# ESE Kongress 2014

## Vortragsskript:

# Voll agil, oder doch nur halb

## Agile Werte, Prinzipien und Methoden in der traditionellen Entwicklung

Remo Markgraf, MicroConsult GmbH

**Embedded-Software entsteht in aktuellen Projekten zumeist „irgendwie“ agil. Größte Hürden für „richtig“ agil sind in erster Linie Sicherheitsanforderungen, vorhandene System-Prozesse nach V-Modell XT, ein schwer abschätzbares Risiko im Umstieg des Entwicklungsprozesses und vor allem dessen Einbindung**

Wer agil entwickeln will, aufgrund von Sicherheitsanforderungen aber nicht umsteigen kann oder den Sprung ins kalte Wasser noch scheut, kann durch die hier vorgestellten Denkanstöße zu einem iterativen traditionellen Prozess trotzdem agile Vorteile nutzen und Erfahrung in der agilen Denke sammeln.

### Agile Werte, Prinzipien und Methoden

Um tiefer in die Vorteile der agilen Bewegung einzutauchen, ist es zunächst unerlässlich, nicht nur agil zu handeln, sondern sich mit seiner Einstellung und Denkweise auf das agile Paradigma einzulassen.

*„Man entwickelt nicht agil, man ist agil.“*

Wer diesen Grundsatz beherzigt, wird aus allen vier agilen Werten des Manifests für agile Softwareentwicklung [1] auch in einer traditionellen Entwicklung Vorteile erzielen, übrigens unabhängig von der Domäne, also auch in der Hardware und im Gesamtsystem. Die vier agilen Werte lauten:


Abb. 1: Die vier agilen Werte des Manifests für agile Software-Entwicklung

Positive Auswirkungen der vier agilen Werte können auch im traditionellen Entwicklungsumfeld wie folgt erzielt werden:

**Individuen und Interaktionen mehr als Prozesse und Werkzeuge**

Es sind die Menschen, die innovative und komplexe Lösungen gestalten. Als Konsequenz sollte also der Menschen im Mittelpunkt von Entwicklungsprojekten stehen. Den Prozess statt die Anwender zu ändern ist erfolgversprechender und einfacher, solange eine strukturierte und geordnete Vorgehensweise gesichert ist. Der Entwicklungsprozess soll eine Linie vorgeben und die Arbeit strukturieren, aber nicht bevormunden und nicht nur als zusätzliche Last empfunden werden. Lassen Sie unnötige Prozessschritte einfach weg oder vereinfachen Sie diese, soweit es die Forderung nach Prozessqualität zulässt.

**Funktionierende Software mehr als umfassende Dokumentation**

Entgegen dem häufigen Irrglauben, Dokumentation sei in agilen Prozessen unwichtig, geht es darum, den Fokus auf das Wesentliche zu richten. So steht das eigentliche Produkt im Mittelpunkt. Damit das Produkt auch den nicht-funktionalen Ansprüchen wie z.B. Testbarkeit, Änderbarkeit und Wartbarkeit genügen kann, braucht es passende Dokumentation. Investieren Sie mehr Aufwand in die wichtigen Dokumente, die dann auch gepflegt sein sollten und immer die realisierte Produktlösung widerspiegeln, und lassen Sie als Ausgleich weniger wichtige Dokumente einfach weg. Oder legen Sie einfache Varianten an, um den formalen Prozessforderungen zu genügen.

**Zusammenarbeit mit dem Kunden mehr als Vertragsverhandlung**

Viele vertragliche Unstimmigkeiten beruhen nicht auf der vermeintlich unterschiedlichen Interpretation des Lasten-/Pflichtenheftes, sondern darin, dass der Auftraggeber während der Projektlaufzeit ein klareres Bild seiner Anforderungen entwickelt. Kundenzufriedenheit und -bindung steigern Sie nur, wenn der Kunde das erhält, was er wirklich braucht - unabhängig davon, was er ursprünglich gefordert hat. Aber wie bekommen Sie am besten ein qualifiziertes Feedback so rechtzeitig, dass Sie noch die Chance haben, das Projekt entsprechend auszusteuern? Das agile Entwicklungsframework Scrum sieht hierfür das Sprint Review Meeting und für interne Verbesserungen das Retrospective-Meeting vor. Die Grundideen beider Meetings lassen sich auch in der traditionellen Entwicklung in vergleichbarer Weise realisieren, indem Sie möglichst viele und kurze Feedbackschleifen einbauen. Im Abschnitt „Agile Stärken im traditionellen Gewand“ werden wir diese Themen genauer erörtern.

**Reagieren auf Veränderung mehr als das Befolgen eines Plans**

Die größte Stärke agiler Entwicklungsprozesse ist sicherlich die Änderungsfreundlichkeit und -geschwindigkeit. Hier stoßen traditionelle Prozesse sehr schnell an ihre Grenzen. Wer kennt nicht das folgende Szenario? …Die Planung steht, das Grob- und Feinkonzept ist erstellt, das Produkt ist eventuell gerade mitten in der Entwicklung - und nun wird eine Änderung erwartet. Dazu gibt es in traditionellen Prozessen ein Change-Request-Verfahren, das den Weg beschreibt, die erforderlichen Änderungen abzuschätzen und in die Entwicklung einzuschleusen. Ein Projektleiter wird i.d.R. viel Zeit mit diesen Aktivitäten verbringen, und nicht selten entstehen Projektverzögerungen durch die Vielzahl und Komplexität der Änderungsanforderungen. Der effizienteste Weg aus diesem Dilemma ist eine iterative Entwicklung inklusive kontinuierlicher Integration und zumindest teilweiser Testautomatisierung.

Schon in der Projektplanung wird die Entstehung des Produktes in mehreren aufeinander aufbauenden Stufen berücksichtigt. Jede Stufe durchläuft den Weg von der Feinplanung über Implementierung, Integration und Test, ähnlich einer Änderungsanforderung. Der hohe Aufwand für späte Änderungen lässt sich minimieren, indem man diese Änderungen in einer ohnehin vorgesehenen Stufe umsetzt. Die Frage, ob ein weniger wichtiges Feature dafür weggelassen werden kann, ist unabhängig vom Entwicklungsprozess eine unternehmerische Entscheidung.

**Einbettung im V-Modell**

Auf der einen Seite gibt es die agile Entwicklung mit kleinen inkrementellen Schritten, die jeweils auch erst einer nach dem anderen geplant und eventuell durch Test-Driven Development in winzigen Mikro-Schritten realisiert werden. Auf der anderen Seite durchläuft ein Produkt nach traditionellem Prozess immer als Ganzes die Prozessschritte. In technisch komplexen Projekten wird in der traditionellen Entwicklung das Prototyping eingesetzt, um Lösungswege auszuloten und höhere Planungssicherheit zu erhalten.

Nach der reinen Lehre müsste der Prototyp verworfen und das Produkt neu entwickelt werden. Doch jeder weiß, dass Prototypen die längste Lebensdauer haben. Die Folge ist meist eine schlechte Architektur, die ja nur für den Prototypen gedacht war und die Kosten für Erweiterungen und Wartbarkeit in die Höhe treibt.

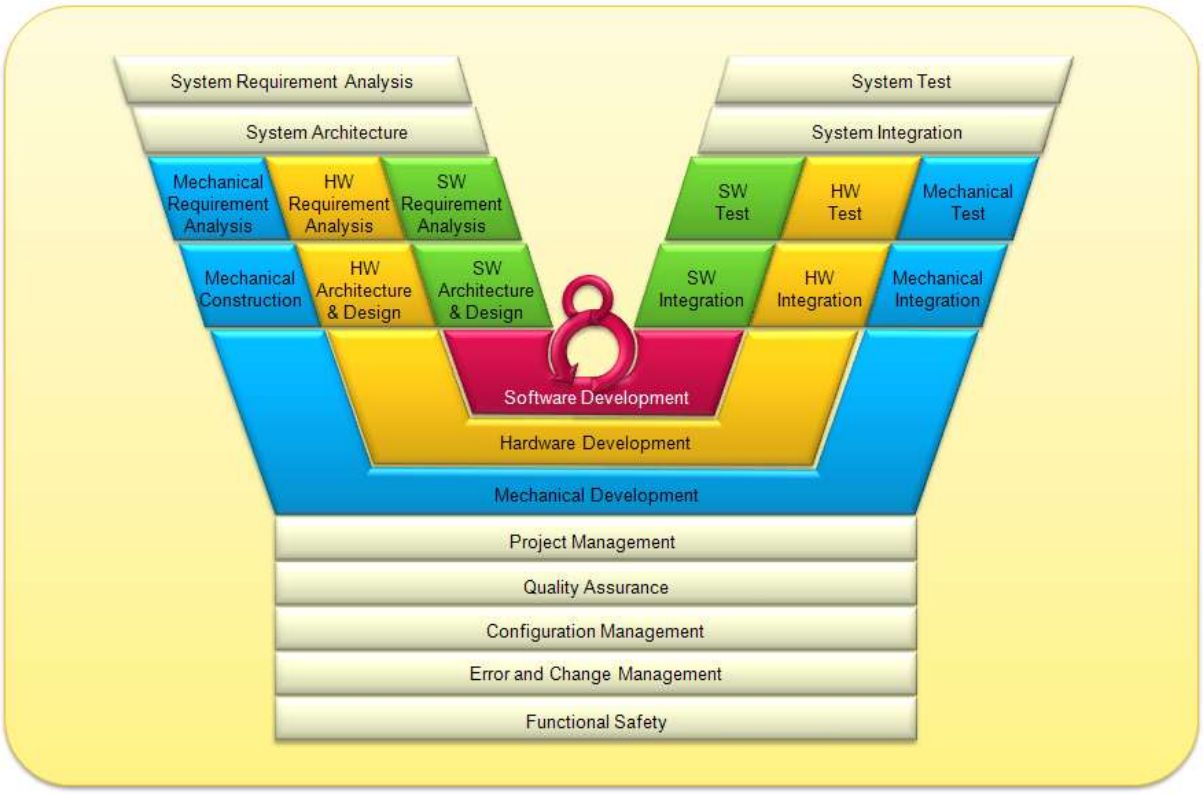


Abb. 2: Agiles Vorgehen in der Software-Implementierung im V-Modell

Warum also nicht die Not zur Tugend machen und das Produkt auch im V-Modell iterativ entstehen lassen? Das Zauberwort heißt Refactoring. Anstatt die Prototypen zu verwerfen oder schlechte Architektur zu akzeptieren, werden mit der Methode des Refactorings die Prototypen überarbeitet, bis sie als Funktionshübe mit sauberer Architektur das Produkt wachsen lassen. Diese Funktionshübe werden von Anfang an in der Projektplanung berücksichtigt.

Entsteht die Feinspezifikation nicht in einem großen Wurf, sondern ermöglicht man das iterative Wachsen mit den Funktionshüben, dann nimmt die Flexibilität weiter zu. Die Änderungskosten sinken und die Übereinstimmung der Dokumentation mit der realisierten Lösung steigt. Dazu sollte die Feinspezifikation in zwei Stufen entstehen. In der ersten Stufe wird das Design relativ oberflächlich beschrieben und legt den Grundstein für die Verfeinerungsschritte, die folgen. Die zweite Stufe ist in die erwähnten Funktionshübe gesplittet und sorgt so für eine iterative Vertiefung der Spezifikation. Vom Modell her legen Sie für jeden Funktionshub einen neuen Änderungsstand des Dokumentes an.

Damit Sie noch während der Entwicklung sehr wertvolles, qualitativ hochwertiges Feedback vom Auftraggeber /Anwender erhalten können, machen Sie sich den agilen Ansatz zunutze, auch die Integrations- und Testphasen des Prozesses mehrfach für die Funktionshübe zu durchlaufen.

Dafür brauchen Sie eine kontinuierliche Integration, einen automatisierten Unit-Test und zumindest teilautomatisierten System-Test. Sie erhalten analog zu agilen iterativen Prozessen ein potentiell lieferbares Produktinkrement nach jedem Funktionshub. In der Praxis wird aus Kosten- und Zeitgründen die Qualität des Produktinkrements durch geringere Testtiefe eingeschränkt sein. Es soll ja nur zur Produktdemo für das Feedback dienen. Zur Auslieferung wird dann auf gewohnte Weise traditionell das gesamte Testprogramm durchlaufen.

**Agile Stärken im traditionellen Gewand**

Scrum sieht mit dem Sprint Review Meeting eine Produktdemo am Ende eines jeden Sprints vor. Diese Möglichkeit lässt sich auf die iterative Entwicklung nach VModell übernehmen. Je früher sich eine Teilfunktionalität demonstrieren lässt, desto leichter lassen sich Änderungswünsche umsetzen. Eine gute Beziehung zum Kunden und damit die Möglichkeit zu Demos auf Entwicklungsboard oder bestenfalls Early-Prototyping-Hardware mit noch wenig integrierter Softwarefunktionalität zahlt sich besonders aus. Auf Änderungswünsche vorbereitet zu sein und diese relativ einfach in ihre Funktionshübe einfließen zu lassen sorgt für Zufriedenheit und starke Motivation der Beteiligten.

In großen Organisationen gibt es in der traditionellen Entwicklung Lessons-Learned Meetings, die ein- bis zweimal pro Jahr, manchmal auch beim Erreichen wichtiger Meilensteine eines Entwicklungsprojektes, abgehalten werden. Sie verfolgen das gleiche Ziel wie die Retrospective-Meetings in Scrum: Sie dienen letztlich der Steigerung der Produktivität, indem Antworten gesucht werden auf die Fragen „Was lief gut, was schlecht, was hat uns davon abgehalten, optimale Ergebnisse abzuliefern?“ In der agilen Entwicklung werden diese Fragen konsequent am Ende jedes Sprints erörtert, um schnelles Feedback zu erhalten.

Die Bereitschaft zu kontinuierlicher Verbesserung ist völlig unabhängig vom verwendeten Entwicklungsprozess. In Anlehnung an Scrum sollten Feedbackrunden so häufig als möglich durchgeführt werden. Das Ende eines jeden oben beschriebenen iterativen Funktionshubes bietet sich hierfür sehr gut an. Für den Erfolg dieser Runden ist es ausschlaggebend, ein Umfeld zu schaffen, dass durch den Willen geprägt ist zu verändern, auch wenn es unbequem ist. Jeder einzelne muss dazu aus seiner persönlichen Komfortzone heraustreten und Änderungen eine Chance geben.

**Fazit**

Die Grenze zwischen traditioneller und agiler Entwicklung ist fließend. Aus allen agilen Werten lassen sich auch Vorteile in der traditionellen Entwicklung erzielen. Warum sollten Sie dann eigentlich noch auf eine agile Vorgehensweise umstellen?

Agile Entwicklung ist kein in Stein gemeißeltes Gesetz. Der Wille und die Motivation, Dinge besser zu machen, sorgen für stetigen Wandel. Diese kontinuierliche Veränderung kann zu Orientierungsmangel führen. Sich an existierenden Frameworks wie z.B. Scrum zu orientieren gibt Halt und Sicherheit. So ist z.B. die Einführung selbstorganisierender Teams und deren Zusammensetzung nach Kundenmehrwert in Scrum einfacher umzusetzen als in einem angepassten VModell, in dem eine Rollenverteilung in den Prozessschritten vorgesehen ist.

In Summe muss der hier aufgezeigte iterative traditionelle Prozess gut an das existierende Umfeld angepasst werden, um unnötige Reibungsverluste zu vermeiden. Der hier beschriebene Ansatz zeigt Ihnen einen Weg auf, die Kundenzufriedenheit sowie die Motivation der Beteiligten kontinuierlich zu steigern. Auf Wunsch werden wir Sie gerne auf dieser Reise mit agilen Schulungen, Prozess-Workshops und Projekt-Consulting begleiten.

## Literatur- und Quellenverzeichnis

[1] http://agilemanifesto.org
[2] http://www.microconsult.de/um/agile-dev
[3] http://www.microconsult.de/um/agile-tdd
[4] http://www.microconsult.de/um/scrum

## Autor

Remo Markgraf ist Senior Management Consultant bei der MicroConsult GmbH. Neben Begeisterung für Innovation und Leidenschaft für Embedded-Systeme verfügt er über langjährige Projekt- und internationale Führungserfahrung in Softwareentwicklung, Systems Engineering, Projekt-, Produkt-, Innovations- und Business Development Management sowie dem technischen Vertrieb.