



Wie Agilität und Prozessreifegrad-Modelle zusammenpassen

Geht doch!

**Christian Brandes,
Thomas Roßner**

Agile Methoden sind erfolgreich, da sich mit ihnen über flexible und minimalistische Prozesse gute Software erzeugen lässt. Prozessreifegrad-Modelle wie CMMI oder SPICE scheinen aber gerade auf stringente, schwergewichtige Prozesse viel Wert zu legen. Kommt ein agiles Projekt bei der Beurteilung durch ein solches Modell deshalb zwangsläufig „schlecht weg“? Sollen agile Entwickler einen Bogen um diese machen?

Agiles Projektmanagement ist über zehn Jahre nach Erscheinen des „Agilen Manifests“ [a] weit verbreitet. Bei einer anonymen Online-Umfrage [b] 2011 gab gut ein Drittel der Befragten an, mit einem agilen Vorgehensmodell zu arbeiten. Die Teilnehmer der Studie waren vorrangig Tester und Testmanager. Dass unter den agilen Vorgehen Scrum [c] mit fast 60 Prozent dominierte, war keine Überraschung. Inzwischen diskutiert die IT-Community weiterführende Themen wie „agile enterprise“ oder den erfolgreichen Einsatz agiler Vorgehensweisen in einem regulativen Umfeld.

Als Mitarbeiter eines auf Test und Qualitätssicherung spezialisierten Beratungshauses sehen die Autoren die Entwicklung aus einem eigenen Blickwinkel: Zunehmend mehr Projekte und Unternehmen steigen auf agile Prozesse um und suchen nach Unterstützung bei der Transition, sobald diese ins Stocken gerät oder zu scheitern droht. Dabei fällt auf, dass es aus Testsicht eine Reihe „Anti-Patterns“ gibt:

- Die Akzeptanzkriterien an User Stories sind gar keine, sondern ihrerseits bloße Requirements, deren Überprüfung zunächst völlig unklar ist.
- Nicht jede User Story trägt Akzeptanzkriterien.
- Die „Tester“ im agilen Team sind räumlich getrennt vom Rest des Teams. (Gemäß Scrum gibt es die klassische Rollentrennung zwischen Tester und Entwickler idealerweise nicht. Die Verantwortung für den Test tragen das gesamte Team und jedes einzelne Teammitglied. Da das aber in der Praxis leider noch nicht umfassend anzutreffen ist und genau das ja wiederum der Auslöser für das Entstehen dieses Artikels ist, wird hier bewusst von Testern gesprochen.)

- Die Tester sind nicht an den Scrum-Meetings beteiligt. Beispiel: Im Sprint Planning werden pro User Story keine Testaufwände geschätzt, die über Unit-Tests hinausgehen.
 - Die „Definition of Ready“ erhebt keine ausreichenden Forderungen zur Testbarkeit, und als Konsequenz entstehen nicht oder schwer testbare User Stories und Software, die nicht für Testbarkeit entworfen wurde. Beispiel: Die testbare User Story liegt erst am Sprint-Ende vor, das heißt, der Sprint wurde zur „Reifung“ und Konkretisierung der Anforderung verwendet.
 - Die „Definition of Done“ äußert sich nicht ausreichend zu den Test- und Qualitätszielen, die es für jede umgesetzte User Story zu erreichen gilt.
 - Die gewählte Testautomatisierung erzeugt von Sprint zu Sprint eine zunehmende „technische Schuld“.
 - Bewährte und zielführende Testmethoden kommen nicht oder nur ansatzweise zum Einsatz – oftmals, da sie den Protagonisten schlichtweg nicht bekannt sind.
 - Das Team hat sich keine definierte Teststrategie gegeben.
- Ein wichtiger gemeinsamer Faktor vieler Anti-Patterns ist eine nicht ausreichende Integration und Kommunikation der einzelnen Disziplinen wie Anforderungsmanagement, Entwicklung und Test miteinander. Das ist wenig verwunderlich, denn vor der agilen Transition arbeiteten die meisten Projektmitarbeiter vermutlich in „Silos“ entsprechend ihrer technischen Expertise, in abgegrenzten Bereichen ohne viel Außenbeziehung. Nun sind sie plötzlich (und ggf. ohne Vorbereitung) aufgerufen, als integriertes „cross-functional team“ eng zusammenzuarbeiten. Am Willen der Beteiligten hierzu mangelt es dabei selten. Oft

Zahlreiche Prozesse bilden das Prozessreferenz-Modell. Nur einige davon sind für agile Entwicklungsprojekte essenziell wichtig (Abb. 2).

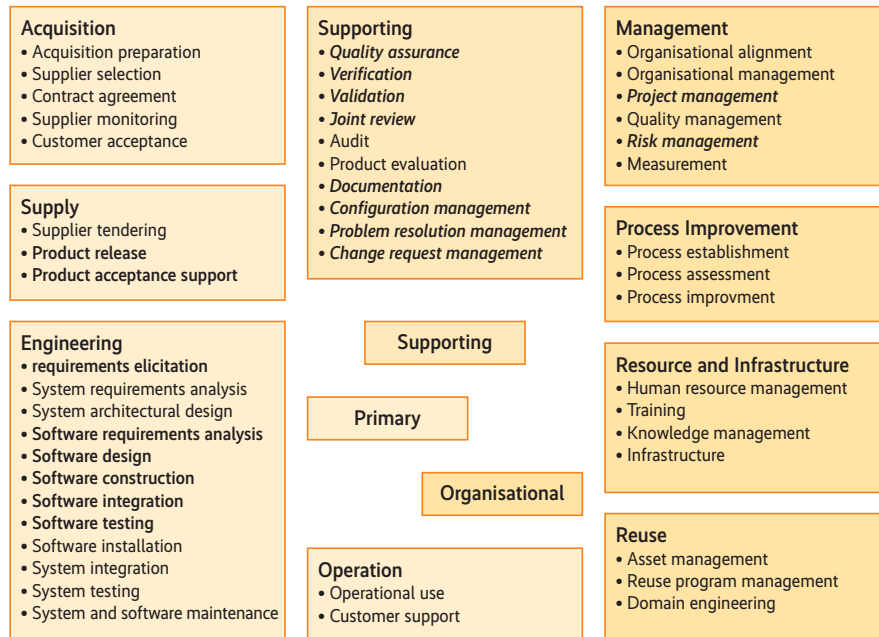
fehlen aber zur Beseitigung des Missstands konkrete Verbesserungsvorschläge und ein Leitfaden, welche Maßnahme in welcher Projektsituation am besten greift, um wirklich ein integriertes Team zu formen.

Zur systematischen Analyse von Grad und Erfolg der Integration der Disziplinen im gemeinsamen agilen Projekt wäre es hilfreich, wenn es ein einfaches Modell gäbe, das den „Idealzustand“ der Integration beschreibt. Mit ihm könnte man das konkrete Projekt vergleichen, die Abweichungen feststellen und so konkrete Maßnahmen ableiten. Für klassische Prozesse gibt es solche Modelle seit Langem in Form sogenannter Prozessreifegrad-Modelle. Es stellt sich nun die Frage, ob diese für die hier beschriebene Aufgabenstellung anwendbar sind.

Es gibt noch einen zweiten Grund, warum die Anwendbarkeit eines Reifegradmodells auf ein agiles Projekt von Interesse ist: In manchen Branchen, beispielsweise in der Automobilindustrie, werden sie häufig eingesetzt, um die Qualität eines Zulieferers beurteilen zu können. Die Einstufung kann ausschlaggebend für die Vergabe von Aufträgen sein. Viele Zulieferer adaptieren jedoch inzwischen agile Vorgehensweisen, sodass für den wirtschaftlichen Erfolg entscheidend ist, ob eine Reifegradbewertung auch nach einem Umstieg auf zum Beispiel Scrum oder Kanban noch zufriedenstellend ausfallen kann.

Exkurs: Reifegradmodelle für die Softwareentwicklung

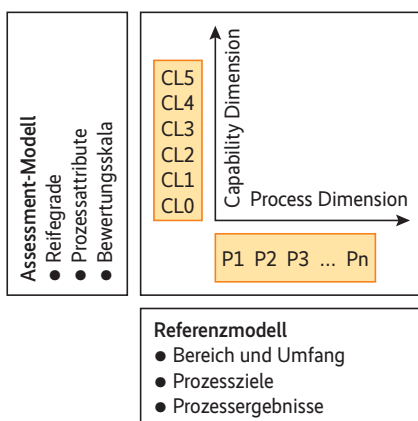
Es gibt einige Modelle zur Analyse und Verbesserung von Softwareentwicklungsprozessen. Die Grundidee ist bei allen die gleiche: Für den zu untersuchenden Prozess wird ein Referenzmodell definiert, das Zweck, grundlegende Eigenschaften, Eingangs- und Ausgangsinformationen des Prozesses abstrakt de-



finiert und für eine Prozessanalyse Richtlinien und Indikatoren festlegt, mit denen sich der Grad der Konformität des untersuchten Prozesses zu diesem Modell „messen“ lässt. Die bekanntesten und derzeit gebräuchlichsten Modelle sind zweifellos erstens CMMI (Capability Maturity Model Integration) vom amerikanischen SEI beziehungsweise das auf Basis von CMMI definierte SCAMPI (Standard CMMI Appraisal Method for Process Improvement) [1] und zweitens die internationale Norm ISO/IEC 15504, häufig einfach als SPICE (Software Process Improvement and Capability Determination) bezeichnet [2]. Die Unterschiede zwischen diesen beiden Modellen sind im Kontext des Artikels nicht relevant; als Beispielmodell wird daher im Folgenden ausschließlich SPICE herangezogen.

SPICE ordnet einen gegebenen Prozess anhand zahlreicher Kriterien auf einem von sechs Reifegraden ein. Mit jedem erreichten Reifegrad realisiert der Prozess bestimmte Qualitätsmerkmale, die einen positiven Einfluss auf die Qualität des erzeugten Produkts (also der Software) haben und so der Organisation zugute kommen. SPICE unterstützt grundsätzlich die Analyse beliebiger Prozesse. Hierzu unterscheidet der Standard zwischen dem sogenannten Process Reference Model (PRM) und dem Process Assessment Model (PAM). Ersteres beschreibt Zweck sowie Eingangs- und Ausgangsinformationen der zu betrachtenden Prozesse für einen bestimmten Anwendungsfall. PAM betrachtet die Reifegrade und die zu nutzenden Attribute und Indikatoren zur Einstufung eines Prozesses (s. Abb. 1).

Für Softwareentwicklungsprojekte ist das gebräuchliche PRM in einer weiteren Norm, der ISO/IEC 12207, beschrieben. Das zugehörige PAM findet sich im Teil 5 der Norm ISO/IEC 15504 als sogenanntes „exemplar software life cycle process assessment model“. Letzteres enthält die Prozessbeschreibungen aus dem PRM nochmals, sodass es für eine Übersicht über das Modell als einzelne Quelle ausreicht. Die beschriebenen Prozesse, unterteilt in drei Bereiche (Primär-, Unterstützungs- und Managementprozesse) und neun Kategorien, decken alle denkbaren Prozesse in einer Software produzierenden Organisation ab. Für eine konkrete Reifegradbestimmung wird man selten alle betrachten, sondern orientiert am Zweck des Assessments eine sinnvolle Auswahl treffen. In der Abbildung 2 sind zwei Teilmengen als Beispiele markiert:



- Die fett markierte Auswahl dient dazu, die Entwicklung eines einzelnen Softwareprodukts ohne Hardwareanteile zu bewerten. Dabei wird bei der Erhebung der Anforderungen begonnen und bei der Lieferung der Software geendet.
- Die kursive Auswahl ließe sich zusätzlich dazu verwenden, bestimmte Projektaktivitäten genauer zu untersuchen – insbesondere dann, wenn bei einer vorangegangenen Analyse mit der ersten Auswahl Problembereiche festgestellt worden sind. Die ausgewählten Support- und Managementprozesse betrachten viele Aspekte, die innerhalb der Primärprozesse dargestellt sind, aus einem anderen Blickwinkel.

Mit einer SPICE-Prozessanalyse lässt sich detailliert bestimmen, wo Stärken, Schwächen, Risiken und Verbesserungspotenziale eines Projekts liegen. Aber funktioniert das auch bei agilen Projekten?

Wie bewerten Reifegradmodelle agile Projekte?

Die Benennung und die Reihenfolge der Engineering-Prozesse scheint einen phasenorientierten „V-Modell-artigen“ Prozess einzufordern. Der Gedanke, sich an einen solchen vorgegebenen Prozess halten zu müssen, dürfte den meisten agilen Teams angesichts der Forderungen des agilen Manifests nach Selbstorganisation und ständiger Prozessanpassung ziemlich widersinnig erscheinen. Das ist jedoch größtenteils einem verbreiteten Missverständnis anzulasten: Es geht bei der Anwendung von SPICE nicht darum, einen vorgegebenen Prozess stur zu implementieren, sondern lediglich darum, den eigenen Prozess dahingehend weiterzuentwickeln, dass er die durch SPICE definierten Qualitätsmerkmale erreicht. In der Beurteilung der Qualitätsmerkmale muss man bei einer SPICE-Reifegradbestimmung auf den Kontext des agilen Vorgehens Rücksicht nehmen, was auf den folgenden Reifegraden in unterschiedlicher Weise einfließen muss.

Der SPICE-Level (= Reifegrad) 1 („durchgeführt“) wird unabhängig vom Vorgehensmodell erreicht, wenn die angewendeten Techniken und Prozesse dazu geeignet sind, den Prozesszweck zu erreichen. Das agile Manifest stellt klar die Erzeugung funktionierender Software in den Mittelpunkt des Projekts. Das heißt, ein gelebter agiler Prozess kann die Anforderungen von Level 1 erfüllen. Allerdings gilt es zu beachten, dass agile Frameworks wie Scrum oder Kanban nur das Projektmanagement definieren und über die inhaltliche Arbeit (etwa wie eine Systemarchitektur erstellt wird, wie Reviews durchzuführen sind und wie man zu Testfällen kommt) wenig aussagen. Erprobte Praktiken wie Test-Driven Development, Pair Programming und Agile Modeling lassen sich einsetzen, um die in [3] geforderten „Base Practices“ umzusetzen und die erwarteten „Work Products“ zu erzeugen. Die entsprechenden Listen in der Norm sind bei der Betrachtung eines agilen Projekts allerdings eindeutig als Vorschlag, nicht als verbindliche Checkliste zu betrachten. Sonst besteht tatsächlich die Gefahr, dass Dokumente eingefordert werden, die keinen Wert aus Sicht des Teams haben.

SPICE-Level 2 („gemanagt“) fordert zum einen eine überwachte und fortgeschriebene Planung. Scrum erfüllt diese Forderungen beispielsweise durch die Planungsmechanismen der Backlogs (Release, Product und Sprint) und die regelmäßigen Standup-Meetings, in denen das Team den Fortschritt diskutiert und die Planung bei Bedarf korrigiert. Grundlage für die Bewertung der Planungen muss sein, dass beim agilen Vorgehen bewusst nicht alle drei Projektgrößen (Budget, Termin und Inhalt) vorab festgelegt sind. Zum anderen schreibt SPICE vor, dass Ar-

beitsprodukte qualitätsgesichert und gelenkt werden müssen. Ein Scrum-Team kann den Anspruch durch eine Team-Chartra realisieren, die die Vereinbarungen im Team hinsichtlich Verwendung eines Knowledge-Management-System, Nutzung von Vorlagen, „Done“-Kriterien und Review- beziehungsweise Pairing-Gewohnheiten festhält – unter der Voraussetzung, dass ein reifes Team diese auch ernst nimmt und anwendet. Der wesentliche „Hebel“ hierfür ist die Selbstorganisation: Regeln, die das Team autonom für sich aufstellt, wird es auch umsetzen wollen. Nicht jede Whiteboard-Skizze und jeder Notizzettel sind zu archivieren und zu versionieren. Es gilt hier, kritisch zu bewerten, ob die Lenkung eines bestimmten Arbeitsprodukts zum Projekterfolg beiträgt oder nicht.

Für SPICE-Level 3 („etabliert“) wird ein standardisierter Prozess gefordert, der zur Erreichung der Prozessziele geeignet ist. Das scheint dem agilen Vorgehen mit dem Grundsatz der ständigen Anpassung entgegenzustehen. Tatsächlich lässt SPICE aber ähnlich der ISO 9000 offen, welchen Geltungsbereich der Standardprozess hat, wie detailliert er definiert sein muss und wie umfangreich das sogenannte Tailoring, die Anpassung des Standardprozesses an ein konkretes Projekt, ausfallen darf. In agilen Projekten findet durchaus Wiederverwendung bewährter Vorgehensweisen statt. Agile Frameworks wie Scrum oder Kanban lassen sich sehr wohl als Teil eines Standardprozesses betrachten. Des Weiteren praktizieren agile Teams „practice sharing“, den Austausch praxisbewährter Methoden und Techniken. Die zentralen Anforderungen des SPICE-Level 3 kann ein Team wie folgt umsetzen:

- Die Definition und Anwendung eines Standardprozesses kann in Form einer Team-Chartra erfolgen, in der das Team zum Beispiel basierend auf dem Scrum Guide [c] seine internen Spielregeln zu Themen wie Coding Guidelines, Pairing, Nutzung des KM-Systems und Ableitung von Testfällen festhält.
- Beschreibung der Zusammenarbeit des Prozesses mit anderen: Das agile Team sollte in der Team-Chartra ebenso beschreiben, wie die Kooperation mit dem Product Owner, den Anwendern, weiteren agilen Teams – beispielsweise durch Scrum of Scrums – und Organisationseinheiten wie IT und Produktsupport aussieht.
- Beschreibung von Rollen und Qualifikationsmaßnahmen für die Mitarbeiter je Rolle: Die meisten agilen Teams werden hier nach Einfachheit streben und sich zum Beispiel auf das Rollenmodell von Scrum – Product Owner, Team, Scrum Master – beschränken. Da für diese Rollen geeignete Ausbildungsmaßnahmen wie Trainings verfügbar sind, ist die Definition der Qualifikationsmaßnahmen recht einfach.
- Die Identifikation und zuverlässige Bereitstellung der Bestandteile der Arbeitsumgebung werden das natürliche Bestreben des Scrum Master sein, der ja dafür sorgen soll, dass das Team ungestört arbeiten kann.
- Durch die Festlegung von Methoden zur Bewertung der Effektivität und Angemessenheit des Prozesses kann ein Scrum-Team auf vielfältige Feedback-Mechanismen auf mehreren Ebenen verweisen – von der direkten und offenen Kommunikation, die das agile Manifest einfordert, über Pairings und Peer-Reviews bis hin zu Sprint-Review und Retrospektive. Als Langzeitgedächtnis für die Erkenntnisse kann das Team beispielsweise ein Wiki einsetzen (in dem dann auch gleich die Team-Chartra abgelegt wird) und so dafür sorgen, dass sich neue Teammitglieder schnell auf den Kenntnisstand der „alten Hasen“ bringen lassen.

SPICE-Level 4 („vorhersagbar“) stellt die Erhebung quantitativer Kennzahlen in den Mittelpunkt, mit denen der Prozess innerhalb definierter Grenzen gesteuert und die Ergebnisqualität

vorhersagbar wird. Gemäß den zwölf agilen Prinzipien ist lauffähige Software das primäre Fortschrittsmaß. Viele Teams messen daher den Umfang der erfolgreich implementierten (und getesteten) Kundenanforderungen und nutzen das Wissen darüber in Form der „Team Velocity“ als Grundlage für die Planung der Weiterentwicklung der Software. Ob die Geschwindigkeit in Arbeitsstunden pro Sprint oder in abstrakten „Story Points“ ausgedrückt wird, ist aus Sicht von SPICE zwar unerheblich. Etliche Diskussionsbeiträge im Internet [d, e] zeigen aber, dass die Velocity und die Sprint-Schätzungen leider oftmals missbraucht werden, um Managementenerwartungen gegenüber dem Team durchzusetzen, woraufhin Teams oftmals als Akt der Selbstverteidigung dazu übergehen, Schätzungen und Messungen zu manipulieren.

Das deutet darauf hin, dass sich Story-Schätzungen und Velocity-Messungen eventuell nicht als verlässliche quantitative Kennzahlen im Sinn von SPICE nutzen lassen, da sie erfahrungsbasiert und personenabhängig sind. Da diese Größen aber auch als Planungs- und Kontrollgrößen im Sinne von SPICE-Level 2 verwendet werden, führt das unter Umständen auch zur Gefährdung der zweiten Prozesseinstufung. Bei einer Prozessanalyse sollte detailliert untersucht werden, ob Schätzungen, Burn-down-Charts und Velocity-Festlegungen gut funktionieren und sinnvoll genutzt oder von irgendeiner Seite für die eigenen Zwecke instrumentiert werden. Die Verwendung abstrakter Story Points statt konkreter Personenstunden hilft hierbei, ist aber oftmals nicht ausreichend.

SPICE-Level 5 („optimierend“) fordert, den Prozess auf das Erreichen von Geschäftszielen zu optimieren. Hierzu werden seine Leistung auf Basis der Kennzahlen fortlaufend verfolgt und der Prozess durch gezielte Innovation angepasst. Dieser kontinuierliche Verbesserungsprozess ist fester Bestandteil agiler Prinzipien. Arbeitet ein agiles Team mit den dargestellten Planungs- und Fortschrittsverfolgungstechniken, wird es im Rahmen der bereits bei Level 3 erwähnten Feedback-Mechanismen permanent daran arbeiten, seine Velocity zu erhöhen und die Genauigkeit seiner Schätzungen zu verbessern.

Durch die eingangs getroffene Prozessauswahl kann ein agiles Projekt also durchaus einen hohen Reifegrad erreichen, wenn es das Modell mit Augenmaß und Erfahrung anwendet. Auch wenn man auf eine bestimmte SPICE-Einstufung angewiesen ist, muss man von sich von einer Umstellung auf agiles Arbeiten also nicht abhalten lassen. Es stellt sich nun die Frage, ob unabhängig von einem „Muss“ ein Reifegradmodell auch als „Kann“ für ein agiles Projekt oder Team sinnvoll und hilfreich ist.

Bringen Reifegradmodelle agilen Projekten einen Nutzen?

Die Betrachtung zeigt, dass beim Entwurf der agilen Frameworks aus der Sicht von SPICE offenbar nichts Wichtiges vergessen wurde. Was sollte also der Einsatz eines Reifegradmodells einem agilen Team bringen?

1. Systematik für die Reihenfolge von Verbesserungen: Insbesondere unerfahrene Teams benötigen bei Retrospektiven Anleitung bei der Identifikation von Problemen, deren Ursachen und der Festlegung von Verbesserungsmaßnahmen. Ein erfahrener Scrum Master stellt hier einen wichtigen Erfolgsfaktor dar. Parallel dazu kann der Einsatz eines Reifegradmodells genaue Rückschlüsse von identifizierten Problemen auf deren mögliche Ursachen beisteuern. Da die Reifegrade aufeinander aufbauen, definiert das Modell auch eine naheliegende Reihenfolge für die Maßnahmen. Damit wirkt es als Leitplanke

für viele folgende Retrospektiven und hilft dabei, aus einzelnen Verbesserungsmaßnahmen einen kontinuierlichen, zielgerichteten Verbesserungsprozess zu machen.

2. Absicherung für Verbesserungsmaßnahmen: Reifegradmodelle definieren im Allgemeinen nicht nur Ziele, sondern auch Vorschläge für Inhalte wie die sogenannten „Base Practices“ und Checklisten für Inhalte von „Work Products“. Benötigt ein Team beispielsweise eine Vorlage für einen bestimmten Dokumententyp oder einen Ausgangspunkt für eine „Definition of Done“, kann es durch Abgleich mit dem Reifegradmodell sicherstellen, nichts Wichtiges zu vergessen.
3. Motivation für Verbesserungsmaßnahmen: Das Reifegradmodell nennt für jede Forderung auch Begründungen und Zusammenhänge. Bei der Diskussion über Prozessänderungen kann das agile Team dies einbeziehen und damit mehr Verständnis, Konsens und „Commitment“ für Änderungen erreichen – wichtige Faktoren für den Erfolg von Änderungen.

Reifegradmodelle leichtgewichtig und agil

CMMI und SPICE sind darauf ausgelegt, für unterschiedliche Vorgehensmodelle eingesetzt zu werden. Als Folge davon sind sie abstrakt und schwergewichtig. Wie in Abbildung 2 ersichtlich ist, benötigt ein agiles Projektteam nur einen Bruchteil des SPICE-Modells, um einen Verbesserungsprozess im eigenen Projekt zu realisieren. Und wie gezeigt ist für diesen dann noch eine umfangreiche „Übersetzungsarbeit“ zu leisten, damit das Modell wirklich anwendbar wird. Die Anpassung sieht SPICE durchaus vor – das PRM lässt sich beliebig reduzieren oder sogar neu definieren, und darauf aufbauend kann das PAM auf Basis des allgemeinen „Measurement Framework“ von SPICE angepasst werden. Die Automobilbranche hat das in Form von Automotive SPICE schon vorexerziert. Dennoch sind die Autoren des Artikels einen anderen Weg gegangen. Der wesentliche Grund hierfür ist, dass die etablierten Modelle nicht nur zu „schwergewichtig“ sind, sondern auch für agile Projekte nicht die richtigen Schwerpunkte setzen und an vielen Stellen zu unkonkret erschienen.

CMMI und SPICE teilen den gesamten Entwicklungsprozess in miteinander interagierende Teilprozesse auf. Diese lassen sich getrennt voneinander betrachten und verbessern. In einem konkreten Projekt könnte es also beispielsweise einen „reifen“ Test- und einen „unreifen“ Anforderungsprozess geben – was nicht wirklich realitätsnah ist, da die Effektivität und Effizienz des Testprozesses stark von der Qualität der zugrunde liegenden Anforderungen abhängt. Daher sind in einer Analyse auch die Zusammenhänge zwischen den Disziplinen zu beleuchten. Und genau hier – bei den für agile Projekte so wichtigen Faktoren der integrierten Zusammenarbeit, der intensiven Kommunikation und der permanenten Interaktion – haben die Modelle wenig zu bieten. SPICE beispielsweise reduziert diese Faktoren auf drei Elemente:

- Sicherstellung der Konsistenz zwischen Arbeitsergebnissen von aufeinanderfolgenden Prozessen (i. A. durch Traceability erreichbar)
- Verteilung von Arbeitsergebnissen an betroffene Stakeholder
- Review und Freigabe von Arbeitsergebnissen durch alle Beteiligten

Als Spezialisten für Qualitätssicherung hatten die Autoren besonders den Testprozess bei Scrum im Auge. Im Sprint kommt es besonders auf frühzeitig einsetzende und kontinuierlich betriebene Zusammenarbeit zwischen dem Product Owner, den Entwicklern und den Testern, schnelles Feedback über Tester-

– aber das Thema ist der geforderten Selbstverantwortung agiler Teams unterzuordnen.

Ergebnisse aus der Praxis

Bei der ersten Anwendung des Modells und des zugehörigen Fragenkatalogs in einem Projekt zeigte sich, dass der Ansatz tragfähig ist und wie gewünscht schnell und aufwandsarm zu nutzbringenden Ergebnissen führt. Das gesamte Assessment – inklusive Kick-off, Intervieworganisation, Durchführung und Ergebnispräsentation – ließ sich innerhalb einer Woche gemeinsam mit dem Scrum-Team abwickeln. Auffällig war, dass schon bei der Durchführung der Interviews (die gewöhnlich eine bis zwei Stunden dauerten) bei den Interviewten sofort handfeste Erkenntnisse und Ideen zur Integrations- und damit zur Prozessverbesserung entstanden. Dadurch bot die Ergebnispräsentation am letzten Tag keine großen Überraschungen mehr, sondern diente vor allem dazu, das Gesamtbild zu präsentieren und Maßnahmenpakete vorzustellen, mit denen sich die nicht erreichten Check Points erfüllen ließen (z. B.: Wie sieht eine minimale Menge an Maßnahmen aus, mit der maximal viele Check Points erreicht werden können? Mit welchen Maßnahmen wären alle Check Points erfüllt?).

Das Feedback der Teammitglieder und des Auftraggebers (in diesem Fall ein „Management-Sponsor“) fiel enorm positiv aus. Themen für die nächste Retrospektive ergaben sich unmittelbar aus den erstellten Bewertungsdokumenten, und darunter waren sowohl „pain points“, die in der Projektvergangenheit „immer mal wieder“ aufgetaucht waren, als auch neue Themen, die das gesamte Team so noch nicht betrachtet hatte. Ebenfalls geliefert wurde auf Wunsch des Kunden eine Untersuchung, mit welchen der vorgeschlagenen Verbesserungsmaßnahmen welcher Beitrag zum Erreichen des gewünschten SPICE-Levels geleistet werden kann – und welcher nicht. Denn im Fokus der Betrachtung standen die Test- und QS-Maßnahmen im agilen Entwicklungsprozess. Das heißt, die Untersuchung konnte und wollte ja kein SPICE-Assessment ersetzen.

Fazit

Mit dem Modell ist ein Schema entstanden, mit dem sich schnell und zuverlässig Verbesserungsmaßnahmen für das agile Testen (genauer: für die Integration des Testens in Prozess und Team) ermitteln lassen. Das ist an sich ein Mehrwert, führt aber zu kritischen Rückfragen. Die wichtigste lautet: Ist es sinnvoll, ein solches Schema mit Fokus „Testen“ anzuwenden und damit den Test vom restlichen Prozess „isoliert“ zu betrachten? Die Praxis zeigt, dass eine solche Betrachtung nötig und hilfreich sein kann, und sie verfolgt ja gerade das Ziel, die Isolation zu beseitigen und die Testintegration zu verbessern.

Mit der positiven Erfahrung im Rücken stellt sich die Frage, welche anderen Silos sich mit einer ähnlichen Vorgehensweise auflösen lassen, zum Beispiel zwischen

- Produktmanagement/Product Owner und Entwicklungsteam
- Entwicklungsteam und Betriebsorganisation
- Entwicklungsteam und QM-Stab

Bei Betrachtung der fünf Dimensionen des T5-Modells erkennt man, dass die Dimensionen „Team“, „Transparenz“ und „Technik“ die schrittweise Integration von Silos im agilen Umfeld angehen, und zwar weitgehend unabhängig davon, um welche konkreten Silos es sich handelt. Die beiden übrigen Dimensionen „Testmethodik“ und „Testvorgehen“ beschäftigen sich mit

Onlinequellen

- [a] Agiles Manifest
www.agilemanifesto.org
- [b] Andreas Spillner, Karin Vosseberg, Mario Winter, Peter Haberl; Wie agil ist die Praxis? Auswertung der Umfrage „Softwaretest in der Praxis“
<http://heise.de/-1369290.html>
- [c] Scrum-Framework
www.scrum.org
- [d] Ron Jeffries; Estimation is Evil; Overcoming the Estimation Obsession
pragprog.com/magazines/2013-02/estimation-is-evil
- [e] Esther Derby; Estimating is often helpful. Estimates are often not
www.estherderby.com/2012/03/estimating-is-often-helpful-estimates-are-often-not.html
- [f] James Shore; The Art of Agile Development: Done Done
www.jamesshore.com/Agile-Book/done_done.html

der dazu synchronisierten Verbesserung der technisch-methodischen Exzellenz. Für die weiteren Aufgabenstellungen liegt es nahe, die ersten drei Dimensionen weitgehend wiederzuverwenden und weitere silospezifische hinzuzunehmen. Das muss nicht zum großen vereinheitlichten Reifegradmodell für alle Silos führen, denn der Ausgangspunkt für die Arbeiten war eben, sich von einem solchen eher schwergewichtigen Modell zu lösen. Vielmehr schlagen die Autoren vor, je nach konkreter Aufgabenstellung ein spezifisches Modell nach den „Bauplänen“ von T5 zu konstruieren.

Auch in agilen Projekten lässt sich also durch den Einsatz eines passenden Reifegradmodells ein Nutzen realisieren. Allerdings liegt dieser weniger in der Prozess- als vielmehr in der Teamreifung. (ane)

Literatur

- [1] Standard CMMI Appraisal Method for Process Improvement (SCAMPISM) A, Version 1.3: Method Definition Document; Carnegie Mellon University 2011
- [2] Information Technology – Process Assessment – Part 2: Performing an Assessment; ISO/IEC, 2003
- [3] Information Technology – Process Assessment – Part 5: An exemplar software life cycle process assessment model; ISO/IEC, 2012.



Christian Brandes

arbeitet als Principal Consultant und Trainer für die imbus AG. Der promovierte Mathematiker, ISTQB Certified Tester (Full Advanced Level) und Professional Scrum Master ist seit vielen Jahren als Testmanager, -architekt, -designer und -automatisierer in IT-Projekten tätig.



Thomas Roßner

ist Technologievorstand der imbus AG. Der SPICE-Assessor und Professional Scrum Master (beides aus Überzeugung) ist (Mit-)Autor mehrerer Fachbücher zum Thema Softwaretest.

