen der agilen Qualität und des Testens, in denen er Standards als Checklisten, Vorlagen und Repositories als eine der Säulen des agilen Softwaretests bezeichnet [Galen]. Eine Vorlage identifiziert die erforderlichen Aktivitäten zum Sammeln von Informationen und zur Dokumentation der Ergebnisse, die in ihrer Struktur enthalten sind. Ein Prozess muss keine strikte Abfolge von Aktivitäten sein. Wenn Abhängigkeiten bestehen, können sie durch Notizen in der Vorlage oder durch Felder in einem Formular erfasst werden, die erst zugänglich sind, wenn andere vorausgesetzte Felder ausgefüllt sind. Vorlagen haben einen praktischen Wert, da sie die tatsächliche Absicht eines Prozesses vermitteln. Vorlagen vermeiden auch die Mehrdeutigkeiten, die häufig in "wortreichen" Prozessdokumenten zu finden sind. Wenn Sie als agile Organisation über wirksame Prozesse verfügen, müssen Sie diese vielleicht nur auf eine leichte Art und Weise dokumentieren und vielleicht ein paar Dinge hinzufügen. Einige wichtige Richtlinien, die in agilen Organisationen bei der Erstellung von Prozessen häufig verwendet werden:

- Die "Must Do's" des Prozesses werden getrennt von den Leitlinien verpackt
- Kein Vorgang ist länger als zwei Seiten (Ziel, weiche Regel)
- Die Prozesse enthalten keine "How to"-Informationen oder Informationen über Werkzeuge, es sei denn, man hat beschlossen, dies für alle Projekte unabhängig von deren Größe und Umfang vorzuschreiben.
- Getrennte Leitfäden enthalten Anpassungs-/Planungsoptionen und "How-to"-Informationen. Beachten Sie, dass man sich bei den "How-to"-Informationen auch einfach auf ein vorhandenes Buch oder Papier beziehen kann.

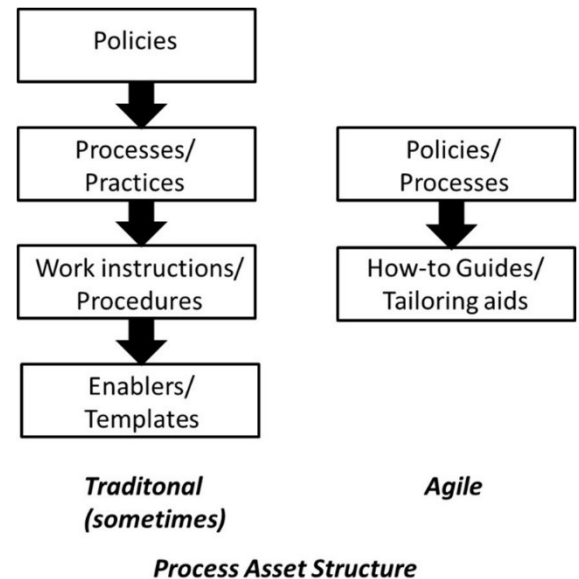
In vielen großen Organisationen ist es nicht ungewöhnlich, dass es vier Ebenen von Prozess-Assets gibt: Richtlinien, Prozesse/Praktiken, Arbeitsanweisungen/Verfahren und Befähiger/Vorlagen. Dies liegt nicht an den Anforderungen des TMMi, sondern an der Art und Weise, wie viele große Organisationen ihre Prozessressourcen implementieren. Die Wahl der Prozess-Assets ist zwar jedem Unternehmen selbst überlassen, aber die meisten agilen Unternehmen haben festgestellt, dass zwei Ebenen von Prozess-Assets ausreichend sind. Dies wird erreicht, indem eine Grundsatzerklärung mit der zugehörigen Prozessbeschreibung konsolidiert wird, die festlegt, "was zu tun ist", wenn der Prozess ausgeführt wird. Die zweite Ebene enthält "How to"-Richtlinien zur Durchführung und Anpassung des

Prozesses. Diese Ebene kann als Hilfestellung für die Anpassung des Prozesses angesehen werden, die dem Team die Autonomie gibt, zu entscheiden, wie genau es innerhalb des definierten Rahmens arbeiten soll, und enthält in der Regel unterstützende Vorlagen. In den meisten agilen Organisationen werden Schritt-für-Schritt-Verfahren durch Tool-Guides und Training/Mentoring ersetzt. Wie bereits erwähnt, kann eine Vorlage, wie z. B. eine Testcharta-Vorlage, eine Vorlage für ein Testsitzungsblatt oder eine Testkonzeptvorlage, als Prozess dienen, der die erforderlichen Prozessaktivitäten enthält. Dies ist verbreitetes Verfahren zur Entwicklung effektiver agiler Prozessbeschreibungen. [McMahon]

Die Standardtestprozesse der Organisation können von den Projekten angepasst werden, um ihre eigenen definierten Prozesse zu erstellen. Im Zusammenhang mit SP 1.3 Festlegung von Kriterien und Richtlinien für die Anpassung verwenden die meisten Organisationen heute einen so genannten Tailor-Down-Ansatz. Bei diesem Ansatz müssen die Mitarbeiter mühsam begründen, warum etwas nicht benötigt wird. Dies verursacht den größten Mehraufwand für die einfachsten Projekte, was kein agile-freundlicher Ansatz ist. Wie weit kann man beim Tailoring Down gehen? Gibt es ein definiertes Minimum? Wenn man jedoch mit einer Mindestmenge beginnt, bei der die Mindestmenge das ist, was jeder tun muss, eliminiert man das Risiko, ein "Muss" herauszuschneiden. Ein Tailoring-up-Ansatz ist viel konsistenter mit agilen Ansätzen und vollständig TMMi-konform. Tailoring ist ein äußerst leistungsfähiger Mechanismus für agile Organisationen, die ihre Agilität beibehalten wollen, oder für hochentwickelte (Test-)Organisationen, die ihre Agilität steigern wollen. Eine Reihe einfacher Kriterien könnte dabei helfen, beim Tailoring die richtigen Fragen zu stellen. Ohne solche Kriterien neigen manche dazu, zu viel zuzuschneiden.

Um die Einführung des Standardtestprozesses zu unterstützen, wird eine Testprozess-Asset-Bibliothek erstellt und gepflegt, oft in einem Wiki, in der Tester beispielsweise Vorlagen, Best Practices und Checklisten finden, die sie bei ihrer täglichen Arbeit unterstützen. Die Erstellung einer Testprozess-Asset-Bibliothek ist auch in einem agilen Kontext sinnvoll, solange die Teams einen Mehrwert erhalten, wenn sie die bereitgestellten Assets während ihrer Testaktivitäten nutzen. Mehr oder weniger die gleiche Argumentation gilt für eine Testprozessdatenbank, in der Testdaten, z.B. Schätzungen, Abdeckungen, Qualitätsmessungen, gesammelt und aufbereitet werden und den Teams zur Verfügung stehen. Stellen Sie sicher, dass alles, was gesammelt und zur Verfügung gestellt wird, um teamübergreifendes Lernen und den Austausch von Erfahrungen zu ermöglichen, einen Mehrwert für die Teams hat.

Natürlich werden auch in einem agilen Kontext Arbeitsumgebungsstandards verwendet, um die Schaffung von Projektarbeitsumgebungen anzuleiten.



3.3.2 SG 2 Integration des Modells für den Testlebenszyklus mit den Modellen für die Entwicklung

Das Standardmodell des Testlebenszyklus definiert die wichtigsten Aktivitäten und Leistungen für die verschiedenen Teststufen. Das Standardmodell des Testlebenszyklus ist auf die Modelle des Entwicklungslebenszyklus abgestimmt, um die Testaktivitäten in Bezug auf Phasen, Meilensteine, Ergebnisse und Aktivitäten zu integrieren. Die Integration des Lebenszyklus erfolgt so, dass eine frühzeitige Einbindung des Testens in die Projekte gewährleistet ist. Das hört sich alles so an, als gäbe es hier nichts zu tun, wenn ein agiler Lebenszyklus eingeführt wurde; es wurde bereits alles erledigt. Bei Agilität sollten Entwicklung und Testen vollständig in den Lebenszyklus integriert sein, und das Testen sollte bereits so früh wie möglich stattfinden.

So sehr dies auf viele Organisationen zutrifft, die Agilität einsetzen, so gibt es auch Organisationen, die mit diesem Aspekt zu kämpfen haben. Ihre Iteration ähnelt einem Mini-V-Modell des Lebenszyklus für die Entwicklung, bei dem das Testen erst gegen Ende der Iteration beginnt, oft unter Zeitdruck durchgeführt und manchmal sogar abgekürzt wird. Natürlich handelt es sich hierbei nicht um eine Implementierung eines integrierten agilen Entwicklungs- und Testlebenszyklus, wie er sein sollte. Es zeigt jedoch, dass für einige Organisationen, die "Agile-like" sind, dieses spezifische Ziel ein Bereich ist, der verbessert werden kann. Es ist wichtig, dass ein gemeinsames Verständnis

geschaffen wird, indem definiert wird, wie das Testen frühzeitig in der Iteration einbezogen und durchgeführt werden kann.

Dieses spezifische Ziel gilt also auch für eine Organisation, die agile Softwareentwicklung betreibt. Wenn es richtig gemacht wird, sollten Entwicklung und Testen vollständig integriert sein, und es gibt ein gemeinsames Verständnis darüber, wie dies zu tun ist. In solchen Fällen muss nichts Zusätzliches geschehen. Unter anderen Umständen, wie oben beschrieben, ist dieses spezifische Ziel ein sehr wichtiger Verbesserungsbereich und sollte zu einer viel besseren Abstimmung der Entwicklungs- und Testaktivitäten in einer Iteration führen.

3.3.3 SG 3 Erstellung eines Master-Testkonzepts

Auf TMMi-Stufe 3 befasst sich das Testen mit der Gesamttestplanung, die die Koordinierung von Testaufgaben, Verantwortlichkeiten und Testansatz über alle Teststufen hinweg betrifft. Dies verhindert unnötige Redundanzen oder Auslassungen von Tests zwischen den verschiedenen Teststufen und kann die Effizienz und Qualität des gesamten Testprozesses erheblich steigern. Der Master-Testkonzept beschreibt die Anwendung der Teststrategie für ein bestimmtes Projekt, einschließlich der einzelnen durchzuführenden Stufen und der Beziehung zwischen diesen Stufen. Auf TMMi-Stufe 2 wird die (Test-)Planung bereits sowohl auf Release- als auch auf Iterationsebene als Teil des Prozessbereichs Testplanung behandelt. Ein agiles Team führt in der Regel mehrere Teststufen durch, die alle im Rahmen der Release- und Iterationsplanung angemessen berücksichtigt werden sollten.

Einige Projekte, die nach einem agilen Lebenszyklusmodell arbeiten, haben jedoch einen etwas hybriden Ansatz gewählt, bei dem zusätzlich zu den Tests innerhalb eines agilen Teams auch einige Tests außerhalb des Teams organisiert werden. In solchen Fällen definiert und verwaltet ein Master-Testkonzept die Beziehung zwischen den agilen Teams und den Teststufen, die außerhalb ihres Zuständigkeitsbereichs durchgeführt werden, z. B. spezifische nicht-funktionale Tests (Sicherheit, Leistung), Hardware-/Software-Integrationstests, Systemintegrationstests oder Beta-Tests. Alle auf dieses spezifische Ziel bezogenen spezifischen Praktiken werden dann in der Regel auch anwendbar sein. Vor allem in größeren Projekten und bei wachsenden Organisationen wird es immer wichtiger, einen übergreifenden Master-Testkonzept und die damit verbundenen Entscheidungen auf konsistente Weise zu dokumentieren. Dies gilt insbesondere, wenn eine Organisation skalierte agile Frameworks verwendet.

Wenn alle Tests mit den agilen Teams durchgeführt werden, hat ein spezifischer Master-Testkonzept keinen zusätzlichen Nutzen, so dass dieses TMMi-Ziel in solchen Fällen wahrscheinlich nicht relevant ist. Hier wird erwartet, dass die Planung auf Release- und Iterationsebene alle notwendigen Testaspekte abdeckt.

3.4 Prozessbereich 3.4 Nicht-funktionale Tests

Der Zweck des Prozessbereichs Nicht-funktionales Testen ist die Verbesserung der Testprozessfähigkeiten, um nicht-funktionales Testen während der Testplanung, des Testdesigns und der Testdurchführung einzubeziehen.

Ob nicht-funktionale Tests wichtig sind, ist unabhängig vom Lebenszyklus. Das zu entwickelnde Produkt gibt vor, welche nicht-funktionalen Aspekte für das Testen relevant sind. Dies gilt natürlich sowohl für sequenzielle Lebenszyklusmodelle als auch für agile Entwicklungsmodelle. Nicht-funktionales Testen ist daher auch für die agile Softwareentwicklung anwendbar.



Je nach Art der nicht-funktionalen Aspekte und des verwendeten Testansatzes können jedoch einige nicht-funktionale Tests, z. B. Performanz und Zuverlässigkeit, nicht in einer kurzen Iteration getestet werden, wenn sie auf herkömmliche Weise durchgeführt werden. Da diese nicht-funktionalen Tests je nach den verwendeten Methoden und Techniken Wochen für die Erstellung, Organisation und Ausführung benötigen können, ist ihre Durchlaufzeit weniger geeignet für den Kern der Iterationen. Daher ist häufig ein anderer Ansatz für das Testen nichtfunktionaler Eigenschaften erforderlich. Dies ist eine der grundlegenden Veränderungen zum Besseren bei agile, da Qualitätsattribute bereits früh während der Iterationen getestet werden und nicht erst am Ende, wie bei einem traditionellen Leben

Zyklusmodell. Möglicherweise sind nicht alle Komponenten/Funktionen aus der ersten Iteration verfügbar, um einen

vollständigen nicht-funktionalen Test durchzuführen, doch können die Ergebnisse dieses Tests genutzt werden, um frühzeitig Hinweise auf Qualitätsprobleme zu erhalten.

Die User Stories sollten sowohl funktionale als auch nicht-funktionale Elemente ansprechen, um sicherzustellen, dass das richtige Produkt für die Benutzer und Kunden entwickelt wird. Die Qualitätsmerkmale nach ISO 25010 können bei der Strukturierung der Anforderungen helfen, und die Tester sollten diese nicht-funktionalen Elemente berücksichtigen [ISO25010].

3.4.1 SG 1 Durchführung einer nicht-funktionalen Produktrisikobewertung

Die Produktrisikositzung(en) für agile Projekte, wie sie im spezifischen Ziel SG1 Risikobewertung durchführen als Teil des Prozessbereichs Testplanung identifiziert und beschrieben wurden, werden nun explizit auch auf nicht-funktionale Aspekte und Risiken ausgeweitet. Auf Release-Ebene kann die Risikobewertung, nun auch für nicht-funktionale Tests, auf Basis der Produktvision durchgeführt werden. Auf Iterationsebene wird dies anhand der User Stories durchgeführt, die nicht-funktionale Anforderungen als wichtigen Input definieren.

Vorzugsweise sollten alle Teammitglieder, einschließlich des Produkteigentümers, und möglicherweise einige andere Beteiligte an den Produktrisikositzungen teilnehmen. Für einige nicht-funktionale Bereiche können Spezialisten zur Unterstützung herangezogen werden. Die Produktrisikositzungen führen in der Regel zu einer dokumentierten Liste von priorisierten (nicht-funktionalen) Produktrisiken. Wie bereits im Prozessbereich Testplanung erwähnt, sind in agilen Projekten sowohl der verwendete Prozess als auch die daraus resultierende Dokumentation im Vergleich zu einem traditionellen Projekt, das einem sequenziellen Lebenszyklusmodell folgt, sehr viel leichter zu handhaben.

3.4.2 SG 2 Festlegung eines nicht-funktionalen Testansatzes

Ein Testansatz für die relevanten nicht-funktionalen Qualitätsmerkmale wird definiert, um die identifizierten und priorisierten nicht-funktionalen Produktrisiken zu mindern. Für eine bestimmte Iteration werden die zu testenden nicht-funktionalen Merkmale während der Iterationsplanung identifiziert. Die priorisierte Liste der zu testenden nicht-funktionalen Merkmale bezieht sich in der Regel auf die in dieser Iteration zu testenden User Stories. Während der Iteration können neue nicht-funktionale Produktrisiken auftauchen, die zusätzliche Tests erfordern. Themen wie neue nicht-funktionale Produktrisiken, die zusätzliche Tests erfordern, werden in der Regel in täglichen Standup-Meetings besprochen.

Der nicht-funktionale Testansatz, der auf Iterationsebene definiert wird, um die nicht-funktionalen Risiken zu mindern, umfasst in der Regel die Identifizierung geeigneter nicht-funktionaler Testmethoden und Testtechnik(en) auf der Grundlage der Höhe und Art der nicht-funktionalen Risiken. In der Regel wird auch die Verwendung von unterstützenden Tools und der Ansatz zur Testautomatisierung für nicht-funktionale Tests behandelt. Der Testansatz für nicht-funktionale Tests auf Release-Ebene wird auf einer viel höheren Ebene angesiedelt sein und sollte auf der definierten Teststrategie auf Programm- oder Organisationsebene und auch auf dem definierten Master-Testkonzept basieren, falls dieser existiert. (Siehe SG3 Erstellung eines Master-Testkonzepts innerhalb des Prozessbereichs Testlebenszyklus und Integration zur Begründung, ob ein MasterTestkonzept erstellt werden sollte oder nicht). Sowohl der nicht-funktionale Testansatz auf Release- als auch auf Iterationsebene ist Teil des gesamten Testansatzes und wird im Team-/Projekt-Wiki festgehalten oder angezeigt.

Nicht-funktionale Endekriterien sind Teil der sogenannten Definition of Done (DoD). Es ist wichtig, dass die DoD spezifische Kriterien für nicht-funktionale Tests enthält, z. B. Mean-Time-Between-Failures (MTBF) oder "Front-End-Webseiten haben für die OWASP-Top-10-Risikoliste getestet wurden". Die Iteration sollte zur Implementierung des vereinbarten Satzes nicht-funktionaler User Stories (einschließlich ihrer Akzeptanzkriterien) führen und die nicht-funktionalen (Test-)Endekriterien, wie im DoD definiert, erfüllen. Beachten Sie, dass nicht-funktionale Attribute auch Teil der Akzeptanzkriterien für funktionale User Stories sein können und nicht unbedingt als separate nicht-funktionale User Stories spezifiziert werden müssen. Es gibt nicht nur eine Definition of Done auf Iterationsebene, sondern oft auch eine DoD auf Release-Ebene, die mehrere Iterationen umfasst. Auch die DoD auf Release-Ebene kann nicht-funktionale Kriterien enthalten.

3.4.3 SG 3 Analyse und Entwurf nicht-funktionaler Tests

Dieses spezifische Ziel folgt weitgehend den gleichen Praktiken, aber nun aus einer nicht-funktionalen Perspektive, wie bei dem spezifischen Ziel SG1 Testanalyse und -entwurf unter Verwendung von Testentwurfstechniken aus dem Prozessgebiet Testentwurf und -durchführung durchführen. Während der Testanalyse und des Testdesigns wird der Testansatz für das nicht-funktionale Testen in konkrete Testbedingungen und Tests umgesetzt. Bei Agilität sind Testanalyse und -design sowie die Testdurchführung sich gegenseitig unterstützende Aktivitäten, die in der Regel während einer Iteration parallel laufen. Dies gilt auch für die meisten nicht-funktionalen Tests. Die nicht-funktionale Testanalyse ist somit keine explizite separate Aktivität, sondern eine implizite Aktivität, die Tester als Teil ihrer Rolle innerhalb der kollaborativen User-Story-Entwicklung durchführen.

Auf der Grundlage der Analyse der nicht-funktionalen User Storys werden die nicht-funktionalen Testbedingungen ermittelt. Testbedingungen sind im Grunde eine Identifizierung von "Dingen", die getestet/abgedeckt werden müssen. Sofern die definierten Akzeptanzkriterien detailliert und klar genug sind, übernehmen sie in der Regel die Rolle der traditionellen Testbedingungen. Die Akzeptanzkriterien werden anschließend in nicht-funktionale Tests umgesetzt. Bei nicht-funktionalen Tests ist es oft vorteilhaft, die Testanalyse auf einer höheren Ebene als nur den User Stories durchzuführen. So kann beispielsweise ein Feature oder ein Epos oder eine Sammlung von Stories analysiert werden, um nicht-funktionale Testbedingungen zu identifizieren, die abstrakter sind als die auf User-Story-Ebene und auch mehrere User-Stories umfassen. Mit dem Test-First-Prinzip, das bei Agilität angewendet wird, werden nicht-funktionale Tests, die die Menge der nicht-funktionalen Testbedingungen abdecken, vor oder zumindest parallel zur Entwicklung des Codes identifiziert (und möglicherweise automatisiert).

Bei den meisten manuellen nicht-funktionalen Tests werden die Tests im Laufe der Durchführung der nicht-funktionalen Tests durch das Team identifiziert/verfeinert. Die Tests werden in den meisten Fällen nicht so detailliert dokumentiert wie in traditionellen Projekten, sondern eher in Form von Testideen, wenn explorative Tests verwendet werden. Für komplexe und kritische nicht-funktionale Bereiche kann ein traditionellerer Ansatz für den Testentwurf und die Testfallentwicklung unter Verwendung formaler nicht-funktionaler Testentwurfstechniken der beste Weg sein, um die Risiken abzudecken. Allerdings ist auch bei diesem Ansatz der Umfang der Dokumentation im Vergleich zum nicht-funktionalen Testen in einer traditionellen sequenziellen Lebenszyklsumgebung begrenzt. Die Priorisierung von nicht-funktionalen Tests folgt in der Regel der Priorisierung der User Story, die sie abdecken. Die Priorisierung kann jedoch auch durch den Zeitaufwand für die Vorbereitung und Durchführung eines bestimmten nicht-funktionalen Tests bestimmt werden.

Es werden spezifische Testdaten identifiziert, die zur Unterstützung der Durchführung von nicht-funktionalen Tests erforderlich sind. Bei Agilität werden die benötigten Testdaten in der Regel nicht zuerst vollständig in einem Testspezifikationsdokument spezifiziert. Wenn sie spezifiziert werden, werden sie oft als Teil der User Story aufgezeichnet. Sofern die notwendigen Werkzeuge und/oder Funktionalitäten zur Verfügung stehen, werden die zur Unterstützung der nicht-funktionalen Tests benötigten Testdaten jedoch meist sofort erstellt, so dass mit der Ausführung der nicht-funktionalen manuellen Tests sofort begonnen werden kann. Im Gegensatz dazu müssen die Daten für automatisierte nicht-funktionale Tests in der Regel im Voraus spezifiziert werden.

Die Rückverfolgbarkeit zwischen den nicht-funktionalen Anforderungen, den Testbedingungen und den Tests muss hergestellt und aufrechterhalten werden. Die Teams müssen deutlich machen, dass sie die verschiedenen nicht-funktionalen User Stories und Akzeptanzkriterien als Teil ihrer Tests abgedeckt haben. In vielen agilen Organisationen bilden die User Stories die Grundlage für die Entwicklung einer Reihe von zugehörigen Akzeptanzkriterien und anschließend für die Tests. Mit diesem Ansatz kann eine horizontale Rückverfolgbarkeit von den Anforderungen bis zum Test erreicht werden.

3.4.4 SG 4 Realisierung von nicht-funktionalen Tests

Bei der Testrealisierung geht es darum, alles bereitzustellen, was für die Durchführung von Tests erforderlich ist. In der Regel wird die Entwicklung von Testdokumentation, z. B. von Testverfahren, zur Unterstützung der Testdurchführung auf ein Minimum reduziert. Stattdessen werden automatisierte (Regressions-)Testskripte entwickelt und priorisiert.

Die nicht-funktionale Testrealisierung wird viele der Praktiken befolgen, die bereits unter dem spezifischen Ziel SG2 Testrealisierung als Teil des Prozessbereichs Testentwurf und -durchführung beschrieben wurden.

Was im Einzelnen zu tun ist und wie nicht-funktionale Tests durchgeführt werden, hängt weitgehend von dem definierten Ansatz, den verwendeten Techniken und den zu prüfenden nicht-funktionalen Merkmalen und deren Umfang ab. So kann beispielsweise ein exploratives Testen, das weniger Vorbereitung erfordert, für das Testen der Benutzerfreundlichkeit geeignet sein, ist aber oft weniger geeignet für umfangreiche Zuverlässigkeits- und Performanztests.

Für einige nicht-funktionale Qualitätsmerkmale ist die Verfügbarkeit von Testdaten unerlässlich und muss während der Testdurchführung erstellt werden. Dies ist weitgehend dasselbe wie bei herkömmlichen Projekten, abgesehen davon, dass sie vielleicht nur kurz dokumentiert werden, was eine Wiederverwendung für Regressionstests ermöglicht.

Natürlich wird die Testdurchführung und -vorbereitung so bald wie möglich parallel zu anderen (Test-)Aktivitäten beginnen. Es handelt sich dabei nicht um eine separate Phase, sondern um eine Reihe von Aktivitäten (die auf der Aufgabentafel aufgeführt sind), die durchgeführt werden müssen, um eine effiziente und effektive Testdurchführung zu ermöglichen.

3.4.5 SG 5 Durchführung von nicht-funktionalen Tests

Wie beim vorherigen Ziel in diesem Prozessbereich werden wir uns weitgehend auf das zugehörige spezifische Ziel SG 3 Testdurchführung ausführen des Prozessbereichs Testentwurf und Testdurchführung beziehen, das weiter oben in diesem Dokument besprochen wurde. Die Praktiken für die Durchführung von nicht-funktionalen Tests, das Berichten von Testvorfällen und das Schreiben von Testprotokollen in einer agilen Umgebung sind im Grunde die gleichen wie für funktionale Tests in einer agilen Umgebung. Sie werden typischerweise mit viel weniger Dokumentationsintensität durchgeführt als in einem traditionellen Projekt. Oft werden keine detaillierten Testverfahren und Testprotokolle erstellt.

Die Durchführung nicht-funktionaler Tests erfolgt entsprechend den bei der Iterationsplanung festgelegten Prioritäten. Einige Tests können unter Verwendung eines dokumentierten Testverfahrens durchgeführt werden, aber typischerweise werden auch viele nicht-funktionale Tests unter Verwendung explorativer und sitzungsbasierter Tests als Rahmen durchgeführt. Bei der iterativen Entwicklung besteht ein erhöhter Bedarf an der Organisation und Strukturierung von Regressionstests. Dies geschieht manchmal manuell, wird aber vorzugsweise mit automatisierten Regressionstests und unterstützenden Tools durchgeführt. Regressionstests beziehen sich natürlich auch auf die nicht-funktionalen Aspekte des Systems, die als wichtig für den Test identifiziert worden sind.

Die beim Testen gefundenen nicht-funktionalen Vorfälle können vom Team protokolliert und gemeldet werden. In agilen Projekten wird in der Regel diskutiert, ob tatsächlich alle gefundenen Vorfälle protokolliert werden sollten. Manche Teams protokollieren nur Vorfälle, die Iterationen entgehen, manche protokollieren sie, wenn sie nicht heute behoben werden können, manche protokollieren nur Vorfälle mit hoher Priorität. Wenn nicht alle gefundenen Vorfälle protokolliert werden, müssen Kriterien vorhanden sein, um zu bestimmen, welche Vorfälle protokolliert werden sollen und welche nicht. Manchmal werden Vorfälle auf dem agilen Task Board protokolliert, entweder als Aufkleber auf einer Aufgabe oder als separate Aufgabe, um zu verdeutlichen, dass sie die Fertigstellung einer Story und ihrer Aufgaben blockieren. Manche entscheiden sich dafür, gefundene Vorfälle mit einem Tool zu protokollieren und zu dokumentieren, z. B. mit einem Defect Management oder Defect Tracking Tool. Ein solches Tool sollte dann so schlank wie möglich eingesetzt werden und keine Elemente von nicht-funktionalen Testvorfallberichten erzwingen, die keinen oder einen zu geringen Mehrwert haben.

Auch in agilen Teams gilt es als gute Praxis, während der Durchführung nicht-funktionaler Tests Daten zu protokollieren, um festzustellen, ob die getesteten Elemente die definierten Abnahmekriterien erfüllen und tatsächlich als "erledigt" bezeichnet werden können. Die protokollierten Informationen sollten in einer Form von Statusmanagement-Tools (z. B. Testmanagement-Tools, Aufgabenmanagement-Tools, Task Board) erfasst und/oder zusammengefasst werden, so dass das Team und die Stakeholder den aktuellen Status aller durchgeführten Tests leicht nachvollziehen können.

3.5 Prozessbereich 3.5 Peer-Reviews

Der Zweck des Prozessbereichs Peer-Review besteht darin, zu überprüfen, ob die Arbeitsprodukte die festgelegten Anforderungen erfüllen, und Mängel an ausgewählten Arbeitsprodukten frühzeitig und effizient zu beseitigen. Ein

wichtiger Nebeneffekt ist die Entwicklung eines besseren Verständnisses der Arbeitsprodukte und der Fehler, die vermieden werden könnten.

3.5.1 SG 1 Einführung eines Peer-Review-Ansatzes

Agile Teams führen in der Regel keine formellen Peer-Reviews in dem Sinne durch, dass sie sich nicht zu einem bestimmten Zeitpunkt treffen, um Feedback zu einem Produkt zu geben. Sie erreichen jedoch die Absicht von Peer-Reviews, indem sie während der gesamten Entwicklung kontinuierliche, weniger formelle Peer-Reviews durchführen. Dies ist eine gängige Praxis in vielen Agilen Organisationen. Dennoch muss bei der Durchführung dieser Aktivitäten eine gewisse Disziplin herrschen. Dieses TMMi-Ziel befasst sich mit den Verfahren zur Einführung eines Peer-Review-Ansatzes in einem Projekt. Ein Review-Ansatz definiert, wie, wo und wann Review-Aktivitäten stattfinden sollen und ob diese Aktivitäten formell oder informell sind. Die Etablierung eines Peer-Review-Ansatzes ist auch auf agile Projekte anwendbar, allerdings sind die angewandten Review-Techniken und die Art und Weise, wie die Reviews organisiert werden, typischerweise sehr unterschiedlich.

Beispiele für Peer-Reviews, die typischerweise in agilen Projekten durchgeführt werden:

- regelmäßige Verfeinerungs-/Pflege-Sitzungen zu den Spezifikationen (z.B. User Stories) mit dem Team und den Stakeholdern während einer Iteration;
- tägliche Besprechungen mit anderen Teammitgliedern, um die Arbeitsprodukte, z. B. Code oder Tests, die entwickelt werden, offen zu diskutieren und Feedback zu geben.
- die frühzeitige und häufige Vorführung von Produkten bei den Kunden, zumindest am Ende einer Iteration im Rahmen der Iterationsprüfung;

Schlechte Spezifikationen sind oft ein Hauptgrund für das Scheitern von Projekten. Probleme bei der Spezifikation können daraus resultieren, dass die Benutzer keinen Einblick in ihre wahren Bedürfnisse haben, dass es keine globale Vision für das System gibt, dass Funktionen redundant oder widersprüchlich sind und dass es andere Missverständnisse gibt. Bei der agilen Entwicklung werden User Stories geschrieben, um die Anforderungen aus der Perspektive von Geschäftsvertretern, Entwicklern und Testern zu erfassen. Bei der sequenziellen Entwicklung wird diese gemeinsame Vision einer Funktion durch formelle Reviews erreicht, nachdem die Anforderungen geschrieben wurden; bei der agilen Entwicklung wird diese gemeinsame Vision durch häufige informelle Reviews erreicht, während die Anforderungen erstellt werden. Diese informellen Reviewsitzungen werden oft als Backlog-Verfeinerung oder Backlog-Grooming-Sitzungen bezeichnet. Während der Verfeinerungssitzungen verwenden der Vertreter des Unternehmens und das Entwicklungsteam (und die Stakeholder, falls vorhanden) Reviewtechniken, um den erforderlichen Detaillierungsgrad für die Implementierung zu finden und offene Fragen zu klären.

Das nahezu kontinuierliche Review der in der Entwicklung befindlichen Arbeitsprodukte durch das Team ist Teil des Ansatzes des gesamten Teams. Sie werden mit der Absicht durchgeführt, Fehler frühzeitig zu erkennen und auch Verbesserungsmöglichkeiten zu identifizieren. Agile Methoden und Techniken wie XP oder Pairing beinhalten auch Peer-Reviews als eine zentrale Praxis, um Feedbackschleifen für die Teams zu schaffen. Bei hoher Komplexität oder hohem Risiko kann sich das Team natürlich für eine halbformale oder formale Reviewtechnik entscheiden, z. B. eine Inspektion. In diesen Fällen gibt es eine klare Begründung für den Aufwand einer formelleren Überprüfung und die Anwendung einer disziplinierten Arbeitsweise. Ein Teil der Kriterien für den Reviewansatz kann festgelegt werden, wann eine formelleres Reviewverfahren angewendet werden soll.

Während beim Requirements Engineering die Validierung der Anforderungen durch Methoden wie informelle Reviews, Walkthroughs, Inspektionen oder perspektivisches Lesen im Vordergrund steht, streben agile Methoden eher eine Validierung der Anforderungen durch häufiges und frühzeitiges Feedback an, um schnell wertvolle Produktinkremente zu implementieren. Die Notwendigkeit einer frühen formalen Validierung wird dadurch verringert, dass schnelle Ergebnisse in Form von integrierten Produktinkrementen gezeigt werden. Wenn das Inkrement die Anforderungen der Stakeholder nicht vollständig erfüllt, wird das Delta in Form von neuen Anforderungen wieder in das Produkt-Backlog aufgenommen und mit allen anderen Backlog-Elementen priorisiert. Eine Demonstration dessen, was während einer Iteration tatsächlich gebaut wurde, ist einfach ein sehr effizienter Weg, um eine validierungsbasierte Konversation über etwas Konkretes anzuregen. Nichts bringt die Konversation so in Schwung wie die Tatsache, dass man sehen kann, wie etwas tatsächlich funktioniert. Die Demonstration ist eine Aktivität, die während des Iterationsreviews durchgeführt wird. Die Funktionen werden den Stakeholdern vorgeführt und mit ihnen besprochen, und es werden notwendige Anpassungen am Product Backlog oder am Release-Plan vorgenommen,

um neue Erkenntnisse aus der Diskussion zu berücksichtigen.

3.5.2 SG 2 Durchführung von Peer-Reviews

Wie jeder Ansatz sollte auch dieser nicht nur als definierter und vereinbarter Ansatz existieren, sondern auch befolgt werden. Das Team muss während einer Iteration viel Zeit für die Verfeinerung des Backlogs aufwenden, informelle (und möglicherweise formelle) Reviews als Teil der täglichen Routine durchführen und regelmäßig Stakeholder-Demonstrationen abhalten. Stakeholder-/Kundendemonstrationen werden mindestens am Ende jeder Iteration erwartet.

Eingangskriterien sind in der Regel nur bei formellen Reviews anwendbar und daher in agilen Projekten wahrscheinlich weniger oder gar nicht relevant. Die Verwendung von Endekriterien ist jedoch anwendbar und hat einen zusätzlichen Nutzen. INVEST [Wake] ist ein Beispiel für eine Reihe von Endekriterien, auf die in agilen Projekten bei der Überprüfung und Aktualisierung von User Stories häufig Bezug genommen wird. INVEST ist ein Akronym, das die folgenden Konzepte umfasst, die eine gute Benutzergeschichte ausmachen:

- Unabhängig
- Verhandlungsfähig
- Wertvoll
- Einschätzbar
- Klein
- Testbar.



Natürlich gibt es auch andere qualitätsbezogene Ende Kriterien, die ebenfalls vorteilhaft eingesetzt werden können.

Es ist wichtig, dass der Tester als Teil des agilen Teams an den Review-Sitzungen teilnimmt. Dies gilt insbesondere für Sitzungen, in denen Arbeitsprodukte besprochen und überprüft werden, die als Grundlage für das Testen während der Iteration dienen, z. B. bei Backlog-Verfeinerungssitzungen. Es wird empfohlen, dass mindestens ein Entwickler und ein Tester bei der Verfeinerung des Backlogs anwesend sind, um sicherzustellen, dass alternative Sichtweisen auf das System vorhanden sind. Typischerweise wird die einzigartige Perspektive des Testers die User Story verbessern, indem er fehlende Details oder nicht-funktionale Anforderungen identifiziert. Ein Tester kann insbesondere die Identifizierung und Definition von Akzeptanzkriterien für eine bestimmte User Story unterstützen. Ein Tester trägt dazu bei, indem er Geschäftsvertretern offene "Was wäre wenn?"-Fragen zur User Story stellt, Möglichkeiten zum Testen der User Story vorschlägt und die Akzeptanzkriterien bestätigt.

Die spezifische Praxis 2.3 Analyse von Peer-Review-Daten ist besonders bei formellen Reviews relevant. Um sicherzustellen, dass dieser Aufwand sowohl effizient als auch effektiv ist, werden Peer-Review-Daten gesammelt und dem Team mitgeteilt, um daraus zu lernen und den Review-Prozess zu optimieren. Wie bereits erwähnt, sind formale Reviews, z. B. Inspektionen, in agilen Projekten weniger üblich. Während das Sammeln von Daten ein wesentlicher Bestandteil von formellen Reviews ist, ist dies bei informellen Reviews viel seltener der Fall. Infolgedessen könnte man argumentieren, dass detaillierte Peer-Review-Datenerfassung, -Analyse und -Kommunikation in einem agilen Kontext weniger oder gar nicht relevant sind. Im Zusammenhang mit der Entscheidung, bei Peer-Reviews Daten zu sammeln, sollten Sie sich die folgenden Fragen stellen:

- Wer wird diese Daten verwenden, wenn sie gesammelt werden?
- In welchem Zusammenhang stehen diese Daten mit unseren Geschäftszielen?

Wenn niemand die Daten sinnvoll nutzen wird, sollten Sie keine wertvollen Ressourcen für die Sammlung dieser Daten verschwenden. Daraus folgt, dass für die meisten agilen Projekte die spezifische Praxis 2.3 Analyse von Peer-Review-Daten wahrscheinlich als nicht relevant angesehen wird. Beachten Sie, dass einige grundlegende, wertvolle Review-Daten im Rahmen der allgemeinen Praktiken, z.B. GP 2.8 Überwachung und Steuerung des Prozesses, weiterhin gesammelt und verwendet werden.

4 TMMi-Stufe 4 Gemessen

Wie bereits in diesem Dokument erwähnt, ist es wichtig, sich daran zu erinnern, dass zwischen TMMi-Stufe 3 und TMMi-Stufen 4 und 5 ein natürlicher Bruch besteht. Es gibt nach wie vor erhebliche Kontroversen darüber, ob es sinnvoll ist, eine Organisation auf TMMi-Stufe 4 und 5 zu bringen. Insbesondere agilen Organisationen wird empfohlen, die Praktiken der TMMi-Stufen 4 und 5 kritisch auszuwählen und nur solche durchzuführen, die von Bedeutung sind und einen Mehrwert bieten. Das TMMi ist zwar umfassend, aber um erfolgreich zu sein, müssen Unternehmen die wichtigsten Testverfahren und Verbesserungen identifizieren, auf die sie sich konzentrieren müssen. Interessanterweise hat Jeff Sutherland, Mitbegründer von Scrum, ein Papier über den Wert der Verwendung von Scrum zusammen mit Praktiken höherer Reifegrade veröffentlicht [Sutherland *et al*]. Im Folgenden werden Informationen zu den TMMi-Stufen 4 und 5 gegeben, wie man den Wert dieser höheren TMMi-Stufen und ihrer Praktiken gewinnen kann, indem man sie weniger formell zusammen mit einem agilen Ansatz einsetzt.

Erstens kann man im Gegensatz zu dem, was gesagt wurde, auch in Erwägung ziehen, TMMi-Praktiken der Stufen 4 und 5 bereits selektiv mit agilen Techniken einzusetzen, um wichtige Geschäftsziele zu erreichen, während man versucht, TMMi-Stufe 2 oder 3 **zu erreichen**. Sie brauchen nicht zu warten, und das sollten Sie auch nicht. Seien Sie flexibel und bleiben Sie nicht bei den Testverfahren der TMMi-Stufen 2 und 3 hängen. Nutzen Sie das Modell auf der Grundlage Ihrer Geschäftstreiber und verwenden Sie die Prozessbereiche, Ziele und Praktiken, die den größten Wert haben. Verwenden Sie TMMi in gewissem Maße kontinuierlich und legen Sie die Reifegrade nicht zu strikt fest.

4.1 Prozessbereich 4.1 Messung von Tests

Der Zweck der Messung von Tests besteht darin, Messungen zu identifizieren, zu sammeln, zu analysieren und anzuwenden, um eine Organisation bei der objektiven Bewertung der Effektivität und Effizienz des Testprozesses, der Produktivität des Testpersonals, der resultierenden Produktqualität und der Ergebnisse der Testverbesserung zu unterstützen. Als solches wird die Testorganisation eine Testmessungsfähigkeit entwickeln und aufrechterhalten, die zur Unterstützung des Informationsbedarfs des Managements eingesetzt wird.

Agile Ansätze unterstützen das Engagement derjenigen, die die Arbeit in den Projekten durchführen und genau messen müssen. Allerdings bieten agile Ansätze in der Regel nicht viel Unterstützung, wenn es darum geht, sich zurückzuziehen, um sicherzustellen, dass die richtigen Messgrößen in der gesamten Organisation verwendet werden, und um zu gewährleisten, dass diese Messgrößen mit den aus den Geschäftsanforderungen der Organisation abgeleiteten Messzielen in Einklang stehen. Dies sind die Stärken, die der TMMi-Prozessbereich Messung von Tests in agilen Organisationen einbringen kann.

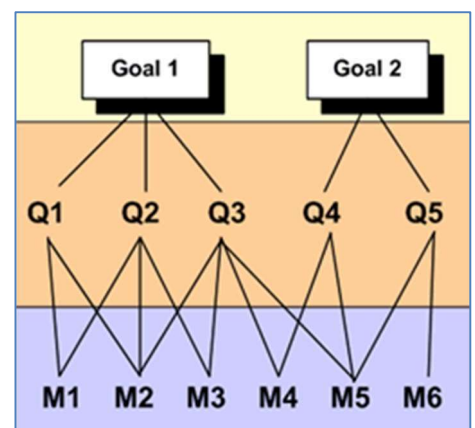
4.1.1 SG 1 Angleichung von Messung von Tests und Analyseaktivitäten

Allzu oft scheinen Messprogramme darauf abzuzielen, so viele Daten wie möglich zu sammeln. Ein Messprogramm sollte sehr zielgerichtet sein, es sollten nur die Dinge erfasst werden, die wichtig sind. Beginnen Sie damit, klare Ziele für das Messprogramm zu definieren und diese regelmäßig zu überprüfen, z. B. "Es sollte uns ermöglichen, in unseren Projekten erfolgreicher zu testen". Diskutieren Sie dann mit den Beteiligten, welche Testmetriken diese Ziele wirklich unterstützen. Auf der Grundlage dieses Brainstormings wird ein begrenzter Satz von Kernmetriken definiert.

Goal-Question-Metric (GQM) [Van Solingen und Berghout] ist eine äußerst praktische Methode, die diesen Ansatz unterstützt. Spezifische kontextrelevante Messgrößen werden auf der Grundlage der Geschäftsanforderungen benötigt. Standardmetriken des Unternehmens sind für eine echte (Test-)Prozessverbesserung meist unzureichend. Die

Der Zweck von Messungen ist es, Entscheidungen zu lenken. Setzen Sie kleine, befähigte Teams ein, um aussagekräftige Maßnahmen abzuleiten, die dann in kurzen Zyklen überprüft und verfeinert werden.

Mit einem fokussierten Testmessprogramm, das vom (geschäftlichen) Mehrwert ausgeht, steht der TMMi-Prozessbereich Messung von Tests ganz im Einklang mit dem agilen Prinzip der Einfachheit. Stellen Sie sicher, dass



es innerhalb der Organisation eine schlanke Strategie für die Messung von Tests und Datenerfassung gibt. Es ist erwähnenswert, dass man ein Testmessungsprogramm entwickeln kann, indem man mit einigen wenigen fokussierten Messzielen und einigen Schlüsselmaßnahmen beginnt und diese später erweitert oder modifiziert, wenn sich die Geschäftsanforderungen entwickeln und ändern.

Beachten Sie, dass im Rahmen von Agilität der Schwerpunkt mehr auf Teamarbeit und Systemdenken liegt. Dies kann dazu führen, dass einige der Metriken auf das Team und das Gesamtsystem ausgeweitet werden, anstatt sich nur auf die Besonderheiten des Testens selbst zu beschränken. Dies kann durchaus zu mehr Herausforderungen in der Phase der Messanalyse führen.

Natürlich hängt es von den Messzielen ab, welche Metriken definiert werden und für wen die Daten gesammelt werden, aber einige Beispiele für typische Testmetriken für agile Teams sind:

- Fehlerzykluszeit: Agile Teams sollten sich bemühen, Fehler so schnell wie möglich zu beheben. Tatsächlich besteht eines der Hauptziele des kollaborativen agilen Ansatzes darin, Fehler schneller zu beheben, damit die Software früher freigegeben werden kann.
- Defect spill-over (Anzahl der Defekte, die auf ein zukünftiges Release verschoben werden); agile Teams haben das Ziel, mit jeder Iteration funktionierende Software zu produzieren. Defect Spillover misst die Fehler, die während einer bestimmten Iteration oder eines Sprints nicht behoben werden, indem einfach die am Ende jedes Sprints oder jeder Iteration verbleibenden Fehler gezählt werden. In einigen agilen Organisationen wird dies als technische Schuld bezeichnet.
- Anzahl der während einer Iteration gefundenen Fehler
- Anzahl der nach der Freigabe an die Produktion/Kunden festgestellten Mängel
- Anzahl der Fehler, die von Personen außerhalb des Teams gefunden wurden
- Anzahl der Supportanfragen von Kunden
- Prozentsatz der automatisierten Testabdeckung
- Prozentsatz der Codeabdeckung.

4.1.2 SG 2 Testmessergebnis bereitstellen

Sobald die relevanten Maßnahmen auf der Grundlage des Geschäftsbedarfs definiert sind, sind die spezifischen Praktiken innerhalb dieses spezifischen Ziels in der Regel alle weitgehend auf die gleiche Weise anwendbar wie in einer traditionellen Organisation. Der einzige Unterschied besteht darin, dass dies mit einer anderen Denkweise geschieht, um die Praktiken so schlank wie möglich zu halten.

Die erforderlichen Testmessdaten werden gesammelt und auf Vollständigkeit und Integrität geprüft, anschließend werden die Daten wie geplant analysiert und die Ergebnisse an alle relevanten Beteiligten übermittelt.

4.2 Prozessbereich 4.2 Bewertung der Produktqualität

Der Zweck der Bewertung der Produktqualität besteht darin, ein quantitatives Verständnis der Produktqualität zu entwickeln und dadurch die Erreichung der Produktqualitätsziele bestimmter Projekte zu unterstützen.

4.2.1 SG 1 Es werden messbare Projektziele für die Produktqualität und deren Prioritäten festgelegt

Im Grunde genommen gibt es keinen wirklichen Unterschied zwischen der Ermittlung und Festlegung messbarer, priorisierter Ziele für die Produktqualität in einem agilen Kontext und der Vorgehensweise in einer traditionellen Umgebung. Das übergeordnete Ziel besteht darin, dazu beizutragen, die Bedürfnisse und Wünsche der Kunden und Endnutzer nach qualitativ hochwertigen Produkten zu erfüllen. Es gibt mehrere Möglichkeiten, dies in einem agilen Kontext zu tun. Einige verwenden Definition of Done, um messbare Ziele für die Produktqualität zu definieren, andere verwenden Features, Epics oder User Storys, um sie zu definieren, wobei sie auch Akzeptanzkriterien verwenden, um sie messbar zu machen. Die Wahl (Arbeitsweise) hängt unter anderem von der Art des Produktqualitätsziels ab. Einen Überblick über die Qualitätsattribute finden Sie in [ISO25010].

Definition of Done (DoD) ist eine umfassende Checkliste notwendiger Teamaktivitäten und/oder Arbeitsprodukte, die sicherstellen, dass nur wirklich fertige User Stories geliefert werden, und zwar nicht nur in Bezug auf die Funktionalität, sondern auch in Bezug auf die Produktqualität. Der Vorteil der Festlegung von

Produktqualitätszielen innerhalb des DoD besteht darin, dass sie für das gesamte Team bei der Entwicklung der Inkremente sichtbar und damit umsetzbar bleiben. Es liegt auf der Hand, dass die Qualitätsziele global auf die Anwendung anwendbar sein müssen, damit sie in der DoD erwähnt werden können. Wenn die Qualitätsziele nicht global anwendbar sind, ist es wahrscheinlich die beste Alternative, sie als Feature oder Epic zu definieren.

Manchmal wird zunächst "nur" eine übergeordnete Liste relevanter Produktqualitätsattribute auf Release-Ebene festgelegt. Diese werden während der Verfeinerungssitzung schrittweise als Feature, Epic oder User Story ausgearbeitet. Anschließend überlegt das Team zu Beginn einer Iteration, ob eines oder mehrere der festgelegten Produktqualitätsziele Teil des Ziels für diese spezifische Iteration sein werden.

4.2.2 SG 2 Der tatsächliche Fortschritt bei der Erreichung der Produktqualitätsziele des Projekts wird quantifiziert und gemanagt.

Sobald die quantitativen Produktqualitätsziele festgelegt sind, werden sie überwacht, kontrolliert und bei Bedarf angepasst. Da die Produktqualität als Teil der DoD und/oder als spezifische Features, Epics oder User Stories mit messbaren Akzeptanzkriterien definiert wird, können für die Verfolgung und das Management des Fortschritts bei der Erreichung der Produktqualitätsziele dieselben Methoden und Techniken verwendet werden, die bereits im Prozessbereich 2.3 Testüberwachung und -steuerung auf TMMi-Stufe 2 erläutert und beschrieben wurden. Der Fortschritt wird täglich gemessen und im Rahmen des täglichen Standup-Meetings besprochen, in dem über den Fortschritt, die Qualität und das Risiko berichtet wird. Darüber hinaus wird die Bewertung der Produktqualität in der Regel auch durch den Prozessbereich Messung von Tests unterstützt, der eine Messinfrastruktur bereitstellt.

4.3 Prozessbereich 4.3 Fortgeschrittene Reviews

Der Zweck der Fortgeschrittenen Reviews, die auf den Praktiken des TMMi-Level-3-Prozessbereichs Peer-Reviews aufbauen, besteht darin, die Produktqualität in einem frühen Stadium des Lebenszyklus zu messen und die Teststrategie und den Testansatz zu verbessern, indem Peer-Reviews (statische Tests) mit dynamischen Tests verbunden werden.

4.3.1 SG 1 Koordinierung des Peer-Review-Ansatzes mit dem dynamischen Testansatz

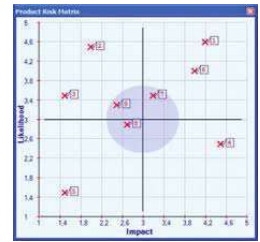
Während in traditionellen Projekten die statischen Tests, z. B. Reviews, meist nicht mit den dynamischen Tests eines unabhängigen Testteams verknüpft sind, wird erwartet, dass dies in einem agilen Kontext völlig anders ist. In einer agilen Umgebung sollte der Ansatz zur Sicherstellung der Produktqualität so aussehen, dass alle testbezogenen Aktivitäten, sowohl statische als auch dynamische, als ein kohärenter und koordinierter Ansatz organisiert werden. Es liegt natürlich in der Verantwortung des agilen Teams, statische und dynamische Tests zu integrieren und die optimale Mischung zu finden. Es wird erwartet, dass das agile Team die Führung bei der Festlegung eines kohärenten und effektiven integrierten Ansatzes übernimmt. Der Ansatz für statische Tests (Peer-Review) wird daher mit dem Ansatz für dynamische Tests abgestimmt und koordiniert.

4.3.2 SG 2 Messung der Produktqualität in einem frühen Stadium des Lebenszyklus durch Peer-Reviews

Auf TMMi-Stufe 4 setzt die Organisation quantitative Ziele für Softwareprodukte und damit verbundene Arbeitsprodukte. Peer-Reviews spielen eine wesentliche Rolle bei der Erreichung dieser Ziele. Während auf TMMi-Stufe 3 Peer-Reviews hauptsächlich durchgeführt werden, um Fehler zu finden, liegt der Schwerpunkt jetzt auf der Messung der Produkt-/Dokumentenqualität, um die Produktqualität in einem frühen Stadium des Lebenszyklus zu kontrollieren. Die Überprüfungspraktiken werden erweitert und umfassen nun auch Praktiken wie Stichproben, die Anwendung von Endekriterien und das Vorschreiben von Regeln. Tom Gilb bezeichnet dies als agile Inspektion, bei der es darum geht, den Schwerpunkt von der Bereinigung auf Stichproben, Messungen, Motivation und Fehlerprävention zu verlagern [Gilb]. Dies ist eindeutig eine Praxis, die zu einer agilen Organisation mit hohem Reifegrad passt. Ein weiteres Beispiel wäre die Verwendung der INVEST-Regeln [Wake] für User Stories als Definition of Ready (DoR) und die Messung der Qualität der User Stories durch Überprüfung anhand dieser Regeln. Eine "Definition of Ready" bedeutet, dass die Geschichten sofort umsetzbar sein müssen.

4.3.3 SG 3 Anpassung des Testansatzes in einem frühen Stadium des Lebenszyklus auf Basis der Reviewergebnisse

Wenn Peer-Review-Ergebnisse und dynamische Tests koordiniert werden, können frühe Review-Ergebnisse und Daten Produktrisiken und den Testansatz beeinflussen. Wie bereits unter dem spezifischen Ziel 1 dieses Prozessbereichs erwähnt, ist Qualität bei Agilität eine Teamleistung, und die Ergebnisse der auf Verifizierung und Validierung basierenden Aktivitäten werden in Teambesprechungen diskutiert. Das Prinzip, frühe Testergebnisse zu nutzen, um nachfolgende Testaktivitäten zu beeinflussen, passt daher nahezu perfekt zur Arbeitsweise in agilen Teams und Projekten.



5 TMMi-Stufe 5 Optimierung

Die Ziele und Praktiken der TMMi-Stufe 5 leiten sich von Qualitätsmanagement- und Prozessverbesserungsansätzen mit hohem Reifegrad ab. Bei diesem hohen Reifegrad sind die Ziele und Praktiken nahezu unabhängig vom angewandten Software-Lebenszyklus. Natürlich wird eine schlanke, auf Agilität basierende Organisation die Dinge anders angehen als herkömmliche, hierarchisch aufgebaute Organisationen, aber die wesentlichen Praktiken rund um Qualitätskontrolle, Prozessoptimierung und Fehlerprävention werden weitgehend unverändert bleiben. Der Grund dafür ist, dass diese Praktiken als Qualitätsmanagement-Praktiken angesehen werden, die in der Regel auf einer höheren Ebene als der operativen Produktentwicklungsebene durchgeführt werden. Es handelt sich also nicht um Praktiken, die direkt mit den primären Aktivitäten der Softwareentwicklung und/oder des Testens innerhalb eines Projekts oder eines Produktlieferstroms zusammenhängen.

5.1 Prozessbereich 5.1 Fehlerprävention

Der Zweck der Fehlerprävention besteht darin, gemeinsame Fehlerursachen über den gesamten Entwicklungszyklus hinweg zu identifizieren und zu analysieren und Maßnahmen zu definieren, die das Auftreten ähnlicher Fehler in der Zukunft verhindern.

In agilen Organisationen liegt der Schwerpunkt in der Regel bereits auf der Fehlerprävention, was eine wichtige Grundlage darstellt, auf der die spezifischen Ziele und Praktiken dieses Prozessbereichs aufgebaut werden können. Effiziente agile Organisationen konzentrieren sich auf die Fehlerprävention und nicht nur auf die Fehlerentdeckung. Zu einer Kultur der Fehlerprävention gehört die Trennung von "work in process"-Fehlern von Fehlern, die über die Iteration hinausgehen, in der die User Stories entwickelt werden.

Die agile Retrospektive ist eine Praxis für Teams, um zu reflektieren, zu lernen und ihre Arbeit kontinuierlich zu verbessern. Obwohl Retrospektiven meist dazu verwendet werden, die aktuelle Arbeitsweise zu untersuchen, können sie auch dazu genutzt werden, Qualitätsprobleme zu untersuchen oder Maßnahmen zu vereinbaren, die die Qualität der gelieferten Software verbessern können.

Der Prozessbereich Fehlerprävention befasst sich mit den Praktiken zur Identifizierung und Analyse gemeinsamer Fehlerursachen in verschiedenen Teams, Produkten und Wertströmen sowie mit der Festlegung spezifischer Maßnahmen zur Beseitigung der gemeinsamen Ursachen für diese Art von Fehlern in der Zukunft. Alle Fehler, ob sie während der Entwicklung, beim Testen oder im Feld gefunden werden, fallen in den Geltungsbereich dieses Prozessbereichs. Da für die Fehlerprävention auf dieser Ebene Messdaten erforderlich sind, baut die Fehlerprävention auf den Messverfahren der TMMi-Stufe 4 und den verfügbaren Messdaten zu Entwicklung, Tests und Produktqualität auf.

Es gibt grundsätzlich zwei verschiedene Arten von Fehlerpräventionsaktivitäten in agilen Organisationen: die Teamebene und die teamübergreifende Ebene. Die teambezogenen Fehlerpräventionsaktivitäten liegen in der Verantwortung des Teams und wurden bereits auf TMMi-Stufe 3 behandelt. Die teamübergreifenden Fehlerpräventionsaktivitäten werden von einer Testprozessgruppe oder Testgilde durchgeführt und sind diejenigen, die speziell in diesem Prozessbereich behandelt werden...

5.1.1 SG 1 Gemeinsame Ursachen von Fehlern ermitteln

Im Rahmen von Agilität finden Verbesserungen, einschließlich Verbesserungen des Testprozesses, im Allgemeinen auf der Grundlage häufiger Feedback-Schleifen statt. Da der Umfang oft auf die vorherige Iteration beschränkt ist, werden kleine, aber häufige Verbesserungen vorgenommen, die sich hauptsächlich auf die Lösung spezifischer lokaler Probleme konzentrieren. Der Schwerpunkt dieser Verbesserungen liegt häufig nicht auf dem teamübergreifenden Lernen und der Institutionalisierung von Verbesserungen. Probleme nur auf Teamebene zu lösen, kann auch leicht zu einer Suboptimierung führen und den Blick für das große Ganze verlieren. Hier schafft dieser TMMi-Prozessbereich einen Mehrwert. Aufbauend auf den bestehenden Agile-Praktiken wird die Fehlerprävention nun auch auf organisatorischer Ebene (über Teams, Produkte und Wertströme hinweg) angewandt und typischerweise von einer (Test-)Prozessgruppe oder Testgilde verwaltet (siehe Abschnitt 3.1.1).

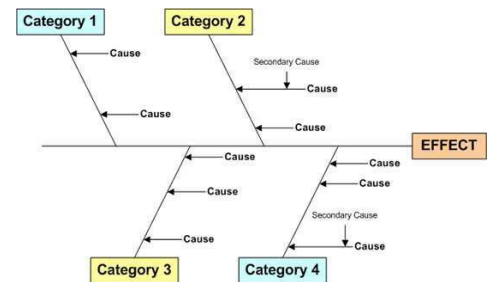
Die Auswahl der zu analysierenden Fehler sollte auf der Grundlage verschiedener Faktoren erfolgen, darunter Risiko und Geschäftswert. Der Schwerpunkt muss auf die Bereiche gelegt werden, in denen die Fehlerprävention den

größten Mehrwert bringt (in der Regel in Form von geringeren Kosten oder Risiken) und/oder in denen die Fehler am kritischsten sind. Obwohl der Prozessbereich der Fehlerprävention üblicherweise für Defekte verwendet wird, kann er auch für Probleme wie z.B. Geschwindigkeitsprobleme verwendet werden.

Obwohl eine Testprozessgruppe die Aktivitäten zur Fehlerprävention koordinieren kann, sollte dies durch ein multidisziplinäres Team geschehen, z. B. mit Vertretern aus dem Requirements Engineering, dem System Engineering und/oder der Softwareentwicklung, da Verbesserungsmaßnahmen oft andere Disziplinen betreffen. Natürlich ist dies bei agilen Liefermethoden und Teams bereits üblich. Stellen Sie jedoch sicher, dass alle erforderlichen Disziplinen an den Verbesserungsmaßnahmen beteiligt sind.

Analytische Methoden zur Bewertung der Fehlerursachen, wie z.B. Ursache-Wirkungs-Diagramme, Fishbone-Diagramme oder die 5 Warum's, werden typischerweise zur Fehlerprävention eingesetzt und sind bereits gängige Agile-Praktiken.

Alle drei spezifischen Praktiken (1.1 Definieren von Parametern für die Fehlerauswahl, 1.2 Auswählen von Fehlern für die Analyse und 1.3 Analysieren der Haupt- und gemeinsamen Ursachen ausgewählter Fehler), die im TMMi-Modell für dieses spezifische Ziel definiert sind, sind auch auf eine agile Organisation anwendbar.



5.1.2 SG 2 Priorisierung und Festlegung von Maßnahmen zur systematischen Beseitigung häufiger Fehlerursachen

Es gibt keine wirklichen Unterschiede bei der Durchführung der spezifischen Praktiken im Rahmen dieses spezifischen Ziels, unabhängig davon, ob sie in einem agilen oder sequenziellen Lebenszyklus durchgeführt werden. Natürlich werden die vorgeschlagene Lösung und der anschließende Verbesserungsvorschlag anders aussehen, da sie zur agilen Arbeitsweise passen müssen, z. B. durch die Erstellung zusätzlicher Backlog-Elemente anstelle von formalisierten Aktionsvorschlägen mit Genehmigungen. Es werden geeignete Maßnahmen ergriffen, um die Wahrscheinlichkeit zu minimieren, dass das Problem wieder auftritt. Oft bedeutet dies eine Kombination aus Prozessverbesserung und zusätzlichem Mentoring und/oder Training für Teammitglieder und andere Mitarbeiter. Man muss sowohl Menschen als auch Prozessprobleme auf integrierte Weise angehen, anstatt zu versuchen, sie künstlich zu trennen.

Auch agile Organisationen beruhen auf selbstverwalteten Teams. Dies bedeutet, dass die Teams selbst entscheiden können, welche Verbesserungen sie einführen oder ablehnen. Daher muss bei der Priorisierung und Festlegung von Maßnahmen eine angemessene Vertretung der Teams einbezogen werden, um zu diskutieren, ob diese Verbesserungen tatsächlich einen Mehrwert für sie darstellen.

Auch die Art und Weise, wie die Verbesserungen umgesetzt werden, wird anders sein, weniger von oben nach unten und mehr wie ein Angebot auf freiwilliger Basis für die selbstverwalteten Teams. Mehr über den Einsatz von Testverbesserungen finden Sie im Prozessbereich Testprozessoptimierung weiter unten.

5.2 Prozessbereich 5.2 Qualitätssteuerung

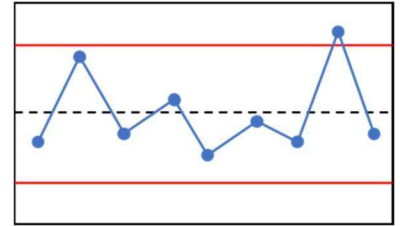
Der Zweck der Qualitätssteuerung ist die statistische Lenkung und Steuerung des Testprozesses. Auf dieser Stufe ist die Leistung des Testprozesses vollständig vorhersagbar und innerhalb akzeptabler Grenzen stabilisiert. Die Prüfungen auf Projektebene werden mit statistischen Methoden auf der Grundlage repräsentativer Stichproben durchgeführt, um die Produktqualität vorherzusagen und die Prüfungen effizienter zu gestalten.

5.2.1 SG 1 Etablierung eines statistisch gesteuerten Testprozesses

Die Grundsätze und Grundlagen für die Qualitätskontrolle sind unabhängig von der angewandten Lebenszyklusmethode dieselben. Die Prozessqualitätssteuerung umfasst die Festlegung von Zielen für die Leistung des Standardtestprozesses. Auf der Grundlage der von den Teams vorgenommenen Messungen der Testprozessleistung werden Analysen durchgeführt und Anpassungen vorgenommen, um die Testprozessleistung innerhalb akzeptabler Grenzen zu halten. Wenn sich die Leistung des Testprozesses innerhalb akzeptabler Grenzen stabilisiert hat, werden der definierte Testprozess, die zugehörigen Messungen und die akzeptablen Grenzwerte für

die Messungen als Basislinie festgelegt und zur Steuerung des Testprozesses verwendet.

Leistung statistisch zu erfassen. Die Testprozessfähigkeit des Standardtestprozesses der Organisation, d.h. die Testprozessleistung, die ein Team erwarten kann, ist nun vollständig verstanden und bekannt. Infolgedessen kann in einem Team frühzeitig und konsequent auf Abweichungen von diesen Erwartungen reagiert werden, um sicherzustellen, dass die Leistung innerhalb der akzeptablen Grenzen liegt.



Es ist wichtig, Teilprozesse für die statistische Kontrolle auszuwählen, die für das Unternehmen kritisch sind. Unternehmen haben in der Vergangenheit Probleme erlebt. Die Erfahrung zeigt, dass die Teilprozesse, die diese Kriterien am besten erfüllen, meist nicht auf einzelne Teams beschränkt sind, sondern mehrere Teams, Projekte und Produktlieferströme umfassen. Bei der Anwendung agiler Methoden und Techniken wird die Arbeit in kurze Iterationen aufgeteilt, wobei die Arbeit in jeder Iteration die Durchführung mehrerer Entwicklungs- und Testaktivitäten umfasst. Daher unterstützt die agile Entwicklung effektiver als die traditionelle Entwicklung die Überwachung der wertvollsten Teilprozesse für die statistische Kontrolle, da die Zyklen viel kürzer sind, was zu fast sofortigem Feedback über die Leistung des Testprozesses führt. Schnelles Feedback ist in der Regel sehr wertvoll für Teams und ein wichtiges agiles Prinzip.

5.2.2 SG 2 Nutzung statistischer Methoden für Tests

Die Produktqualitätssteuerung stützt sich auf Betriebsprofile [Musa] und Nutzungsmodelle des Produkts in seiner vorgesehenen Umgebung, um statistisch gültige Rückschlüsse zu ziehen, die zu einer repräsentativen Stichprobe von Tests führen. Dieser Ansatz, der besonders auf Systemebene nützlich ist, verwendet statistische Testmethoden, um die Produktqualität auf der Grundlage dieser repräsentativen Stichprobe vorherzusagen. Mit anderen Worten: Beim Testen einer Teilmenge aller möglichen Verwendungen, die durch das Verwendungs- oder Betriebsprofil repräsentiert werden, können die Testergebnisse als Grundlage für Schlussfolgerungen über die Gesamtleistung des Produkts dienen.

Dies ist ein Ansatz, der gut zur Agilität passt, da er von Natur aus schlank ist und auf Rückmeldungen aus der betrieblichen Nutzung und aus Überprüfungssitzungen zum Produkt beruht. Allerdings muss eine agile Organisation über den richtigen Reifegrad verfügen, um diesen Ansatz anwenden zu können. Außerdem ist er stark auf Messungen und Daten angewiesen, die gesammelt werden und verfügbar sind.

Eine Organisation, die statistische Methoden beim Testen einsetzt, ist in der Lage, Konfidenzniveau und Vertrauenswürdigkeit zu quantifizieren (weitere Informationen zu diesen Begriffen finden Sie im TMMi-Rahmenwerk). Sowohl das Konfidenzniveau als auch die Vertrauenswürdigkeit werden in der Regel als Definition-of-Done bei der Anwendung statistischer Tests verwendet.

5.3 Prozessbereich 5.3 Testprozessoptimierung

Der Zweck der Testprozessoptimierung ist die kontinuierliche Verbesserung der bestehenden Testprozesse in der Organisation und die Identifizierung neuer Testtechnologien (z.B. Testwerkzeuge oder Testmethoden), die geeignet sein könnten, und deren geordnete Einführung in die Organisation. Die Testprozessoptimierung unterstützt auch die Wiederverwendung von Testressourcen innerhalb des Unternehmens. Die Verbesserungen unterstützen die Ziele der Organisation in Bezug auf Produktqualität und Testprozessleistung, die sich aus den Geschäftszielen der Organisation ableiten.

5.3.1 SG 1 Auswahl von Testprozessverbesserungen

Es werden Vorschläge zur Testverbesserung gesammelt und ausgewertet. Die Testverbesserungsvorschläge stammen aus internen Quellen und sind typischerweise team-, projekt- und produktübergreifend. Zu den internen Quellen gehören Probleme und Ideen, die aus Aktivitäten zur Fehlerprävention, Teambesprechungen, Retrospektiven, Kapiteln und Gilden stammen. Nach einer Kosten-Nutzen-Analyse werden die interessantesten Testverbesserungsvorschläge in Pilotprojekten erprobt, um neue und unbewiesene größere Änderungen zu bewerten, bevor sie gegebenenfalls in der gesamten Organisation in den Teams eingeführt werden. Schließlich wird eine Auswahl von Testverbesserungen getroffen, die in der gesamten Organisation eingesetzt werden sollen.

5.3.2 SG 2 Evaluation neuer Testtechnologien zur Ermittlung ihrer Auswirkungen auf den Testprozess

Im Rahmen der Testprozessoptimierung sucht die zentral organisierte Testprozessgruppe (oder Testgilde) proaktiv auf dem Markt nach neuen Technologien, wie z. B. Tools, Methoden, Techniken oder technischen Innovationen, die die Testfähigkeit der agilen Teams verbessern können. Indem sie das Bewusstsein für testbezogene technologische Innovationen aufrechterhält und diese systematisch evaluiert und ausprobiert, wählt die Organisation in Absprache mit den Teamvertretern geeignete Testtechnologien aus, um die Qualität ihrer Produkte und die Produktivität ihrer Testaktivitäten zu verbessern. Innerhalb der Teams werden Pilotprojekte durchgeführt, um neue und unerprobte Prüftechnologien zu bewerten, bevor sie in die Standardpraxis aufgenommen werden.

5.3.3 SG 3 Einsatz von Testverbesserungen

Testverbesserungen und geeignete neue Testtechnologien werden in den agilen Teams eingesetzt, um das Testen zu verbessern. Ihr Nutzen wird gemessen und Informationen über neue Innovationen werden in der gesamten Organisation verbreitet. Natürlich gibt es einen Unterschied zwischen dem Einsatz in einer traditionellen und einer agilen Organisation. In einer traditionellen Organisation führt der Einsatz meist zu neuen Praktiken, die von allen Beteiligten angewendet werden sollen. In einer agilen Organisation hat das Team mehr Autonomie und verwaltet sich selbst. Dies bedeutet auch, dass der Bereitstellungsprozess anders ist. Testverbesserungen und geeignete neue Testtechnologien werden in den organisatorischen Test integriert, Prozess-Assets, Schulungen und Unterstützung werden organisiert und angeboten. Letztendlich sollte jedoch das agile Team selbst entscheiden, ob die Testverbesserung Teil seiner Arbeitsweise werden soll oder nicht.

5.3.4 SG 4 Wiederverwendung von hochwertigen Test-Assets

Auf TMMi-Stufe 3 kann bereits eine gewisse Wiederverwendung von Testmitteln über Teams, Projekte und Produktströme hinweg stattfinden; auf TMMi-Stufe 5 wird die Wiederverwendung von Testressourcen jedoch zu einem wichtigen Ziel. Das Team sollte das Rad nicht neu erfinden, sondern auf das bereits vorhandene Fachwissen und die vorhandenen Ressourcen zurückgreifen und so Aufwand und Zeit sparen. Qualitativ hochwertige Testressourcen, die wahrscheinlich in der gesamten Organisation wiederverwendet werden können, werden identifiziert, z. B. bei retrospektiven Treffen, Lessons Learned-Sitzungen, Diskussionen innerhalb der Testgilde oder Testbewertungen. Anschließend werden wiederverwendbare Test-Assets ausgewählt und dem zentralen Test-Repository in einem Format hinzugefügt, das für die Wiederverwendung in zukünftigen Produktlieferungen und Projekten modifiziert werden kann.

6 Überblick Anwendbarkeit TMMi-spezifische Ziele und Praktiken

6.1 TMMi Assessments

Um Lead-Assessoren und Assessoren bei ihren Assessments zu unterstützen, wird in diesem Kapitel ein Überblick über die Anwendbarkeit von Prozessbereichen, spezifischen Zielen und spezifischen Praktiken in einem agilen Kontext gegeben. In den vorangegangenen Kapiteln wurde eine begrenzte Anzahl spezifischer Praktiken identifiziert, von denen angenommen wird, dass sie weniger oder vielleicht sogar gar nicht relevant sind. Diese spezifischen Praktiken werden im Folgenden aufgeführt. Während eines TMMi-Assessments muss ein Lead-Assessor oder Assessor diese Punkte explizit auf ihre Relevanz für den spezifischen agilen Kontext hin bewerten. Falls entschieden wird, dass ein Punkt tatsächlich weniger oder gar nicht relevant ist, wird erwartet, dass dies sowohl in den Prozessen der Organisation als auch im Assessment-Bericht klar mit Begründung dokumentiert wird.

6.2 TMMi-Stufe 2 Gelenkt

Prozessbereich 2.2 Testplanung

SG2 Festlegung eines Testansatzes	Anwendbar
- SP2.3 Definieren Sie Eingangskriterien	Wahrscheinlich weniger/nicht relevant
- SP2.5 Definition von Kriterien für die Aussetzung und Wiederaufnahme	Wahrscheinlich weniger/nicht relevant
SG3 Erstellung von Testschätzungen	Anwendbar
- SP3.2 Definition des Testlebenszyklus	Wahrscheinlich weniger/nicht relevant
SG5 Einholung der Zustimmung zum Testkonzept	Anwendbar
- SP5.2 Abgleich von Arbeit und Ressourcen	Wahrscheinlich weniger/nicht relevant

Prozessbereich 2.3 Testüberwachung und Teststeuerung

SG2 Überwachung der Produktqualität anhand des Plans und der Erwartungen	Anwendbar
- SP2.3 Eingangskriterien überwachen	Wahrscheinlich weniger/nicht relevant
- SP2.5 Kriterien für die Aussetzung und Wiederaufnahme überwachen	Wahrscheinlich weniger/nicht relevant

Prozessbereich 2.4 Testentwurf und Testdurchführung

SG2 Testdelasierung durchführen	Anwendbar
- SP2.3 Spezifizieren Sie den Smoke-test	Wahrscheinlich weniger/nicht relevant
- SP2.4 Entwicklung eines Zeitplans für die Testdurchführung	Wahrscheinlich weniger/nicht relevant
SG3 Testdurchführung ausführen	Anwendbar
- SP3.1 Durchführen von Smoketests	Wahrscheinlich weniger/nicht relevant

6.3 TMMi-Stufe 3 Definiert

Prozessbereich 3.5 Peer-Reviews

SG2 Durchführung von Peer-Reviews	Anwendbar
- SP2.3 Analyse von Peer-Review-Daten	Wahrscheinlich weniger/nicht relevant

6.4 TMMi-Stufe 4 Gemessen

Alle Prozessbereiche, spezifischen Ziele und spezifischen Praktiken von TMMi Stufe 4 sind anwendbar. Keine spezifischen Praktiken werden als wahrscheinlich weniger/nicht relevant angegeben.

6.5 TMMi-Stufe 5 Optimierung

Alle Prozessbereiche, spezifischen Ziele und spezifischen Praktiken der TMMi Stufe 5 sind anwendbar. Keine spezifischen Praktiken werden als wahrscheinlich weniger/nicht relevant angegeben.

Referenzen

- [Black, Van Veenendaal] Rex Black, Erik van Veenendaal und Dorothy Graham (2012), *Foundations of Software Testing ISTQB Certification* (3. Auflage), Cengage
- [Cohn] M. Cohn (2009), *Succeeding with Agile: Software Development using Scrum*, Addison-Wesley
- [Galen] R. Galen (2015), *Three Pillars of Agile Quality and Testing: Achieving Balanced Results in your Journey Towards Agile Quality*, RGCG, LLC
- [Gilb] T. Gilb (2004), *Agile Specification Quality Control*, International Council on Systems Engineering (INCOSE)
- [ISO25010] ISO/IEC 25010 (2011), *Systems and Software Engineering - Systems and Software Quality Requirements and Evaluation (SQuaRE) System and Software Quality Models*, Internationale Organisation für Normung
- [Sutherland et al] C.R. Jakobsen, K. Johnson und J. Sutherland (2007), *Scrum and CMMI level 5: The Magic Portion for Code Warriors* in: IEEE Agile 2007 Konferenz, Washington, D.C.
- [McMahon] Paul. E. McMahon (2011), *Integrating CMMI with Agile Development - Case Studies and Proven Techniques for Faster Performance Improvement*, Addison Wesley
- [Musa] J. Musa (1998), *Software Reliability Engineering Testing*, McGraw-Hill Education
- [Van der Aalst und Davis] L. van der Aalst und Cecile Davis (2013), *TMap NEXT in Scrum - Effektives Testen in agilen Projekten*, Sogeti Nederland B.V.
- [Van Solingen und Berghout] R. van Solingen und E. Berghout (1999), *The Goal/Question/Metric method*, McGrawHill
- [van Veenendaal] E. van Veenendaal (2014), *PRISMA: Product Risk Assessment for Agile Projects*, in: *Testing Experience*, Ausgabe 04/12, Dezember 2012
- [Wake] B. Wake (2003), *INVEST in Good Stories, and SMART Tasks*, xp123.com/articles/invest-in-good-stories-and-smart-tasks