OBJEKTspektrum ONLINE Themen special

Artikel

Auch Agilität ist kein „Silver-Bullet“

Wann agile Vorgehensweisen und leichtgewichtige Dokumente an ihre Grenzen kommen

von Stan Bühne

Seit Jahren wächst der Hype um agile Vorgehensweisen, wobei ich das Gefühl habe, dass sich der Hype in den vergangenen zwei Jahren, zusammen mit den Themen Digitalisierung und Industrie 4.0 exponentiell gesteigert hat. Fast täglich erscheinen neue Artikel, White-Paper, Management-Präsentationen oder gar ganze Bücher, die Agilität als eine Art Wunderwaffe predigen, um der stetig steigenden Komplexität und dem Bedarf einer immer kürzer werdenden Time-To-Market gerecht zu werden. Aber ist es tatsächlich der Fall, dass mit agilen Vorgehensweisen nun – mehr als 30 Jahre nach Brooks [Bro86] Aussage: „there is no silver bullet in software engineering“ – endlich ein Allheilmittel gefunden wurde, um Software mit besserer Qualität in kürzerer Zeit zu liefern? Oder öffnen auch agile Vorgehensweisen nur einen weiteren Pfad, mit eigenen Herausforderungen? Im Grunde ist es ganz egal, im Wandel der Zeit sehen viele Unternehmen keine andere Möglichkeit und wollen agil werden, da sie sich von diesem Vorgehen bessere Ergebnisse und schnellere Erfolge versprechen. Dass diese Umstellung aber mehr ist, als nur das Weglassen wenig geliebter Aufgaben und deutlich weitreichendere Umstellungen mit sich bringt als nur die des Software-Entwicklungsprozesses, gerät dabei häufig in Vergessenheit. Vor allem große Unternehmen sind heute mit ihren Organisationsstrukturen an klassischen (wasserfallartigen bzw. phasenbasierten) Prozessen ausgerichtet. Darüber hinaus hat das Thema Outsourcing in den vergangenen Jahren dafür gesorgt, dass Entwicklung, Test und Betrieb ausgegliedert und von Lieferanten und Dienstleistern in Nearshore und Offshore Ländern durchgeführt werden. Diese Gegebenheiten erschweren die grundsätzliche Idee von agilen Vorgehensweisen. In diesem Artikel möchte ich einige meiner Erfahrungen teilen und aufzeigen, warum agile Vorgehensweisen in großen Unternehmen mit strikten Freigabeprozessen, organisatorischen Konzernstrukturen, zerfasertem Expertenwissen und klassischen Lieferantenbeziehungen mit Lasten- und Pflichtenheften an ihre Grenzen kommen.

Agile Softwareentwicklung

Agiles Manifest

- **Individuen und Interaktionen** stehen über Prozessen und Werkzeugen.
- **Funktionierende Software** steht über einer umfassenden Dokumentation.
- **Zusammenarbeit mit dem Kunden** steht über der Vertragsverhandlung.
- **Reagieren auf Veränderung** steht über dem Befolgen eines Plans.

Kasten 1: Agiles Manifest

Im Gegensatz zu klassischen Vorgehensweisen (vgl. Kasten 2), verfolgen agile Vorgehensweisen (z. B. Scrum, Extreme

Programming, Feature Driven Development) das Ziel, bereits in möglichst kurzer Zeit einen Mehrwert für das Unternehmen zu schaffen, der sich über das sog. „Minimal Viable Product“ [Kni16] definiert und über die Zeit kontinuierlich weiterentwickelt wird (Kasten 3).

Merkmale klassischer Vorgehensweisen

- Vorabdefinition der Anforderungen z. B. in schwergewichtigen Lasten- und Pflichtenheften.
- Strikte Qualitäts-Gateways, um von einer Entwicklungsphase in die nächste überzugehen.
- Entwicklung startet erst nachdem die gesamten Anforderungen an das Produkt final sind.
- Fest definierte Release-Zeitpunkte pro Jahr, um Änderungen in Produktion zu bringen.
- Änderungen und neue Anforderungen werden über Change-Requests eingebracht.

Kasten 2: Merkmale klassischer Vorgehensweisen

Klassische, schwergewichtige Vorgehensweisen sind in der Regel darauf bedacht, das Gesamtprodukt, das alle Probleme erschlägt, auf einen Schlag zu erstellen und laufen damit nicht selten in die Falle, dass trotz aller Sorgfalt die zukünftige Markt-, Technologie-, Gesetzeslandschaft eine andere ist als die, die man zum Start erwartet hat. In einen anderen Kontext gesetzt, ist agile Entwicklung eher ein „Fahren auf Sicht“ und kein Versuch, eine gesamte Strecke vom Start bis zum Ziel mit allen Eventualitäten durchzuplanen.

Vorteile agiler gegenüber klassischen Vorgehensweisen

- schnellerer Mehrwert für das Unternehmen, durch das sog. Minimal Viable Product und dessen kontinuierliche Weiterentwicklung [Kni16],
- geringeres Risiko, dass zum Start bereits ein veraltetes Produkt vorliegt, durch „Fahren auf Sicht“,
- schnellere Reaktion auf Problem-/Marktveränderungen, durch kontinuierliche Bewertung. der ausstehenden Anforderungen im sog. Product Backlog,
- weniger Blindleistung in Design und Entwicklung in Folge von sich ändernden Bedürfnissen durch kürzere Entwicklungszeiträume.

Kasten 3: Vorteile agiler Vorgehensweisen

Allen agilen Vorgehensweisen gemein sind die agilen Werte vgl. [Bec01] und Kasten 1. Diese Werte wird niemand – auch kein klassischer Projektmanager oder Requirements Engineer in Abrede stellen. Fragen, die sich bei der Anwendung dieser Werte, in großen Unternehmen mit bestehenden Organisationsstrukturen, gegebenen Lieferantenbeziehungen, verteilter Entwicklung und ausgelagerten Test- und Betriebsbereichen, allerdings stellen sind u. a. folgende:

> welche Prozesse und Werkzeuge sind für die Zusammenarbeit innerhalb des Unternehmens, aber auch in Zusammenarbeit mit den Lieferanten mindestens nötig, um „Chaos“ zu vermeiden?
> wie leichtgewichtig darf eine Dokumentation sein, wenn aufgrund der vorhandenen Organisationsstruktur oder Lieferantenbeziehung die Softwareentwickler und Tester nicht gemeinsam mit dem Auftraggeber (Product Owner) an einem Tisch sitzen können?
> wie viele Freiheitsgrade dürfen Verträge zulassen, wenn der interne Finanz-Freigabeprozess bereits in einer frühen Konzept-Phase einen exakten Business Case (mit allen Kosten) für das Gesamtvorhaben benötigt?
> wie viele Änderungen (Scope-Changes) können ohne maßgeblichen Einfluss auf die Faktoren Budget, Qualität oder Zeit vorgenommen werden, ohne die im Vorfeld definierten Gesamtkosten aus den Augen zu verlieren?

Ein agiler Evangelist mag sagen, dass die Fragen an sich bereits falsch sind – da diese in der idealen agilen Welt nicht aufkommen sollten. Allerdings sieht die Realität heute in den meisten Unternehmen anders aus. Zumindest in großen Unternehmen mit gewachsenen organisatorischen Strukturen und Lieferantenbeziehungen treten diese Fragen auf, sobald man sich zum ersten Mal mit agilen Vorgehensweisen beschäftigt.

Herausforderungen im Umgang mit agilen Vorgehensweisen

Eine Kernherausforderung bei der Einführung von agilen Vorgehensweisen in klassisch geprägten Unternehmen ist vor allem die Fehleinschätzung, es wäre ausreichend, einzelne Aspekte aus der agilen Welt zu adaptieren ohne von bestehenden, klassischen Strukturen und Prozessen zu abzuweichen. Dass die Einführung einer agilen Vorgehensweise aber mehr ist als nur die Umstellung eines Software-Entwicklungsprozesses oder das Weglassen wenig geliebter Aufgaben, gerät dabei im Rahmen der ersten agilen Gehversuche häufig in Vergessenheit und verwehrt dadurch den durchaus vielversprechenden agilen Ansätzen jegliche Chance auf einen Erfolg.

Der sicherlich naivste Weg zur Einführung einer agilen Vorgehensweise ist, sich ausschließlich auf den „Zeitfresser“ Anforderungserhebung und die bisher schwergewichtigen Anforderungsdokumente (z. B. Lastenhefte) zu fokussieren und diese durch einen leichtgewichtigen Prozess zur Erhebung und Dokumentation (z. B. User Stories) zu ersetzten. Dies ist zumindest dann fahrlässig, wenn der Austausch zu Anforderungen weiterhin ausschließlich über Dokumentenreviews stattfindet, man Verträge auf Basis dieser leichtgewichtigen Dokumentation schließt und das Produkt erst nach monatelanger Entwicklung zu Gesicht bekommt.

Vergleicht man die Aufgabe der Anforderungserhebung und Dokumentation in agilen und klassischen Vorgehensweisen, stellt man fest, dass es beiden Vorgehensweisen daran gelegen ist, ein gemeinsames Verständnis zu schaffen um auf dieser Basis ein hochwertiges Produkt zu entwickeln – sie unterscheiden sich allerdings in der Art und Weise, wie alle Beteiligten auf denselben Wissensstand gebracht werden.

In klassischen Vorgehensweisen ist der Requirements Engineer für die vollständige Ermittlung, Abstimmung, Dokumentation sowie für das Verwalten der Anforderungen verantwortlich. Durch seine Tätigkeit stellt er sicher, dass alle Beteiligten (Fachbereich, SW-Architekt, Entwickler, Tester) ein gemeinsames Verständnis von der gewünschten Lösung haben und die Gesamtheit der Anforderungen vollständig, widerspruchsfrei und verständlich dokumentiert sind. Agile Vorgehensweisen setzen hierbei eher auf leichtgewichtige Dokumentationen, die nicht jedes Detail beschreiben, da über den gesamten Entwicklungsprozess hinweg der Requirements Engineer bzw. der Product Owner Teil des Teams ist und Unklarheiten direkt geklärt werden können [Ste14]. Die leichtgewichtige Dokumentation z. B. über User Stories oder Story Cards, soll vielmehr über die Kommunikation im Team dafür sorgen, dass ein gemeinsames Verständnis geschaffen wird, was umgesetzt werden muss und über welche Akzeptanzkriterien die Erfüllung validiert wird (siehe auch [Jef01] drei Cs: „Card, Conversation, Confirmation“).

Die Verwendung leichtgewichtiger Dokumentation stößt vor allem dann an ihre Grenzen, wenn die vorhandenen prozessualen und organisatorischen Rahmenbedingungen noch keine idealtypische agile Welt zulassen [Röh13]. Diese Herausforderungen werden umso größer, je größer und komplexer das zu entwickelnde Produkt ist. Im Folgenden gehe ich zuerst auf einige Herausforderungen ein, die auftreten, wenn der Hauptfokus auf eine leichtgewichtige Dokumentation gesetzt wird, ohne dass die grundsätzlichen agilen Werte und Prinzipien (vgl. [Bec01]) korrekt verfolgt werden, bevor ich im nächsten Kapitel einige Lösungsansätze vorstelle.

Herausforderung 1: unflexible Lieferantenbeziehungen durch Festpreisverträge:

Wenn beide beiden Parteien gezwungen sind, den vollständigen Produkt-Scope, Zeit und Kosten in Verträgen festzuschreiben, ist eine leichtgewichtige Dokumentation problematisch. Fehlende Details stellen für beide Parteien Risiken dar – aus Projektsicht bedeuten fehlende Details unter Umständen zusätzlichen nicht kalkulierbaren Aufwand, welcher i. d. R. in einem Risiko-Puffer landet und den Gesamtaufwand künstlich in die Hohe treibt. Bei nicht vorhandenen oder ausgeschöpften Risiko-Puffern ist im schlimmsten Fall ein Change-Request und eine zusätzliche Budgetfreigabe erforderlich.

Herausforderung 2: fehlende Kommunikation durch organisatorische Trennung:

Eine strikte Linienorganisation mit organisatorischer und örtlicher Trennung der Fachbereiche (z. B. Produktmarketing, Vertrieb, Kundenbetreuung, Recht, Systemexperten, Entwicklung, Test) und gegebene Outsourcing bzw. Lieferantenbeziehungen (z. B. ausgegliederte Softwareentwicklung, Test, Betrieb) führen häufig zu räumlichen Kommunikationsbarrieren. Falls diese nicht beseitigt werden und zur Schaffung eines gemeinsamen Verständnisses über alle Anforderungen, ausschließlich auf Dokumenten-

Reviews und Freigaben gesetzt wird, ist eine leichtgewichtige Dokumentation nicht empfehlenswert. Fehlende Details führen bei den unterschiedlichen Parteien (Business, Entwicklung, Test, Betrieb) zu einer eigenen Interpretation der Lösung – Sie erinnern sich sicherlich an das bekannte Schaukelbeispiel.

Herausforderung 3: fehlende „Fast-Feedback“-Mentalität durch abgeschirmte Entwicklung:

Falls die Entwicklung weiterhin „im stillen Kämmerlein“ nach Wasserfall-Manier stattfindet und der Auftraggeber keine Zwischenergebnisse (z. B. Sprintergebnisse) zu sehen bekommt, wächst das Risiko von Design-Gaps durch Fehlinterpretation von Anforderungen, mit jedem Tag. Hier ist es unerheblich ob die Entwicklung „In-House“ oder bei einem externen Lieferanten in Bangalore stattfindet – einzig die Auflösung des Problems könnte bei einer „In-House“-Entwicklung einfacher werden. In Projekten, in denen Zwischenergebnisse nicht in regelmäßigen kurzen Abständen vorgestellt werden, sind leichtgewichtige Dokumente mit vielen Freiheitsgraden zu risikobehaftet und nicht zu empfehlen.

Herausforderung 4: fehlende ganzheitliche Produktsicht durch modulare Entwicklerteams:

Falls die Software-Entwicklung nicht innerhalb des agilen Teams stattfindet und über mehrere Abteilungen bzw. Lieferanten verteilt oder zu stark fragmentiert und beispielsweise nach einzelnen Modulen ausgerichtet ist, steigt das Risiko einer fehlenden ganzheitlichen Ende-zu-Ende Sicht. Je verteilter die Entwicklungsteams sind, umso mehr formale Dokumente und Abstimmungen (z. B. zu Interfaces) sind nötig. Falls es nicht möglich ist, dass das gesamte agile Team zusammen in einer gemeinsamen physikalischen Räumlichkeit (Co-Location) arbeitet, gerät eine leichtgewichtige Dokumentation schnell an Ihre Grenzen und es müssen geeignete Maßnahmen ergriffen werden, um allen Entwicklerteams ein Ende-zu-Ende-Verständnis zu geben und, um ihr Bewusstsein zu abhängigen Schnittstellen zu stärken.

Herausforderung 5: schlechte Qualität des Produkts durch fehlende Einbindung des Testteams:

In agilen Vorgehensweisen wird das erstellte Produkt direkt innerhalb des agilen Teams validiert, sodass Tester direkt in diese Teams eingebunden sind. Falls eine Einbindung von Testern in das agile Team durch organisatorische Strukturen (z. B. Outsourcing) nicht möglich ist, besteht ein erhöhtes Risiko, dass das Produkt nur unzureichend getestet wird. Testteams, die gewohnt sind, ihre Testfälle mehr oder weniger ausschließlich auf Dokumentenlage zu erstellen, laufen bei agilen Vorgehensweisen mit leichtgewichtiger Dokumentation in das Problem, dass das neue Produkt auf Basis der Dokumente nicht gut genug verstanden werden kann, um eine ausreichende Testabdeckung zu gewährleisten. Das Resultat ist dann in der Regel ein unzureichend getestetes Produkt. Falls die organisatorischen Voraussetzungen (z. B. fehlende Ressourcen, Outsourcing, Wasserfall-Denken) verhindern, dass mindestens ein Vertreter aus den Testteams in das agile Team eingezogen wird, erhöhen leichtgewichtige Dokumentationen das Erfolgsrisiko und den Aufwand anstatt diesen zu reduzieren.

Herausforderung 6: schwerfällige Weiterentwicklung durch verlorenes Expertenwissen:

In Organisationen, die stark linienorientiert aufgesetzt sind, zerfallen agile Teams nach der Fertigstellung gern wieder und werden wieder ihrer Linie zugeordnet. So ist es nicht selten der Fall, dass neue Projekte im selben Produktkontext (z. B. Weiterentwicklungen eines bestehenden Produkts) durch vollständig andere Teams durchgeführt werden. Diesen Teams fällt es aufgrund der leichtgewichtigen Dokumentation schwer, den aktuellen Status Quo des Produktes zu ermitteln um etwaige Anpassungen in Bezug auf deren Auswirkung und Aufwand zu bewerten. Da das Wissen in agilen Vorgehensweisen vermehrt in den Köpfen der Teams existiert, ist eine gewisse Kontinuität der Teamzusammensetzung erforderlich, um den gewünschten Erfolg zu garantieren. Kann dies nicht gewährleistet werden, stoßen leichtgewichtige Dokumentationen ebenfalls im Rahmen der Evolution eines Produktes an ihre Grenzen und sind nicht zu empfehlen.

Im schlimmsten Fall treten diese Probleme nicht einzeln auf, sondern summieren sich als Konsequenz nicht vorhandener agiler Rahmenbedingungen mit einer leichtgewichtigen Dokumentation, sodass spätestens mit der Inbetriebnahme klar wird, dass die inkonsequente Vermischung von agilen und klassischen Vorgehensweisen, bereits im Design zu fehlerhaften Annahmen geführt hat, die in der Entwicklung konsequent weitergeführt und im Rahmen der Abnahme aufgrund mangelnder Dokumentation nicht erkannt wurden. Glücklicherweise ist die Realität nicht immer nur schwarz-weiß, sondern hat viele graue Schattierungen, sodass

einzelne Wissensträger das Ruder häufig noch herumreißen können.

Die genannten Herausforderungen haben selbstverständlich keinen Anspruch auf Vollständigkeit. Darüber hinaus, treten diese Herausforderungen nicht nur in agilen Vorgehensweisen auf, sondern existieren überall. Diese Herausforderungen sollen an dieser Stelle vor Augen führen, dass Agilität definitiv nicht funktioniert, wenn nur ein paar ungeliebte Aufgaben aus klassischen Vorgehensweisen gestrichen werden, um Zeit zu sparen, oder „beliebige Personen“ in einen Raum sperrt und diese als agiles Team bezeichnet. Agilität bedeutet vielmehr ein Umdenken für das gesamte Unternehmen und ein Hinterfragen bestehender Strukturen und Hierarchien, ein Hinterfragen bestehender Projekt- und Freigabeprozesse, sowie ein Hinterfragen bestehender Lieferantenbeziehungen.

Lösungsansätze

Um agile Projekte zukünftig erfolgreich umzusetzen, müssen vorerst die notwendigen organisatorischen Rahmenbedingungen geschaffen werden (siehe Kasten 4). Zusätzlich muss ein Umdenken auf allen Ebenen (organisatorisch, kulturell und kognitiv) – der Mitarbeiter und des Managements – stattfinden (siehe Kasten 5), um den Weg für ein erfolgreiches agiles Arbeiten zu ebnen.

Die in diesem Kapitel skizzierten Maßnahmen bieten erste Lösungsansätze für die zuvor beschriebenen Herausforderungen. Hierbei zahlen i. d. R. gleich mehrere Maßnahmen positiv auf die Lösung ein. So sind bspw. für Herausforderung 4 neben der richtigen Dokumentationstiefe (siehe Kasten 6) mehrere Maßnahmen hilfreich, um das Problem vollständig zu lösen (z. B., RB-3, RB-5, MS-4, MS-5).

Maßnahmen für **organisatorische Rahmenbedingungen** in agilen Projekten:

RB-1.	Vertragliche Grundlagen mit Lieferanten schaffen, um gemeinsam agil zu arbeiten und z. B. agile Festpreisangebote definieren [Ope18].
RB-2.	Organisatorische Projekt- und Freigabeprozesse an agile Vorgehensweisen anpassen.
RB-3.	Organisatorische Voraussetzungen für interdisziplinäre Teams schaffen.
RB-4.	Technische Voraussetzungen für agiles Arbeiten schaffen (DevOps Infrastruktur) [Pie18].
RB-5.	Räume schaffen, in denen gemeinsam gearbeitet wird (Collaboration Spaces, Co-Locations).
RB-6.	Dem Team die nötigen Befugnisse erteilen, um Entscheidungen treffen zu können [Ste14].

Kasten 4: Organisatorische Rahmenbedingungen

Maßnahmen für ein **Umdenken** in agilen Projekten:

MS-1.	Zulassen, dass zu Beginn nur ein Produkt existiert, das nicht alle Anforderungen erfüllt aber bereits nutzbar ist und einen Mehrwert bringt (Mimimal Viable Product) [Kni16].
MS-2.	Verstehen, dass korrekte Priorisierung hilft, um schnellstmöglich Mehrwert (Business Value) zu schaffen [Rei09].
MS-3.	Verstehen, dass De-Priorisieren nicht das grundsätzliche Verlieren eines Features bedeutet – sondern das es Anforderungen gibt, die aktuell wichtiger sind [Rei09].
MS-4.	Verstehen, dass das agile Team gemeinsam für die Erstellung der Lösung verantwortlich ist.
MS-5.	Einfordern, von Feedback und Annehmen von Änderungen, um ein qualitativ hochwertiges Produkt zu erstellen, z. B. im Rahmen von Review-Sessions nach jedem Sprint [Bot16].

Kasten 5: Umdenken in agilen Projekten (Mind-Shift)

Zur Schaffung eines gemeinsamen Verständnis sind neben den genannten Maßnahmen weitere methodische Fähigkeiten notwendig, um den leichtgewichtigen Anforderungsprozess so effizient wie möglich zu gestalten und ein Mindestmaß an konsistenter Dokumentation zu erstellen, die auch nach einem Entwicklungszyklus (Sprint) genutzt werden kann, um die Abhängigkeiten zu bestehenden Features im nächsten Sprint besser zu verstehen bzw. das Produkt über die nächsten Jahre weiterentwickeln zu können. Aus diesem Grund darf und sollte man sich in der Rolle des Requirements Engineers in agilen Projekten die Frage stellen: Wie leichtgewichtig darf meine Dokumentation sein,

(a) um eine agile Vorgehensweise optimal zu unterstützen (so wenig „Abfall“ wie möglich) und
(b) um ein gemeinsames und vollständiges Verständnis für das laufende Projekt und
(c) um eine verständliche Dokumentation für Folgeentwicklungen zu schaffen?

Dokumentieren Sie ihre Anforderungsartefakte im Rahmen einer agilen Vorgehensweise

- **so genau wie nötig**: d. h. so viel wie es braucht um ein gemeinsames Verständnis zu schaffen und als Basis für Folgeentwicklungen zu dienen) und
- **so leichtgewichtig wie möglich:** d. h. so wenig wie möglich, um das agile Paradigma nicht zu torpedieren und den Aufwand in Grenzen zu halten.

Kasten 6: Die richtige Dokumentationstiefe finden

Als Antwort darauf schreibt die agile Community „Just Barely Good Enough“, vgl. [Amb05] – also „gerade noch gut genug“. Der wesentliche Grund für diese Aussage ist, dass in agilen Vorgehensweisen die dokumentierten Artefakte nur als eine Art temporäre Gedankenstütze dienen sollen, um die Kommunikation zu starten oder um Rückfragen im Rahmen der Entwicklung direkt mit dem Requirements Engineer bzw. Product Owner zu klären, um ein gemeinsames Verständnis zu schaffen. Damit die dokumentierten Artefakte einen Mehrwert für alle Herausforderungen bringen, plädiere ich an dieser Stelle für „so genau wie nötig aber so leichtgewichtig möglich“, siehe Kasten 6.

Auch oder vor allem im agilen Vorgehen bietet es sich an, in unterschiedlichen Abstraktionsebenen zu denken und zu dokumentieren (z. B. Themes, Epics, User Stories). Da agile Vorgehensweisen in der Regel keine zusammenhängenden Anforderungsdokumente erstellen, die das gesamte Produkt beschreiben, kann u. U. schnell der Überblick verloren gehen. Als Unterstützung können hier User Story Maps [Pat14] genutzt werden. User Story Maps bringen das sonst eindimensionale und unsortierte Product Backlog in eine zweidimensionale Sicht, wobei die Horizontale die Ende-zu-Ende-Prozesskette darstellt und die Vertikale die User Stories in Releases unterteilt, siehe Abbildung 1.

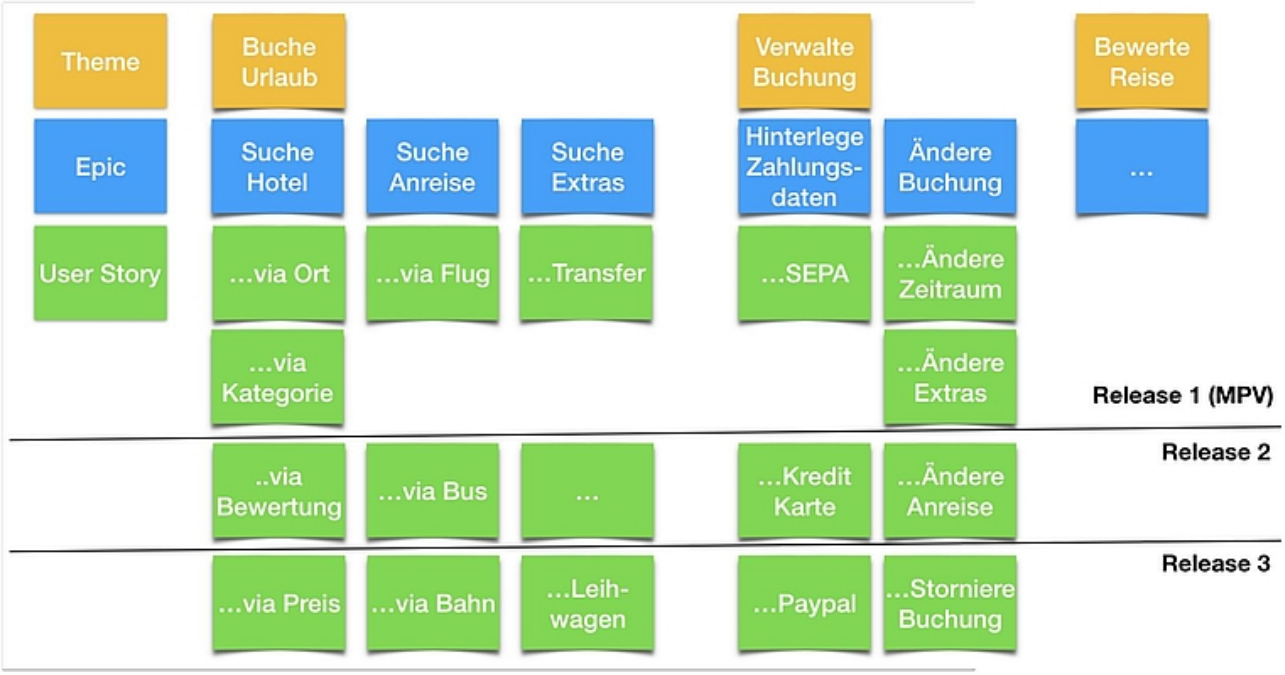

Abbildung 1: Beispiel User Story Map

Viel wichtiger, als eine ausreichende Dokumentation ist allerdings die Lösung der organisatorischen und prozesstechnischen Herausforderungen, um den tatsächlichen Vorteil agiler Vorgehensweisen zu erleben. Können die Rahmenbedingungen nicht geschaffen werden, so sind agile Vorgehensweisen für das jeweilige Unternehmen eventuell einfach nicht geeignet – hier ist auch ein „Mehr“ an Dokumentation nicht ausreichend, um die prozessuale und organisatorische Inkompatibilität abzufangen. An dieser Stelle sind dann eventuell iterative Vorgehensweisen, z. B. nach V-Modell [Ang06], besser geeignet, um schrittweise Ergebnisse

zu produzieren, die kontinuierlich verbessert und weiterentwickelt werden.

Zusammenfassung und Ausblick

Agile Vorgehensweisen erfordern ein Umdenken im gesamten Unternehmen. Zudem müssen die organisatorischen Rahmenbedingungen geschaffen werden, um agile Entwicklung effizient zu ermöglichen. Der Versuch der Einführung von agilen Vorgehensweisen, ohne sich mit den genannten Herausforderungen zu beschäftigen, die sich aus einer strikten Linienorganisation, gewachsenen Lieferantenbeziehungen, sowie im Rahmen von Offshoring und Nearshoring ausgelagerten Organisationseinheiten ergeben, endet wenig überraschend eher im Chaos als im Erfolg. Das soll nicht heißen, dass agile Vorgehensweisen für große Unternehmen nicht geeignet sind – sondern vielmehr, dass agile Vorgehensweisen ein Umdenken im gesamten Unternehmen bedeuten und entsprechende Voraussetzungen für eine erfolgreiche Einführung geschaffen werden müssen.

Agile Vorgehensweisen bringen ein neues Verständnis in den kreativen aber dennoch ingenieurmäßig geprägten Software-Entwicklungsprozess und können in vielen Projekten zum Erfolg beitragen. Ob agile Vorgehensweisen nun allerdings ein „Silver Bullet“ sind, bleibt abzuwarten.

In den kommenden Jahren wird sich zeigen,

- ob jede Projekt-Art (z. B. innovative Neuentwicklung, Ablösung bestehender Systeme, Erweiterung bestehender Systeme) für agile Vorgehensweisen geeignet ist,
- welche Lieferanten- und Vertragsbeziehung (z. B. Festpreis vs. Time-and-Material vs. agiler Festpreis) am geeignetsten ist, um agile Vorgehensweisen erfolgreich zu nutzen,
- ob und wie ausgegliederte Organisationsbereiche (Test, Betrieb) in Near- und Offshore Ländern in agile Vorgehensweisen (Stichwort DevOps) passen.

Ich freue mich auf neue Erkenntnisse in diesem Bereich und glaube persönlich daran, dass agile Vorgehensweisen ihre Vorteile gegenüber den klassischen Vorgehensweisen haben – diese aber auch nur unter den richtigen Rahmenbedingungen zum gewünschten Erfolg führen können. Es wird vor allem für Unternehmen mit gewachsenen Organisationsstrukturen und Prozessen noch ein langer und steiniger Weg, um agil und wendig zu werden. Wahrscheinlich wird sich auch herausstellen, dass nicht jedes Unternehmen vollständig für ein agiles Vorgehen geeignet ist, sodass wir auch in Zukunft immer noch klassische Vorgehensweisen neben agilen Vorgehensweisen antreffen werden.

Literatur & Links

[Amb05] Scot Ambler. „Just Barely Good Enough Models and Documents: An Agile Best Practice“.
Internet: http://agilemodeling.com/essays/barelyGoodEnough.html (Stand 25.05.2018)

[Ang06] Daniel Angermeier et al. „Das V-Modell XT“. Verein zur Weiterentwicklung des V-Modell XT e.V.
Internet: http://ftp.tu-clausthal.de/pub/institute/informatik/v-modell-xt/Releases/2.2/Dokumentation/V-Modell-XT-HTML/index.html (Stand 30.05.2018)

[Bec01] Kent Beck et. al. „Manifesto for Agile Software Development”.
Internet: http://agilemanifesto.org (Stand 25.05.2018)

[Bot16] Christian Botta „Sprint Review“.
Internet: https://www.projektmagazin.de/methoden/sprint-review (Stand 30.05.2018)

[Bro86] Fred P. Brooks. „No Silver Bullet — Essence and Accident in Software Engineering“. Proceedings of the IFIP Tenth World Computing Conference, 1986: S. 1069–1076.

[Kni16] Henrik Kniberg. „Die Bedeutung des MVP verstehen“ (übersetzte Version) -
Internet: https://www.scrumakademie.de/product-owner/wissen/die-bedeutung-des-mvp-verstehen (Stand 30.05.2018)

[Pat14] Jeff Patton. „User Story Mapping - Discover the Whole Story, Build the Right Product“. O’Reilly UK Ltd, 2014

[Pie18] Frank Pientka. „Wie DevOps die IT beschleunigen“. Computerwoche 22.02.2018.
Internet: Link: https://www.computerwoche.de/a/wie-devops-die-it-beschleunigen,3071433 (Stand 30.05.2018)

[Jef01] Ron Jeffries. „Essential XP: Card, Conversation, Confirmation“.
Internet: https://ronjeffries.com/xprog/articles/expcardconversationconfirmation/ (Stand 25.05.2018)

[Ope18] Andreas Opelt, Boris Gloger, Wolfgang Pfarl, Ralf Mittermayr. „Der agile Festpreis: Leitfaden für wirklich erfolgreiche IT-Projekt-Verträge”. Carl Hanser Verlag 2018

[Rei09] Donald G Reinertsen. „The Principles of Product Development Flow: Second Generation Lean Product Development. Redondo Beach”, California, Celeritas Publishing 2009

[Röh13] Oliver Röhrsheim. „Wesentliche Rahmenbedingungen bei der Einführung agiler Entwicklungsmethoden” Objekt-Spektrum Themenspecial Agility 2013.
Internet:
https://www.sigs-datacom.de/fachzeitschriften/objektspektrum/archiv/artikelansicht/artikel-titel/wesentliche-rahmenbedingungen-bei-der-einfuehrung-agiler-entwicklungsmethoden.html (Stand 30.05.2018) PDF:
https://www.sigs-datacom.de/uploads/tx_dmjournals/Roehrsheim_OS.Agility_2013.pdf

[Ste14] Bernd Stehr, Artur Strasser. „Der Requirements Engineer – Auslaufmodell oder Erfolgsfaktor?“. Objekt-Spektrum Online Themenspecial Requirements Engineering 2014.
Internet: https://www.sigs-datacom.de/uploads/tx_dmjournals/strasser_stehr_OS_RE_2014_8SLW.pdf (Stand 30.05.2018)

Stan Bühne

arbeitet als Principal Expert bei der SEVEN PRINCIPLES AG.
Mit seiner langjährigen Erfahrung im klassischen und agilen Projekt Management und Requirements Engineering, verantwortet er als Projektmanager und Business Analyst, die erfolgreiche Projektabwicklung, bei namenhaften Kunden der Telekommunikationsbranche.
Neben seiner beruflichen Tätigkeit ist Stan Bühne aktives Mitglied im IREB e.V. und an der Erarbeitung von Curricula beteiligt.

E-Mail: Stan.Buehne(at)7p-group.com

Bildnachweise:

Stan Bühne, Teaserbild: Pixabay