Worauf es ankommt!

Testmanagement-werkzeuge

Wolfgang Gaida

Wie kann man sicherstellen, dass in einem IT-Projekt Anforderungen mit Testfällen belegt und nach erfolgtem Test erfüllt sind? Wie ermittelt man den Reifegrad von Software auf möglichst effiziente Weise, um damit den Status darstellen zu können? Die beste Antwort auf diese Fragen lautet: durch Werkzeugeinsatz! Welches Werkzeug für das eigene IT-Vorhaben geeignet ist, hängt von unterschiedlichen Faktoren ab. Professionelle Testmanagementwerkzeuge machen sich jedenfalls bezahlt.

Tabellenkalkulation ist hier nicht das passende Werkzeug

Nägel schlägt man nicht mit der Zange ein, dazu hat man einen Hammer. Wie bei vielem kommt es auf die Auswahl des richtigen Werkzeuges an. Und das gilt nicht nur für Handwerker, sondern auch für Test- und Projektmanager.

Es gibt immer noch Organisationen, die versuchen, mit Tabellenkalkulationen wie Excel & Co ein vernünftiges Testmanagement aufzusetzen. So sinnvoll diese einfache Lösung häufig sein mag, wundert man sich nicht selten später im Projektverlauf, dass plötzlich Entscheidungsgrundlagen für eine erfolgreiche Projektabwicklung nicht leicht ableitbar sind. Seien dies zum Beispiel korrigiert geglaubte Fehler, die in späteren Korrekturversionen wieder auftauchen, oder die Häufung von Fehlern in einem speziellen Bereich. Das führt oft zu Überlast bei Mitarbeitern und dadurch letztendlich zu schlechter Softwarequalität.

Wenn man an Testmanagementwerkzeuge denkt, muss es nicht immer ein teures kommerzielles Produkt sein, auch kostengünstige Lösungen oder Open-Source-Werkzeuge stehen manch „großem“ Werkzeug nicht viel nach. Speziell die „wichtigen“ Features bieten günstige Werkzeuge in guter Qualität an.

Als erfahrener Testmanager gehe ich hier auf Testmanagementwerkzeuge im Allgemeinen und auf einige Vertreter in unterschiedlichen Preisklassen näher ein und möchte das Thema mit einigen „Anti-Patterns“ des Testmanagements aus der Praxis untermalen.

Worauf kommt es bei einem Testmanagement-werkzeug an?

Gemäß dem International Software Testing Qualification Board (ISTQB®) legt man in der *Testplanungsphase* die Werkzeuge für den Test fest. Diese sind neben den Daten-Generatoren, Simulatoren, Last-Generatoren, Automatisierungswerkzeugen, Analysewerkzeuge für zum Beispiel Netzwerk oder Datenbanken auch das Testmanagementwerkzeug. Microsoft Excel – auch wenn ich es sehr schätze und intensiv nutze – gehört hier nicht dazu, denn wir wollen:

- das *Testprojekt planen*, indem wir gemeinsam mit der Entwicklung Korrekturzyklen, Übergaben und Versionen festlegen und Key-Performance-Indikatoren für die Erreichung von Meilensteinen definieren.
- *Anforderungen erfassen* beziehungsweise importieren und verwalten.
- *Testfälle* und deren Schritte gemäß ISTQB®-Standard *erfassen* und verwalten. Neben dem Ziel des Testfalls und den Eingangs- und Ausgangsbedingungen werden für den Soll-ist-Vergleich Testdaten und das erwartete Ergebnis gespeichert. Optimalerweise lassen sich Testfälle aus den Anforderungen generieren, was zu einer sehr guten Testabdeckung führen kann, wenn die Anforderungen in guter Qualität formuliert sind. Viele Testmangementwerkzeuge unterstützen auch die Rückverfolgbarkeit zwischen Anforderung und Testfall (s. Abb. 1).
- *Testsuiten* aus diesem Testfallvorrat definieren.

Wir werden später sehen, dass auch die Priorisierung von Anforderungen und Testfällen sehr wichtig sein kann. Während Komponententests meistens von der Entwicklung durchgeführt werden – wie mit JUnit im Falle von Java –, dienen vor allem in den Teststufen Integration, Systemtest, Systemintegration und Abnahmetest Testmanagementwerkzeuge dazu, die erforderliche Transparenz herzustellen. So können Qualitätsaussagen über die Software getroffen und gegebenenfalls steuernde Maßnahmen eingeleitet werden.

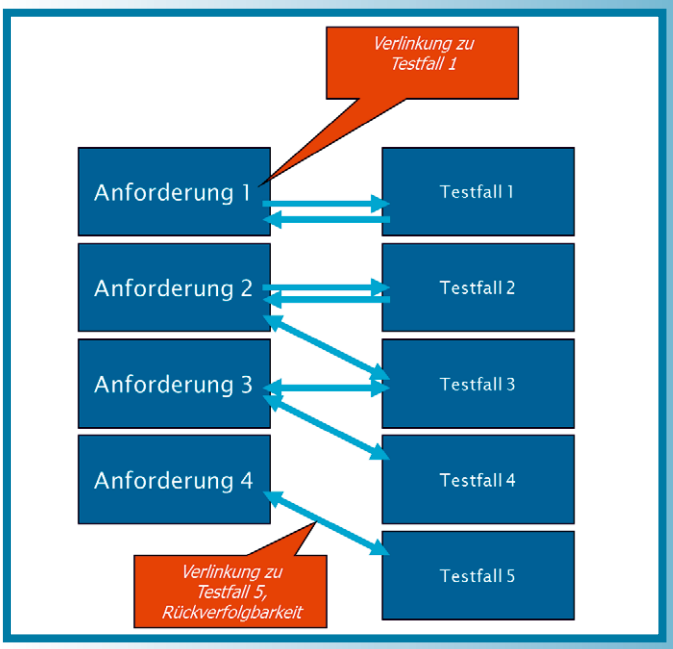


Abb. 1: Anforderungs-Verlinkung bis zu den Testfällen

All diese Informationen lassen sich auch in ein Excel-Sheet packen, aber eine übersichtliche Gestaltung ist mühsam. Besonders kritisch wird es, wenn mehrere Personen gleichzeitig daran arbeiten, ihre Anforderungen beziehungsweise Testfälle eintragen und das Ergebnis dann per E-Mail versenden. Hier ist Chaos vorprogrammiert, das man mit einem geeigneten Werkzeug vermeiden kann. Auch in der nachfolgenden Testausführungsphase stößt man allzu schnell an die Grenzen von Excel.

In der *Testausführungsphase* werden Testfälle aus den Testsuiten methodisch und strukturiert abgearbeitet. Dabei soll möglichst für jeden Testschritt beziehungsweise Testfall der Soll-ist-Vergleich durchgeführt und der Status für den Testfall ermittelt und gesetzt werden. Die meisten Testmanagementwerkzeuge unterscheiden verschiedene Status (s. Tabelle 1).

Besteht ein Testfall aus mehreren Schritten, wird der Gesamtstatus vom Testmanagementwerkzeug ermittelt. Wenn einmal etwas nicht okay ist, dann soll das Fehlverhalten in einem möglichst integrierten Fehlermanagementwerkzeug

Status	Erklärung
No Run	Der Testfall wurde noch nicht ausgeführt
OK	Alle Testschritte wurden mit dem Ergebnis OK ausgeführt
Not OK	Mindestens ein Testschritt wurde mit dem Ergebnis Not OK beendet
Blocked	Die Ausführung des Testfalles ist blockiert
Not Completed	Es wurden nicht alle Testschritte ausgeführt

Tabelle 1: Aus den Status wird der Testfortschritt ermittelt

dokumentiert werden. Ziel dabei ist es, die Fehler korrigieren zu lassen und in einer Korrekturversion nachzutesten. Wenig hilfreich – ja sogar chaotisch – wird es, wenn die beiden Regeln *„Keine Korrektur ohne Fehlermeldung“* und *„Keine Fehlermeldung ohne Testfall-Verlinkung“* missachtet werden und das Fehlverhalten per E-Mail an den Programmierer gemeldet wird. Oft wird dann eine Korrektur „über den Zaun geworfen“ und das Ganze gerät in Vergessenheit. Hier erfährt man des Öfteren ein Déjà-vu: „Diesen Fehler hatten wir schon mal vor zwei Wochen!“ Wer kann sich dann noch an die Korrekturmaßnahmen erinnern und nachvollziehen, wenn er nicht von einem guten Testmanagementwerkzeug unterstützt wird, das ihn möglicherweise gar nicht in so eine Situation hätte kommen lassen.

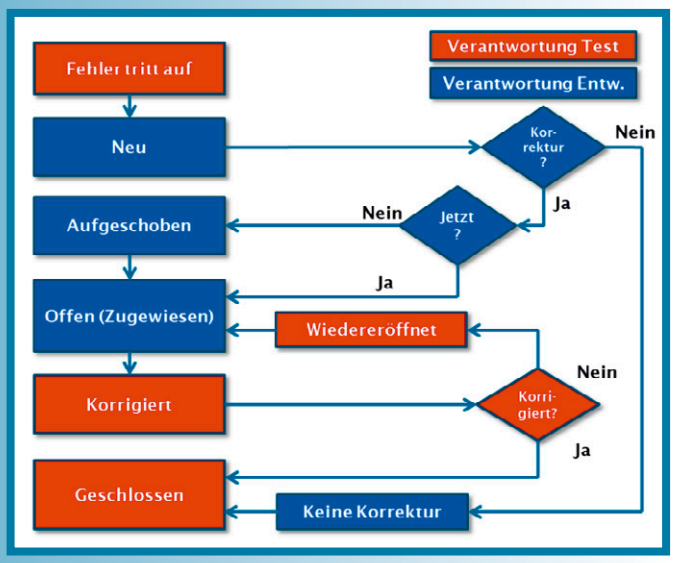


Abb. 2: Einfacher Fehlerlebenszyklus

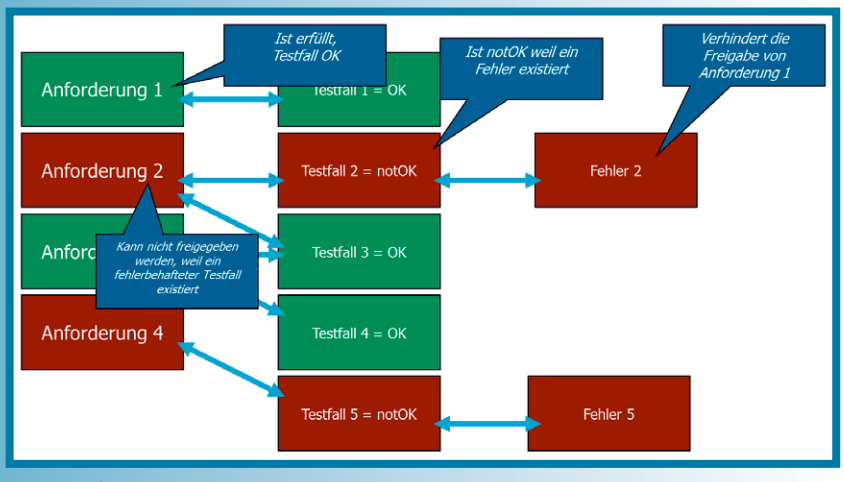


Abb. 3: Anforderungs-Verlinkung bis zu den Fehlern

Als Bestandteil des Testmanagements möchte ich kurz auf das *Fehlermanagement* eingehen, das im Testmanagmentwerkzeug oft integriert ist. Ein gutes Fehlermanagement hilft uns, neben der Nachvollziehbarkeit von Korrekturen auch noch zu erkennen, wenn es in gewissen Bereichen der Softwareentwicklung „brennt“, das heißt, wenn es zum Beispiel in der Datenbankschicht oder dem Webclient zu erhöhtem Fehleraufkommen, Korrekturverzug oder erhöhtem Aufkommen von Rejects beziehungsweise „Ping-Pongs“ kommt: Rejects sind jene Fehler, die seitens der Entwicklung als korrigiert gemeldet wurden, aber dann den Nachtest nicht bestanden haben. Als Ping-Pongs bezeichne ich gerne Rejects von Rejects. In all diesen Fällen müssten Test- und Projektmanager eingreifen, um den Projektfortschritt nicht zu gefährden. Unser Testmanagementwerkzeug liefert uns Basismaterial, um die genannten Phänomene zu erkennen, steuernd einzugreifen und Wildwuchs gar nicht erst entstehen zu lassen. Als Steuerungsmaßnahme sehe ich hier zum Beispiel Fehlerdurchsprachen, Priorisierung von Korrekturen und dergleichen.

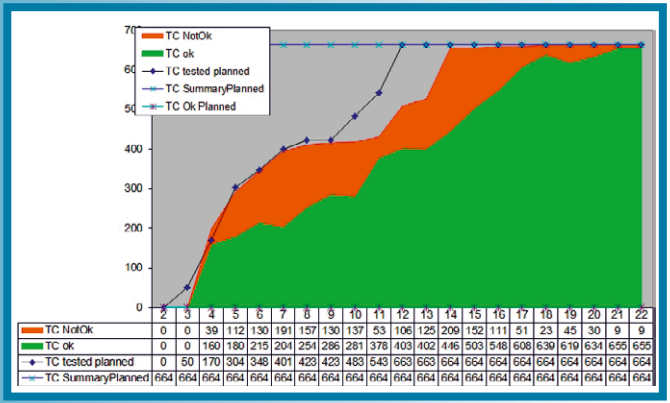
Das Testmanagementwerkzeug unterstützt uns auch, die Wichtigkeit des Fehlers beziehungsweise die Dringlichkeit der Korrektur zu dokumentieren, um den Projektverlauf effizient gestalten zu können. Des Weiteren werden die Fehler gemäß des Fehlerlebenszyklus abgearbeitet. Oft ist dieser flexibel erweiterbar und anpassbar.

Die Verlinkung und Rückverfolgbarkeit zwischen Anforderungen zu Testfällen und gegebenenfalls zu Fehlern hilft uns, Rückschlüsse auf die Qualität der Software zu ziehen, und bietet Entscheidungsgrundlagen für notwendige Eingriffe. Als Testmanager kann man zum Beispiel leicht begründen, warum eine Anforderung erfüllt oder nicht erfüllt ist, wenn man nur die Verlinkung zu den Testfällen beziehungsweise Fehlern betrachtet. Ferner lassen sich Freigaben von Anforderungen auf Basis der Verlinkungen zu Testfällen ableiten und die Test-Ende-Kriterien bewerten.

Diese Aussagen soll man natürlich in unterschiedlichen *Darstellungen* auf Knopfdruck für ein aussagekräftiges Reporting bekommen. Hierzu zählen Testfortschritt, Fehlertrend und Überdeckungsgrad in tabellarischer und grafischer Form. Wem die Out-of-the-Box-Lösungen nicht ansprechend sind, und das sind sie selten, der sei hier auf Excel verwiesen, denn zum Darstellen von Grafiken auf Basis der tabellarischen Out-of-the-Box-Berichte ist die Tabellenkalkulation wiederum sehr gut geeignet.

Managementtaugliche Kurzberichte stellen gute Testmanagementwerkzeuge auf einem frei konfigurierbaren Dashboard aktuell, übersichtlich und knackig dar. In der Praxis werden die Dashboard-Informationen auf einer Projekt-Homepage im Intranet abgebildet.

Weitere wichtige Eigenschaften eines Testmanagementwerkzeuges sind *Zusammenarbeit* mehrerer Mitarbeiter in verschiedenen Rollen an unterschiedlichen Standorten beziehungsweise die *Historisierung* der Dokumente. Das Tüpferl auf dem i ist natürlich ein Zugang über Web-Browser oder ein Zugang in der Cloud, wo man nicht einmal eigene Server-Hardware benötigt. Fein und brauchbar, aber oft holprig realisiert sind so manche Import- und Export-Funktionen von Anforderungen, Testfällen oder



	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
TC NotOk	0	0	39	112	130	191	157	130	137	53	106	125	209	152	111	51	23	45	30	9	9
TC ok	0	0	160	180	215	204	254	286	281	378	403	402	446	503	548	608	639	619	634	655	655
TC tested planned	0	50	170	304	348	401	423	423	483	543	663	663	664	664	664	664	664	664	664	664	664
TC SummaryPlanned	664	664	664	664	664	664	664	664	664	664	664	664	664	664	664	664	664	664	664	664	664

Abb. 4: Testfortschritte auf Basis der Daten im Testmanagementwerkzeug (generiert mit Excel)

Fehlern. Systeme, bei denen eine Clientinstallation erforderlich ist, möchte ich schon fast als aussterbende Exoten bezeichnen.

In der Phase *Bewertung der Endekriterien und Bericht* unterstützt uns das Testmanagementwerkzeug bei der Beurteilung der Test-Endekriterien auf Basis der zu Projektbeginn festgelegten Key-Performance-Indikatoren und hilft uns auch bei der Formulierung des Testabschlussberichts, in dem die Erfüllung der Anforderungen und gegebenenfalls auch noch akzeptable offene Fehler und deren Korrekturplanung niedergeschrieben sind.

Typische Vertreter für Testmanagementwerkzeuge

Im Folgenden werde ich ein paar typische Vertreter kurz charakterisieren.

HP-ALM: nicht billig, aber zielführend

HP-ALM, die Anwendungslebenszyklusmanagement-Software von Hewlett-Packard, bietet alle oben genannten Features und noch einige mehr, wenn man die entsprechenden Module zukauft, wie Business-Components oder Parametrisierung von Testfällen. Diese Funktionalitäten sind zwar recht praktisch, aber dennoch kompliziert in der Handhabung. HP-ALM ist sehr flexibel konfigurierbar; der Bug-Life-Cycle ist ebenfalls erweiterbar, falls der Out-of-the-Box-Ansatz nicht reicht. Kritische Stimmen bemerken oft, dass HP-ALM keine schlanke Lösung ist und in manchen Bereichen aufgeblasen und überladen wirkt und überdies kompliziert in der Bedienung ist. Der Aufbau ist aber dermaßen gestaltet, dass man Module, die man nicht zwingend benötigt, weglassen kann.

Eine Spezialität von HP-ALM sind die Business-Components. Business-Components werden zu Business-Process-Testfällen zusammengefügt und sind wiederverwendbare Einheiten, die eine spezifische Aufgabe in einem Business-Prozess erfüllen. Sie beschreiben den Zustand vor und nach der Ausführung der Aufgabe. Eine weitere wichtige Besonderheit ist die Möglichkeit, Testfälle optional mit Parametern zu versehen. Solche Testfälle werden einmal spezifiziert und laufen mit unterschiedlichen Datensätzen, um unterschiedliche Bedingungen testen zu können.

Ein wirklich gutes Feature ist die Generierung von Testfällen aus Anforderungen. Mit der Funktion „Convert to Test" können aus Anforderungen Testfälle beziehungsweise Testschritte generiert werden. Die automatische Verlinkung zu den Anforderungen, aus denen generiert wurde, ist für mich eine selbstverständliche Funktionalität. Die Benutzeroberfläche hat in der aktuellen Version 12.0 ein wirklich guttuendes Facelifting erfahren, obgleich die Vorgängerversionen schon eine ausreichende Usability aufwiesen.

TestRail: schlanke Lösung, ohne Rückverfolgbarkeit

Während einer Werkzeug-Evaluierung bin ich auf das Werkzeug TestRail gestoßen. TestRail ist eine kommerzielle, schlanke, einfache Lösung zur Verwaltung von Testfällen, die entweder installiert oder in der Cloud verwendet werden kann. Die Testfälle lassen sich in Sections gut und übersichtlich strukturieren. Aus diesem Testfall-Vorrat werden auf einfache Art und Weise Testsuiten definiert, die strukturiert durchgeführt werden. Beim Zuweisen von Verantwortlichkeiten zu Teammitgliedern werden optional Benachrichtigungen per E-Mail versendet. Out-of-the-Box werden brauchbare Berichte angeboten, die am Dashboard dargestellt oder exportiert werden können. Es gibt auch unterschiedliche Rollen, wie Designer, Tester, Lead und Guest. Leider fehlt ein integriertes Anforderungs- und Fehlermanagement, wodurch die Funktionalität der Rückverfolgbarkeit nicht geboten wird. Dem gegenüber stehen die Vorteile: Cloud beziehungsweise Installation, Preismodell, ansprechende Oberfläche und intuitive, einfache Bedienung.

TestLink: intuitiv, aber langweiliges Layout

Wer sich lieber in der Open-Source-Welt bewegen möchte, dem sei das Werkzeug TestLink empfohlen. TestLink wird (noch) nicht in der Cloud angeboten. Hier ist eine Installation in der eigenen Hardwarelandschaft erforderlich. Zum Betrieb sind ein Datenbank- und ein Webserver notwendig, die auf dem gleichen Rechner installiert sein können. Installation und Konfiguration gehen reibungslos vonstatten. Mit TestLink können Anforderungen verwaltet werden, wobei die Rückverfolgbarkeit der Testfälle gewährleistet ist.

Leider fehlt ein integriertes Fehlermanagement. Im Open-Source-Bereich findet man hier Mantis oder Bugzilla, die sehr einfach zu bedienen und stabil sind und auch professionell eingesetzt werden. Die Verlinkung zu Testfällen muss dann aber manuell durch Erfassen des Links oder der Fehlernummer geschehen, was eine gewisse Disziplin der Tester erfordert. Das Reporting liefert alle wichtigen Informationen in HTML beziehungsweise im Word- oder Excel-Format, wo man die Ergebnisse gefälliger darstellen kann. Die Bedienung von TestLink ist zwar einfach und intuitiv gestaltet, jedoch wirkt die Benutzeroberfläche in der aktuellen Version 1.9.9 mehr als veraltet, was das Gesamtbild sehr trübt.

Tarantula: Open-Source-Werkzeug mit professionell erscheinendem Layout

Auf den ersten Blick erscheint Tarantula höchst professionell und die Benutzeroberfläche mutet äußerst modern an. Teilweise erscheint aber trotzdem die Bedienung in manchen Bereichen etwas umständlich. Vielleicht muss man länger damit arbeiten und Erfahrungen sammeln, um sich daran zu gewöhnen. Bei Tarantula gibt es Anforderungsmanagement, Testfallmanagement, Reporting und Import- und Export-Funktionen. Ein integriertes Fehlermanagement sucht man vergeblich. Stattdessen bietet Tarantula eine Verlinkung zu externen Fehlermanagementwerkzeugen wie Bugzilla an. Soll eine große Anzahl von Anforderungen oder Testfällen verwaltet werden, kann es schnell unübersichtlich werden. Die Verlinkung zwischen Testfällen und Anforderungen erfolgt per Drag&Drop, wenn man einmal dahintergekommen ist, wie es funktioniert. Auch die Behandlung von Anwenderfehlern ist noch nicht ganz ausgereift. Es erscheinen des Öfteren Dialogboxen mit abgeschnittenem Text und dergleichen. Das Reporting liefert wie

bei vielen Werkzeugen nur brauchbare Rohdaten. Die Verwendung in der Cloud ist zurzeit nicht möglich, es werden keine neuen SaaS (Software as a Service)-Installationen angeboten.

TestTrack: sehr gute All-In-One-Lösung

TestTrack ist ein kommerziell erhältliches Testmanagementwerkzeug, das alle erforderlichen Bereiche, angefangen vom Anforderungsmanagement, über die Testfallverwaltung, das Fehlermanagement bis hin zum Berichtswesen, abdeckt. Ich habe die Cloud-Variante evaluiert und musste dabei feststellen, dass es sehr große Überdeckungen zu HP-ALM gibt. Lediglich Business-Components und das Parametrisieren von Testfällen gibt es hier nicht. Diese Funktionalitäten sind zwar sehr brauchbar, um effizient Testfälle und Varianten generieren zu können, werden aber oft in der Praxis wenig genutzt. Auch das Generieren von Testfällen aus den Anforderungen fehlt, was schon ein größeres Manko darstellt. Generell wirkt die Bedienung sehr modern und man findet sich schnell zurecht. Einzig die Anordnung der Elemente „Issue Tracking“, „Testing“, „Requirements“, „Folders“ und „Reports“ gefällt mir persönlich nicht so gut.

Für das Reporting steht jede Menge an vorgefertigten Vorlagen zur Verfügung. Die Grafiken, die wie auch bei vielen anderen Werkzeugen sehr einfach wirken, werden wohl eher als Rohdaten für das Projektreporting herangezogen. Leider habe ich eine Exportfunktion nicht vorgefunden und musste mit Drag&Drop die Reports verfeinern. Sehr gut hat mir die Out-of-the-Box-Benachrichtigungsfunktion gefallen, bei welcher die gerade verantwortlichen Mitarbeiter per E-Mail über deren offene Tasks informiert werden. Weniger gut finde ich die Tatsache, dass es nicht möglich ist, Testfälle mit Prioritäten zu belegen – aber vielleicht ist das auch nur in der Demoversion nicht möglich gewesen. Preislich würde ich TestTrack im Mittelfeld ansiedeln.

Zwei weitere Werkzeuge kurz betrachtet

Polarion, eine webbasierte Anwendung zur Verwaltung von Anforderungen, Testfällen und Fehlern, bietet eine gut funktionierende Schnittstelle zum Import von Testfällen aus Excel an. Neben der Importfunktion können Testfälle natürlich auch manuell erstellt werden. Testfälle werden mittels Templates einfach zu Runs zusammengefasst und diese dann während der Durchführung abgearbeitet. Auch ein sehr mächtiges Reporting steht zur Verfügung. Die Erstellung von Artefakten (Anforderungen und Fehler) in Polarion ist stark workflowbasiert, was aufgrund der Konfigurationsmöglichkeiten manchen Anwender leicht überfordern kann.

Jira ist eine webbasierte Anwendung zur Verwaltung von Fehlern, Problemen und Aufgaben. Erweiterungen für eine Spezialisierung in Richtung Testfall- und Dokumentenverwaltung lassen sich mit den Add-Ons Zephyr und Confluenze leicht realisieren. Der Mehrwert von Zephyr besteht in der Gestaltung von Berichten, wie wöchentliche Statusberichte für Fehlertrend und Testfortschritt, sowie durch die Unterstützung von Testfällen, die als spezielle Tasks realisiert sind.

Die Cloud: aktueller Trend mit vielen Vorteilen

Software as a Service (SaaS) liegt auch im Bereich Testmanagement im Trend. Viele Hersteller bieten bereits ihre Werkzeuge in der Cloud an. Das hat vor einiger Zeit noch nach Zukunftsmusik geklungen, ist aber heute fast schon ein Must-Have. Auch Open-Source-Produkte werden in Cloud-Lösungen gehostet, aber meistens nicht kostenfrei. Egal ob kommerziell oder Open Source man erspart sich gerade in der Evaluierungsphase die Installationen und Deployments im eigenen Hardwarepark und somit Zeit und Geld. Nach der Auswahl des richtigen Werkzeuges kann man Werkzeuge in der Cloud mieten und diese sofort nach der Registrierung nutzen. Dabei spart man Kosten für Server in der Anschaffung, Betrieb und Wartung. Demgegenüber steht der Nachteil, dass sensible Daten „irgendwo“ im Internet abgelegt werden und möglicherweise gehackt werden können. Auf alle Fälle sind die Lizenzmodelle genau zu studieren, da sich oft günstig erscheinende Lösungen langfristig als Kostenfallen herausstellen können. In diesem Falle würde man mit einer lokalen Installation besser fahren.

Fazit

Eine Empfehlung für das richtige und wahre Testmanagementwerkzeug kann und möchte ich hier aufgrund der Kontextabhängigkeit nicht geben. Erfahrungen, Projekt/Programmgröße, Umfeld, Budget, Schulungsbedarf und eine eventuell vorhandene beziehungsweise zu schaffende Serverlandschaft oder Infrastrukturüberlegungen müssen abgewogen werden, um sich auf ein Werkzeug einzulassen. Nur so kann die geforderte Qualität in vollem Ausmaß erreicht werden. Die Vorteile für ein professionelles Werkzeug liegen aber klar auf der Hand und rechtfertigen zusätzlichen finanziellen Aufwand und garantieren einen Return-on-Invest.

Frei nach Paul Watzlawicks „Anleitung zum Unglücklichsein“ [Wat88] – und hier im Besonderen „Die Hammergeschichte“ – möchte ich abschließend doch noch eine Empfehlung zum Erfolgreichsein geben, damit es niemandem wie dem Mann mit dem Nagel in dieser Geschichte ergeht: Dieser Mann möchte ein Bild aufhängen, hat einen Nagel, nur der Hammer fehlt ihm. Seinen Nachbarn, der einen Hammer besitzt, will er zwar fragen, grübelt aber lange darüber nach, warum ihm vielleicht der Nachbar den Hammer nicht borgen will. Und so stürzt er nach der Grübelei hinüber zum Nachbarn und schreit ihn an: „Behalten Sie Ihren Hammer!“, ohne ihn überhaupt gefragt zu haben.

Und hier meine Empfehlung: Falls Sie Fragen haben oder Beratung zur Werkzeugauswahl beziehungsweise zum Werkzeugeinsatz benötigen, kontaktieren Sie mich oder einen meiner Kollegen bei ANECON oder schauen in unseren Blog (`http://www.anecon.com/blog`)!

Literatur

[ISTQB/GTB] ISTQB® GTB-Standardglossar der Testbegriffe, online auf Deutsch und Englisch unter `http://glossar.german-testing-board.info/`

[Wat88] P. Watzlawick, Anleitung zum Unglücklichsein, Piper-Verlag, 1988

Wolfgang Gaida ist Software-Testmanager bei der 1998 in Wien gegründeten ANECON Software Design und Beratung GmbH. Sein Interesse gilt vor allem dem Consulting für Test- und Projektmanagement verteilter Systeme in den Bereichen Telekommunikation, Energy und Logistik sowie der Durchführung von Last- und Performancetests.
E-Mail: wolfgang.gaida@anecon.com