IM SPANNUNGSFELD:
AGILITÄT UND SICHERHEITSKRITISCHE SYSTEME

S. Wertheimer[1], Markdorf, DE; C. Haußmann[2], Lindau (Bodensee); DE

[1] one-small-step.de [2] butterflying.de

Zusammenfassung

Bei der Entwicklung von sicherheitskritischen Systemen finden stark formalisierte und standardisierte Prozesse Anwendung, die als lange erprobt gelten. Dies betrifft nicht nur die Luft- und Raumfahrt, sondern auch andere Industriezweige wie etwa den Automobilbau und die Medizintechnik. Um die benötigte Qualität zu wahren und Vorgaben für die Berücksichtigung der Sicherheitsaspekte zu gewährleisten, wurden die entsprechenden Prozesse von internationalen Behörden normiert. Ihre Einhaltung ist verpflichtend, um eine Zertifizierung für den jeweiligen Markt zu erhalten. Wegen ihres Alters und der hohen Anforderungen an die Sicherheit orientieren sich fast alle einschlägigen Standards zu diesem Thema am V-Modell der Deutschen Bundesregierung, das in seinem historischen Kern einem erweiterten Wasserfallmodell entspricht. Im Bereich der Software-Entwicklung wären hier z.B. die Normen DO-178C für die Luftfahrt, ECSS-E-ST-40C für die Raumfahrt und ISO 26262 für die Automobilindustrie zu nennen.

Nun gibt es in vielen anderen Branchen seit Mitte der 1990er Jahre eine Bewegung von Software-Entwicklern, die versucht, das starre Korsett der alten Vorgehensmodelle aufzubrechen und das eigentliche Produkt und die damit beschäftigten Menschen in den Fokus zu rücken. Diese neuen Denkmuster und Herangehensweisen werden unter dem Schlagwort „Agilität“ zusammengefasst. Dabei sollen aber nicht um jeden Preis anscheinend veraltete Methoden abgeschafft werden. Es geht vielmehr um die Identifikation von unnötiger Arbeit oder ungewollter Redundanzen, um Zeit- und Energieverschwendung während der Entwicklung zu vermeiden und die tatsächliche Wertschöpfung wieder in den Mittelpunkt zu stellen. Ebenso wird bei diesen Vorgehensweisen versucht, die stetig wachsende Komplexität der Systeme und der Welt durch neue Arbeitsweisen und Ansätze beherrschbar zu machen. Durch die inzwischen stetig wachsende Verbreitung, insbesondere im Bereich der Webentwicklung und der Startup-Szene, wird auch in den eher traditionellen Industriezweigen versucht agile Methodiken einzuführen. Dies erfolgt meist wegen des versprochenen Erfolges, den der Hype um die neuen Techologiegiganten wie Google, Amazon und Co. erzeugt hat. Dabei kommt es während des eingeleiteten Wandels oft zu Problemen mit den etablierten Prozessen der oben genannten Normen. Dieses Paper behandelt das entstehende Spannungsfeld zwischen agilen Ansätzen und den etablierten Normen für die Entwicklung sicherheitskritischer Software und Systeme. Neben dem Appell, die durchaus sinnvollen und gewinnbringenden Prozessanforderungen trotzdem kritisch zu hinterfragen, werden auch allgemeine Lösungsansätze für eine Verbindung der beiden Welten vorgeschlagen. Durch ein besseres Verständnis der beiden Seiten wird klar, dass es keinen „Heiligen Gral“ geben kann, aber eine kritische Auseinandersetzung mit den Kernaspekten und eine Vermeidung überflüssiger Arbeiten in Zukunft unabdingbar sein wird, um die immer komplexeren und komplizierten Abläufe und Systeme verstehen und umsetzen zu können.

Schlagworte

Agilität, Normen, Sicherheitskritische Systeme, Komplexität, VUCA, Cynefin

1. DER TRAUM VOM FLIEGEN UND DIE SCHÖNE NEUE WELT

Der Traum vom Fliegen ist bekanntlich sehr alt. In seiner aktuellen Form geht er auf die erfinderischen Bemühungen von Menschen zurück, die sich die Kenntnisse selbst beibringen mussten.

Albrecht Ludwig Berblinger (auch bekannt als der „Schneider von Ulm“) unternahm unzählige Flugversuche in den Weinhängen des Ulmer Milchbergs. Er verbesserte seinen Gleiter und sein Verständnis von Thermik und Winden kontinuierlich durch neue Erkenntnisse bis zu seinem verhängnisvollen Schauflug am 31. Mai 1811 von der Adlerbastei über – oder eher in – die Donau.

Ähnlich ging auch der Berliner Otto Lilienthal Ende des neunzehnten Jahrhunderts vor: Theoretische Vorarbeit, stetige Verbesserung und die kontinuierliche Erprobung ließen ihn den Gleitflug

doi:10.25967/490221

soweit voranbringen, dass er als „erster Flieger der Menschheit“ gilt.

Zur gleichen Zeit begannen die Brüder Wilbur und Orville Wright in den Vereinigten Staaten von Amerika mit dem Entwurf und Bau von Gleitern. Die beiden Fahrradmonteure waren Autodidakten, die nicht einmal einen Highschool-Abschluss vorweisen konnten. Sie sammelten alle Literatur, die sie zum Thema „Menschenflug“ finden konnten, und bauten immer neue Gleiter für Testflüge. Noch vor der Jahrhundertwende konnten sie Fehler beim Smeaton-Koeffizient in Lilienthals Tabellenwerken entdecken. Am 17. Dezembers 1903 erfolgte schließlich der berühmte Erstflug des „Wright-Flyers“.

Über die kommenden Jahrzehnte folgten viele weitere Entwicklungsschritte und Erkenntnisse, die meist wieder durch Versuche und Empirik gewonnen wurden. Dieser Erfindergeist im Ingenieurswesen ist auch verständlich, wenn man die Wirkung von Franz Reuleaux und Alois Riedler Ende der 1880er berücksichtigt. In Lehrbüchern zum Maschinenbau betont Riedler stets die Empirik als wichtige Bestätigung von Theorien, während Reuleaux versucht, aus den Erkenntnissen eine exakte Wissenschaft abzuleiten.[1]

Ähnlich, wenn auch zeitversetzt, ging die Entwicklung in der Raketentechnik von Konstantin Ziolkowski und Hermann Oberth bis zur verlässlichen Produktion und dem zuverlässigen Start des Aggregats 4 und seiner leider traurigen, ja erschütternden Geschichte. Und auch im Automobilbau wurde zu Beginn mehr experimentiert als nach Plan gefertigt.

Nach mehreren Jahrzehnten der Erprobung und Verbesserung waren schließlich die technischen Rahmenbedingungen und das Wissen so weit, um in die plangesteuerte Massenproduktion zu gehen. Sehr früh wurden dabei die Automobile „am Fließband“ produziert. Henry Ford revolutionierte durch die Massenfertigung die Fortbewegung des Menschen. Er stützte sich dabei auch auf Theorien von Frederick Winslow Taylor und Henri Fayol. Aus dem komplexen Problem der „Technischen Machbarkeit“ wurde ein kompliziertes Problem des Zusammenbaus. Die Ingenieure konnten ihr empirisch gewonnenes und verbessertes Verständnis in Pläne und Skizzen niederschreiben. Ihre Schöpfungen konnten beliebig oft nachgebaut werden.

Die Prozesse der Fertigung wurden stetig verbessert, um eine gleichbleibende Qualität des Zusammenbaus zu gewährleisten. Standards und Normen wurden verfasst und eingeführt. Und in dieser normgetriebenen Welt befinden wir uns noch heute, als wäre die Welt stehen geblieben.

2. WARUM REICHEN DIE BLAUPAUSEN NICHT MEHR FÜR ALLE PROBLEME AUS?

Das heutige Umfeld befindet sich nun wieder in einem Wandel. Wir bewegen uns getrieben durch neue Technologien weg vom Industriezeitalter hin zum Wissenszeitalter. In diesem Umfeld besteht eine hohe Dynamik mit vielen Schwankungen und schlechter Vorhersehbarkeit, mit welcher Wahrscheinlichkeit ein Ereignis eintritt. Durch die Vernetzung multiplizieren sich die bestehenden Systeme und eine Ursache-Wirkungs-Beziehung ist schwer sichtbar. Hinzu kommt eine Mehrdeutigkeit, die zu einer nicht mehr planbaren Realität führt.

Diese Herausforderungen werden mit dem Akronym VUCA beschrieben.[2] Diese Betrachtungsweise ist die Antwort einer amerikanischen Militärhochschule auf den Zusammenbruch der UdSSR Anfang der 1990er (siehe Abbildung 1). Es gab plötzlich nicht mehr den einen Feind, wodurch eine neue Sichtweise notwendig war.

Abbildung 1: VUCA

Durch Zerlegen der Probleme in kleinere Schritte kann den Herausforderungen in diesem Umfeld begegnet werden. Agilität ist das passende Mindset für diese Welt.

2.1 Was bedeutet Agilität?

Agilität basiert auf Werten und Prinzipien. Durch das agile Manifest, verfasst von 17 Spezialisten aus dem IT-Umfeld im Jahre 2001, wurde dies das erste Mal durch Worte greifbarer (siehe Abbildung 2 auf der nächsten Seite).[3]

Agilität ist in erster Linie eine Grundhaltung und keine reine Ausführung von Methoden. Basis ist das sogenannte agile Mindset bzw. dynamische

[1] Vgl. König, W. (2014)
[2] Vgl. https://www.vuca-welt.de/
[3] Vgl. https://agilemanifesto.org/

Wir erschließen bessere Wege, Software zu entwickeln, indem wir es selbst tun und anderen dabei helfen. Durch diese Tätigkeit haben wir diese Werte zu schätzen gelernt:

- **Individuen und Interaktionen** mehr als **Prozesse und Werkzeuge**
- **Funktionierende Software** mehr als **umfassende Dokumentation**
- **Zusammenarbeit mit dem Kunden** mehr als **Vertragsverhandlung**
- **Reagieren auf Veränderung** mehr als **das Befolgen eines Plans**

Das heißt, obwohl wir die Werte auf der rechten Seite wichtig finden, schätzen wir die Werte auf der linken Seite höher ein.

Abbildung 2: Das Agile Manifest

Selbstbild. Dieses beinhaltet eine lösungsorientierte Denkweise, bei der sich die Person selbstständig einbringt und sich selbst, aber auch die Strukturen, in denen sie sich bewegt, weiterentwickelt.

Dazu gehört aber auch das ständige Hinterfragen, warum man etwas tut und ob es noch das Richtige ist: Diese kontinuierliche Reflexion der bestehenden Arbeitsweise ermöglicht die flexible Anpassung an neue Anforderungen, die durch das dynamische Umfeld erforderlich sind. Agilität ist somit die Kultur, die in einen Prozess mündet.

Erst durch vertrauensvolle Zusammenarbeit, eine ausgeprägte Kundenorientierung, gegenseitiges Verständnis und gedankliche Flexibilität sowie effektive Kommunikation können die derzeitigen Herausforderungen gemeistert werden. Auf Basis der Werte und Prinzipien entsteht am Ende eine Vorgehensweise mit folgenden Eigenschaften:

- Schlanke Entwurfsphase.
- Iteratives Vorgehen in kurzen Abständen.
- Frühzeitige und regelmäßig Lieferung von Ergebnissen, beispielsweise in Form von ausführbarer Software.
- Höchstmaß an Flexibilität und Anpassbarkeit.
- Kontinuierliche Reflexion durch Retrospektiven und Feedbackschleifen.

Für die Umsetzung dieser Eigenschaften stehen verschiedene Frameworks zur Verfügung. Innerhalb dieser Frameworks können unterschiedliche Praktiken abhängig vom Problem Anwendung finden.

2.2 Was haben alle agilen Vorgehensweisen gemeinsam?

Alle agilen Vorgehensweisen haben das PDCA-Modell (siehe Abbildung 3) als Grundlage.

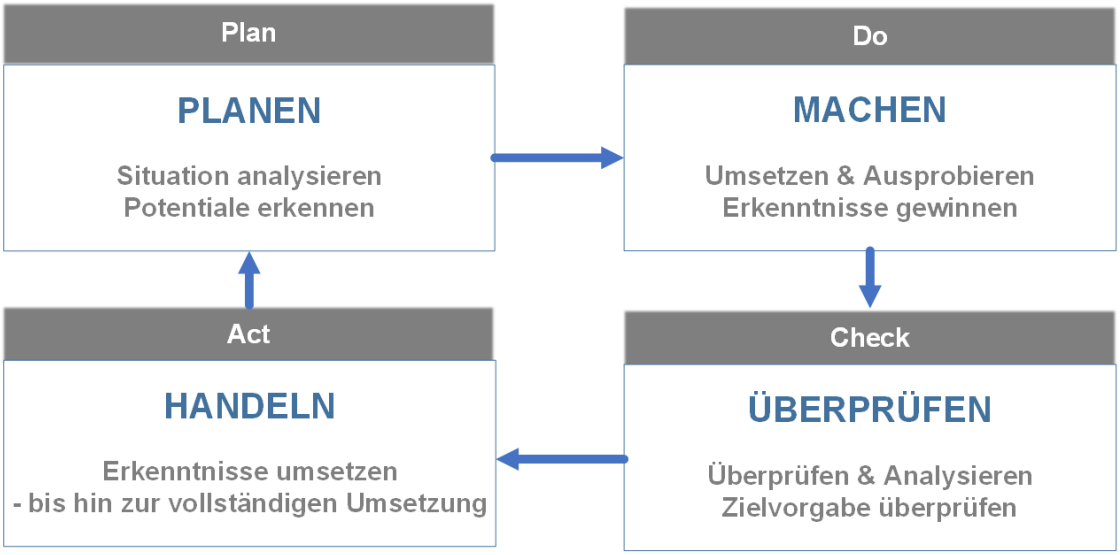

Abbildung 3: PDCA-Zyklus

Das Modell wurde in den frühen 1920er Jahren von Walter A. Shewhart entwickelt[4] und in den 1950er Jahren von William Edwards Deming weiterentwickelt und populär gemacht[5].

PDCA steht für Plan-Do-Check-Act und ist ein Basiskonzept des kontinuierlichen Verbesserungsprozesses. Ein gängiges Synonym für diese Vorgehensweise ist auch „Experiment". Ziel ist es, sich mit kleinen Schritten dem Problem anzunähern und durch regelmäßige Überprüfung sowie Feedback herauszufinden, ob man auf dem richtigen Weg ist.

2.3 Wie finde ich heraus, welche Vorgehensweise für mein Problem am besten geeignet ist?

Das Cynefin-Framework ist ein Wissensmanagementmodell des walisischen Wissenschaftlers David Snowden zur Beschreibung von Problemen, Situationen und Systemen.[6 & 7]

Das Modell basiert auf deren Einteilung in die Kategorien einfach, kompliziert, komplex und chaotisch (siehe Abbildung 4).

Abbildung 4: Cynefin-Framework

Für jede dieser Kategorien gibt es ein abgestimmtes Vorgehensmuster (Strategie), um das System zu steuern.

Das Cynefin-Framework hilft im Hinblick auf den Zusammenhang von Ursache und Wirkung die Probleme zu kategorisieren. Je nachdem, in welcher Kategorie sich das Problem befindet, bieten sich unterschiedliche Lösungsstrategien an.

Die Stacey-Matrix (nach dem britischen Forscher Ralph D. Stacey) versucht zusätzlich einen Zusammenhang herzustellen zwischen den Eigenschaften eines Systems nach dem Cynefin-Framework und der Qualität eines Arbeitsauftrags (Anforderung, Klarheit bei der Vorgehensweise bzw. Technologie).[8 & 9]

Es gilt: Je unklarer die Anforderungen und der Weg bzw. die Technologie sind, desto eher eignet sich eine agile Vorgehensweise (siehe Abbildung 5 auf der nächsten Seite).

3. WHEN WORLDS COLLIDE

Kommen wir wieder zurück zu den sicherheitskritischen Systemen wie Automobile, Flugzeuge und Raumfahrzeuge. Auch hier hat der Wandel begonnen, indem die Industrialisierung schrittweise durch die Digitalisierung ersetzt wird. Der technologische Fortschritt schlich sich in die vermeintlich bekannten und verstandenen Domänen (sei es Flugzeugbau, Raketentechnik oder Automobilbau).

Das greifbarste Beispiel sind die Kraftfahrzeuge: Vor zwanzig Jahren waren nur eine Hand voll Steuergeräte verbaut und ein Mechaniker konnte viele Probleme durch Wissen und technisches Geschick beheben. Heute sind es meist über 100 ECUs (Electronic Control Unit bzw. Steuergerät) und mehrere Kilometer Kabel. Der Mechaniker wurde zum Mechatroniker mit Diagnosegerät, ohne das viele Fehler gar nicht mehr verständlich sind. Erweiterungen im Bereich Autonomes Fahren, Elektromobilität und Fahrerassistenzsysteme (ADAS) und machen das komplizierte Problem „Automobil" wieder zu einem komplexen.

Die Luftfahrt befindet sich in einer ganz ähnlichen Situation und die Raumfahrt ebenso, auch wenn durch die schon lange vorhandene Autonomie der

[4] Vgl. Shewhart, Walter A. (1986)
[5] Vgl. Deming, William E. (1982)
[6] Vgl. Snowden, David J. (2000)
[7] Vgl. Kurtz, Cynthia F. & Snowden, David J. (2003)
[8] Vgl. Zimmerman, Brenda (2001)
[9] Vgl. Stacey, Ralph D. (1996 + 2003)

Systeme andere Aspekte zu betrachten sind.

Das Hinzufügen von immer mehr Elektronik, immer mehr Software und immer mehr Autonomie führen zu Problemen, die sich oft von den bekannten und erprobten Standards und Prozessen nicht mehr regeln lassen. Innovationen in Technologie und Software sind schneller verfügbar, als Entwicklungsprojekte abgeschlossen werden können.

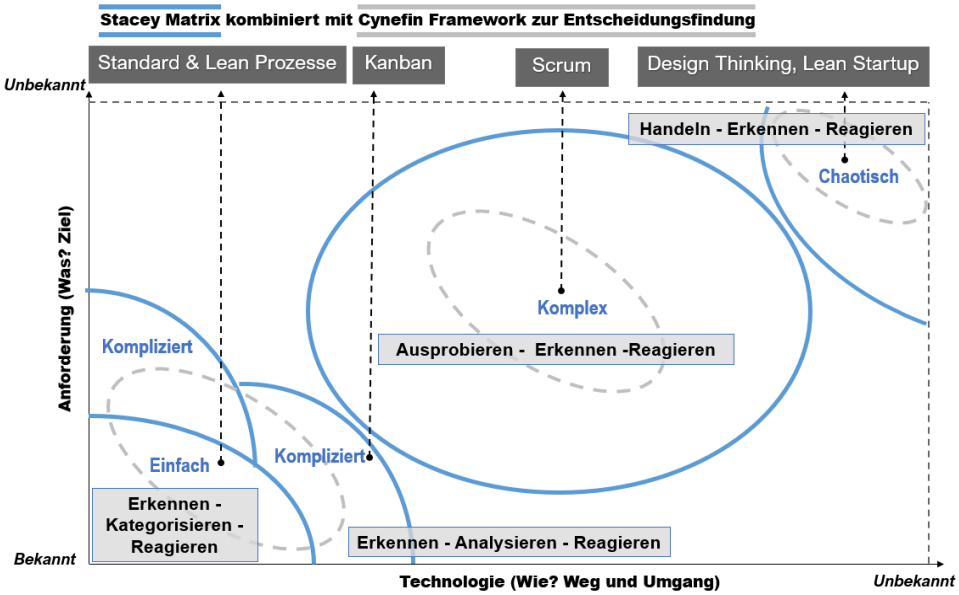

Abbildung 5: Stacey-Matrix

Die Maßnahmen zur Sicherung von Qualität und Sicherheit lassen das Vorankommen behäbig werden. Gleichzeitig sehen die Konsumenten und Kunden hinter den Produkten und Projekten immer neuere Ideen, die sich in Wünschen und Anforderungen manifestieren. Die Spezifikationen ändern sich permanent. Vermeintlich „einfache" eher aber komplizierte Probleme werden zu komplexen Herausforderungen, die durch den vorgegebenen Normprozess nur schwer oder gar nicht gemeistert werden können. Das Experimentieren und die empirische Datenerhebung durch Versuch und Erprobung sind in Vergessenheit geraten. Es herrscht oft die Denke vor: „Wenn wir es so machen, wie immer, wird es wie immer funktionieren" … oder eben nicht. Doch dies wird zu spät erkannt oder gekonnt übersehen. Verschwendung von Geld und Zeit ist die Folge.

Prozesse und Normen stoßen also an ihre Grenzen. Doch was ist nun mit diesem Allheilmittel aus der Software-Entwicklung, diesen „Agilen Methoden"? Zuerst entsteht hier wieder das übliche Missverständnis, das bereits weiter oben ausgeräumt wurde: Agilität ist weder Methode noch Prozess, Agilität ist eine Einstellung. Sehr aktuell postuliert Robert C. Martin, ein Co-Autor des Agilen Manifests, den Satz „Agile is a small idea about the small problems of small programming teams doing small things".[10] Die agile Welt hat also demzufolge keinen Anspruch, für große Probleme da zu sein und in diese Richtung zu agieren.

Dies führt uns zu folgender Hypothese:

Auf Grund des veränderten Umfelds muss über eine andere Sichtweise nachgedacht werden.

Nur die gewinnbringende Kombination aus agiler Organisation und normgetriebenen Sicherheitsaspekten kann neu entstehende Schwierigkeiten besser und schneller lösen.

4. EINE SWOT-ANALYSE DER AGILITÄT

Die SWOT-Analyse ist ein Instrument aus der Unternehmensentwicklung und strategischen Planung. Das englische Akronym SWOT für steht für Strengths (Stärken), Weaknesses (Schwächen), Opportunities (Chancen) und Threats (Risiken). Diese Analyseform ist bereits in den 1960ern an der Harvard Business School entstanden und wird unter anderem von namhaften Professoren wie zum Beispiel Henry Mintzberg vorangetrieben.

Nur durch die radikale Offenlegung der Stärken und Schwächen, der Chancen und Risiken ist eine objektive Bewertung möglich. Gerade bei der Kombination zweier Systeme oder Herangehensweisen offenbaren sich hier die Spannungsfelder. Sehr oft können aber die Schwächen eines Systems durch Stärken des anderen ausgeglichen werden.

Im Folgenden findet sich eine SWOT-Analyse der Autoren (siehe Abbildung 6 auf der nächsten Seite) mit dem Ziel, Agilität aus der Sicht von traditionellen Prozessen zu beleuchten.

5. BEST OF BOTH WORLDS

Betrachten wir nun beispielhaft die ermittelten Stärken. Hier zeigen sich deutlich Vorteile für den

[10] Martin, Robert C. (2019)

Stärken	Chancen
● Häufige Lieferung mit hoher Qualität. ● Durch kontinuierliche Rückkopplungsschleifen und Abgleich der Bedürfnisse der Anwender wird die ordnungsgemäße Umsetzung der Anforderungen sichergestellt. ● Schnelle Reaktion auf sich ändernde Anforderungen - von Iteration zu Iteration. ● Minimierung von Risiken. ● Vermeidung von Verschwendung (Lean-Gedanke) durch Konzentration auf das Wesentliche.	● Probleme und Unsicherheiten werden durch mehr Transparenz schneller sichtbar. ● Neue Fähigkeiten (agile Architektur, Testautomatisierung) können etabliert werden. ● Die Kundenzufriedenheit kann gesteigert werden. ● Durch die Einbeziehung kann der Widerstand bei neuen Funktionen bei der Einführung reduziert werden. ● Eigenverantwortung fördert die Motivation.
Schwächen	**Risiko**
● Langfristige Planung im Detail ist nicht angedacht. ● Neue Vertragsgestaltungen sind notwendig: Das sogenannte magische Dreieck des Projektmanagements ist auf den Kopf gestellt. Kosten, Zeit und Qualität sind die festen Parameter und der Inhalt ist nun variabel. ● Der Einsatz von verteilten Teams ist eine Herausforderung. ● Implementierung von Service Level Agreement (SLA) ist schwieriger. ● Bei Unterbrechungen der Arbeit oder der Reaktion auf Notfälle gelten andere Regeln als bisher.	● Management unterstützt die agile Arbeitsweise nicht. ● Es ist kein Umfeld vorhanden, in dem Fehler gemacht werden dürfen und Entscheidungen unabhängig der Hierarchie getroffen werden können. ● Es werden nicht ausreichend die nötigen Werte wie Offenheit und Mut gelebt. ● Es fehlt Transparenz auf allen Ebenen. ● Entscheidungen des Product Owners werden von der Organisation nicht akzeptiert. ● Das Team hat nicht die Fähigkeit den neuen Weg zu beschreiten. ● Stakeholder sind nicht bereit sich einzubringen oder miteinander zu reden. ● Es entsteht zu wenig Dokumentation für Außenstehende.

Abbildung 6: SWOT-Analyse Agiles Mindset

Erfolg in der modernen VUCA-Welt. Viele der Punkte finden sich in analogen Analysen zu traditionellen Vorgehensmodellen wie Wasserfall oder auch dem klassischen V-Modell (vor XT 2.0) als Schwächen wieder. Meilenstein-basierte Entwicklungsprozesse können auf Grund ihrer Struktur nicht mit Veränderung umgehen. Dafür sind sie gut für die Gewährleistung von Sicherheit und Qualität geeignet.

Doch auch hier zeigt sich auf, dass Qualität nicht durch eine Norm oder einen Prozess erzeugt wird, sondern durch die Menschen, die ihn anwenden. Transparenz und der Sinn hinter einer Tätigkeit spielen eine entscheidende Rolle. Diese Werte sind im Agilen Mindset tief verwurzelt, sodass hier der Kulturwandel in der Denkweise zu einer Qualitätssteigerung führen kann.[11] Die Mitarbeiter sind selbst wieder für ihr Tun verantwortlich und nicht eine abstrakte Norm, die für angeblichen Erfolg sorgt.

Dieser „gesunde Menschenverstand“, von dem so oft die Rede ist, der aber meist durch Vorschriften erstickt wird, ist es, der auch die Umsetzung einer Norm zum Erfolg führt. Ein Plan alleine ist nicht die Lösung, aber das Wissen um die Sinnhaftigkeit seiner Existenz und die Durchführung sind dagegen der Weg zum Ziel.

Durch die ständige Reflektion der Arbeitsweise, wie sie einem jeden agilen Ansatz zu eigen ist, werden Probleme am Prozess früh erkannt. Die Besinnung auf das Wesentliche und auf das Warum rücken die Zielerreichung wieder ins richtige Licht. Ein Effekt, der zwar auch durch das klassische Audit auf Deckung mit Normkonformität erreicht werden kann, aber selten den kurzen Feed-

[11] Vgl. Hofert, Svenja (2018)

back-Zyklus und die Kontinuität der stetigen Retrospektive bietet. Das bedeutet, Fehler werden oft deutlich später erkannt und behoben.

6. HILFREICHE METHODEN

Auch wenn jetzt Agilität als Denkmuster verstanden wird, so gibt es doch Praktiken und Methoden, die in vielen Ausprägungen der agilen Welt Einzug gefunden haben. Sie haben sich bewährt, den Fokus zu behalten und die gelebten Werte zu unterstützen.

In den folgenden Kapiteln werden einige Methoden vorgestellt, die aus Sicht der Autoren jedes Projekt, insbesondere solche mit sicherheitskritischen Aspekten, voranbringen. Die meisten sind zwar aus der Sicht der Software beschrieben, aber viele können auch in angepasster Form in Hardware-Projekten eingesetzt werden.

Zum Abschluss wird an Hand der Raumfahrtstandards ECSS-E-ST-40C und ECSS-Q-ST-80C (Rev. 1) gezeigt, wie die vorgestellten Methoden gewinnbringend in den sicherheitskritischen Entwicklungsprozess integriert werden können.

6.1 Test Driven Development (TDD)

Eine testgetriebene Entwicklung bedeutet die konsequente Erstellung von Tests und Denkarbeit vor der eigentlichen Entwicklung nach dem sogenannten Test-First-Principle. Kent Beck, ein US-amerikanischer Software-Entwickler und Co-Autor des agilen Manifests, gilt als Erfinder oder eher als „Wiedererfinder“ dieser Methodik.[12]

Bei diesem Ansatz geht der Implementierung immer die Erstellung von Unit-Tests voran, wobei zunächst das Verhalten der zu implementierenden Klasse festzulegen ist. Dazu zählen sowohl das erwartete fehlerfreie Verhalten als auch mögliche Fehlerfälle. Anschließend erfolgt die Implementierung in kleinen überschaubaren Einheiten und mit möglichst geringem Aufwand. Im darauffolgenden Schritt werden Verbesserungen mittels Refactoring durchgeführt, ohne das Verhalten des Programmcodes zu ändern. Sichergestellt wird dies durch die bereits vorhanden, umfassenden (automatisierten) Tests, die über diesen Weg kontinuierlich weiterentwickelt werden. Durch die mit dieser Technik angestrebte hohe Testabdeckung entsteht robuster Programmcode.

Das Aufteilen eines Problems in beherrschbare Mengen kleiner, leicht zu lösender Teilprobleme sowie die kontinuierliche Verbesserung durch das Refactoring sind die Schlüsselfaktoren zum Erfolg dieses Vorgehens.

6.2 Continuous Integration (CI)

Unter Continuous Integration versteht man das ständige Zusammenführen und Testen der aktuellen Entwicklungsstände auf einer zentralen Instanz der Entwicklungsumgebung. Dies erfolgt durch eine automatische Abfolge: Zuerst wird der Source Code gebaut und im Anschluss werden vorhandene Unit-, Komponenten- und Integrationstests in dieser Reihenfolge ausgeführt. Sobald ein Schritt der Kette fehlschlägt (z.B. Build-Fehler oder nicht erfolgreicher Test), erhält der Entwickler sofort entsprechendes Feedback, um das Problem unmittelbar beheben zu können.

6.3 Definition of Done (DoD)

Die Definition of Done ist ein Regelwerk zur Qualitätssicherung. Es werden alle Akzeptanzkriterien als Liste definiert, die erforderlich sind, um eine lauffähige Software für den Produktiveinsatz zu erstellen. Dies sorgt für ein gemeinsames Verständnis, was alles zur Fertigstellung benötigt wird und stellt die allgemeinen Qualitätskriterien an das Produkt dar.

Ergänzend zur Definition of Done kann auch eine sogenannte Definition of Ready (DoR) mit Qualitätskriterien für Anforderungen gepflegt werden. Für das Team gibt die Definition of Ready Anhaltspunkte, wann eine Anforderung zur Umsetzung bereit ist.

Für jede Anforderung, beispielsweise in Form einer User Story, können weitere spezifische Akzeptanzkriterien erfasst werden. Darin können sich beispielsweise die nicht-funktionalen Anforderungen widerspiegeln. Das generelle Ziel dieser Regelwerke ist es, Transparenz über die erstellte Software und deren Qualität zu schaffen.

6.4 Prototyping

Der Zweck von Prototypen besteht darin, kritische Aspekte einer Software so früh wie möglich zu erkennen und zu beseitigen. Bei der Erstellung von Prototypen kann zwischen evolutionären und experimentellen Vorgehen unterschieden werden:

- Mit **evolutionären Prototype**n wird die Funktionalität schrittweise aufgebaut und erweitert.
- Das **experimentelle Prototyping** erprobt verschiedene Umsetzungsmöglichkeiten und vergleicht deren Ergebnisse miteinander. Der Verwendungszweck kann dabei sehr unterschiedlich sein: Ein einfaches, grobes Modell

[12] Vgl. Beck, Kent (2003)

zur Demonstration, Prototypen zur Untersuchung von technischen Problemen oder mit einem hohen Reifegrad als Pilotsystem für realitätsnahe Tests und Bewertung.

Zusätzlich kann beim Design der Software zwischen dem horizontalen und vertikalen Prototyp unterschieden werden (siehe Abbildung 7):

- Ein **horizontaler Prototyp** realisiert nur einen ausgewählten Bereich einer Architektur. Üblich sind sogenannte GUI-Prototypen, die nur die Benutzerschnittstelle demonstrieren, ohne eine darunterliegende technische Funktionalität umzusetzen. Ziel dieser Prototypen ist es, den Anwender frühzeitig mit der Benutzerschnittstelle vertraut zu machen und Feedback zu erhalten.
- **Vertikale Prototypen** (auch Durchstich genannt) dient dazu, ausgewählte Aspekte in Ihrer Gesamtheit zu zeigen. Das bedeutet, dass ein Teil des Systems über alle Schichten der Architektur hinweg erstellt wird; also beginnend von der Benutzerschnittstelle bis zur Datenhaltung. Diese Art Prototyp dient dazu, komplexe Funktionalität zu demonstrieren und dem Anwender zur Prüfung zu übergeben. Dies ist in der Regel der Ansatz einer jeden User Story, um immer ein lauffähiges Produkt zu haben.

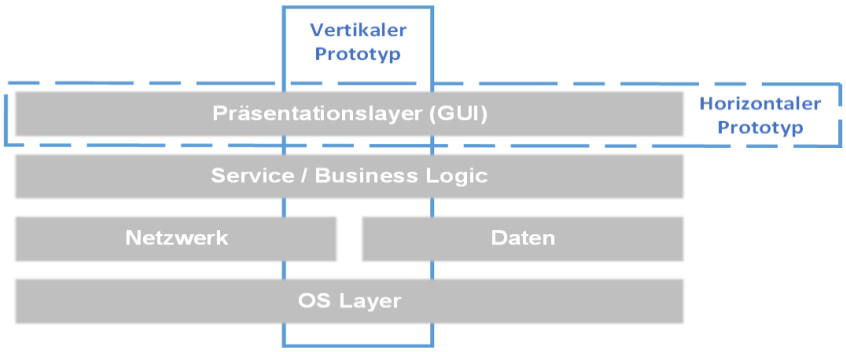

Abbildung 7: Arten des Prototypings

6.5 Wie Methoden Prozesse unterstützen

Wie bereits am Anfang des Abschnitts erwähnt, sollen die Methoden nicht unreflektiert aufgelistet werden, sondern es soll auch ihr Vorteil für die Entwicklung sicherheitskritischer Systeme dargestellt werden. Als Beispiel werden die Software-Standards ECSS-E-ST-40C und ECSS-Q-ST-80C (Rev 1) des ECSS bzw. der ESA beleuchtet.

Test Driven Development schafft bereits sehr früh ein Verständnis für die Testbarkeit und die Testabdeckung von Source Code. Das ECSS fordert in ihren Standards z.B. explizit die Durchführung von Unit Tests für Software an Bord von Raumfahrtsystemen und das z.T. mit sehr hohen Abdeckungsanforderungen. Durch die frühzeitige Implementierung der Tests und die parallele Adaption bzw. Ergänzung bei der weiteren Entwicklung wird die geforderte Coverage verzögerungsfrei und transparent erreicht. Die Quantität und Qualität der Tests wächst zu dem Zeitpunkt, an dem sie auch benötigt werden: Bei der Implementierung neuer Funktionalität.

Ebenso werden die Qualitätsanforderungen an Verifikation und Nachverfolgbarkeit auch durch den Einsatz von *Continuous Integration* unterstützt. Durch die ständige Bereitstellung von Metriken und aktuellen Testergebnissen kann die Qualität sehr gut bewertet werden und schnelle Reaktion bei Abweichung ist möglich. Die gesammelten Daten unterstützen die Qualitätsaudits. Diese können parallel auf *Definition of Done* und *Definition of Ready* aufsetzen, wenn man diese aus Qualitätssicherungssicht als Checklisten versteht. In einem Audit kann dadurch zumindest ein gemeinsames Grundverständnis mit den Entwicklern über Qualität und Ziele erreicht werden.

Das *Prototyping* zuletzt wird zwar nicht explizit in den Standards erwähnt, aber es kann unnötige Mehrarbeit reduzieren und die Validierung des Systems erleichtern. Im Tagesgeschäft wird dieses Vorgehen schon sehr oft eingesetzt. Eine „Formalisierung“ und offene Kommunikation kann hier helfen, eine gemeinsame Basis für die effizientere Entwicklung komplexer Systeme zu erreichen. Durch die Einbindung der sonst unabhängigen Qualitätssicherung in diese Methodik wird ein frühes und umfassendes System- und auch Problemverständnis geschaffen. Die Erkenntnisse experimenteller bzw. die Weiterverwendung von evolutionären Prototypen können hier frühzeitig in Verifikation und Validierung eingebunden werden, was die Qualität und die Sicherheit der Ergebnisse deutlich verbessert.

Insgesamt ist jede „Agile Methode“ dem Entwicklungsprozess förderlich, die die Kommunikation zwischen den Beteiligten steigert und verbessert. Komplexe Systeme können nur durch hochwertige und beständige Kommunikation verstanden und bewertet werden. Genau dies ist bei sicherheitskritischen Systemen der Fall.

7. FAZIT

Eines soll auf jeden Fall klargestellt werden: Es gibt keinen „heiligen Gral“ für die Entwicklung sicherheitskritischer Systeme und derartige Behauptungen sind stets Blendwerk. Sowohl der traditionelle als auch der agile Ansatz besitzen deutlich ihre Stärken und Schwächen. Somit ist es aus Sicht der Autoren vermessen zu sagen, dass sicherheitskritische Systeme nur mit den „altbewährten Methoden“ entwickelt werden können und dürfen. Die Stärken der neuen Methodiken und Denkweisen bieten ein Potential, auf das niemand verzichten kann. Die schnelllebige und komplexe Welt macht es immer wichtiger, sich auf das Wesentliche zu konzentrieren und flexibel auf neue Anforderungen und Aspekte eingehen zu können.

Das Ziel sollte sein, sich stetig im Projektverlauf als Menschen zu treffen und eine gemeinsame Sichtweise zu erarbeiten und zu wahren. Viele Vorgaben der Normen zur Entwicklung sicherheitskritischer System sind berechtigt, aber manche sind auch durch die rasante Weiterentwicklung überholt oder nutzlos. Es gilt also zu erarbeiten, welche Aspekte für ein konkretes Projekt oder ein konkretes Problem wichtig sind und sich auf diese zu konzentrieren. Die Ergebnisse sollten transparent entstehen. Dabei sollte aber das genaue Vorgehen irrelevant sein, solange es bekannt und akzeptiert ist.

Durch die angedachte Kombination der gängigen Normen mit der Agilität und gängigen Methoden aus agilen Vorgehensmodellen können Synergien geschaffen werden, die auch im sicherheitskritischen Bereich die Digitalisierung vorantreiben und etablieren können.

Abbildungsverzeichnis

Literatur

Beck, Kent: *Test Driven Development: By Example*, Addison Wesley, Boston, 2003

Deming, William E.: *Out of the Crisis*; Massachusetts Institute of Technology, Cambridge, 1982

Hofert, Svenja: *Das agile Mindset. Mitarbeiter entwickeln. Zukunft der Arbeit gestalten*, Springer Gabler, Wiesbaden, 2018

König, Wolfgang: *Der Gelehrte und der Manager: Franz Reuleaux (1829-1905) und Alois Riedler (1850-1936) in Technik, Wissenschaft und Gesellschaft (Pallas Athene, Band 49)*, Franz Steiner, Stuttgart, 2014

Kurtz, Cynthia F. & Snowden, David J.: *The New Dynamics of Strategy: Sense-making in a Complex-Complicated World*, In: IBM Systems Journal, Vol. 42, no. 3, S. 462–83, IBM Systems, Armonk, 2003

Martin, Robert C.: *Clean Agile: Back to Basics*, Pearson, Boston, 2019

Shewhart, Walter A.: *Statistical Method from the Viewpoint of Quality Control*, Dover, New York, 1986

Snowden, David J.: *Cynefin: a sense of time and space, the social ecology of knowledge management*; In: Knowledge Horizons: The Present and the Promise of Knowledge Management (Hrsg. Despres, Charles & Chauvel, Daniele), Butterworth-Heinemann, Boston, 2000

Stacey, Ralph D.: *Complexity and Creativity in Organizations*, Berrett-Koehler, San Francisco, 1996

Stacey, Ralph D.: *Strategic Management and Organisational Dynamics: The Challenge of Complexity*, 3rd Edition, Prentice Hall, New York, 2003

ECSS: *ECSS-E-ST-40C; Space engineering: Software*, ECSS Secretariat, Noordwijk, 2009

ECSS: *ECSS-Q-ST-80C (Rev. 1); Space product assurance: Software product assurance*, ECSS Secretariat, Noordwijk, 2017

Zimmerman, Brenda: *Ralph Stacey's Agreement & Certainty Matrix*, Schulich School of Business, York University, Toronto, 2001

Internetquellen

https://agilemanifesto.org/iso/de/manifesto.html

https://www.vuca-welt.de/

Kontakt zu den Autoren:

Dipl.-Ing. (BA) Stefan Wertheimer

Web: http://www.one-small-step.de
E-Mail: stefan@one-small-step.de

Dipl. Inf. (FH) Claudia Haußmann, MA

Web: http://www.butterflying.de
E-Mail: kontakt@butterflying.de