



Agile Softwareentwicklung im Projekt CafE

BMW IT-Messe
München, 17./18. Oktober 2007,
Dr. Christian Süß, BMW
Jürgen Dinsing, ACTANO



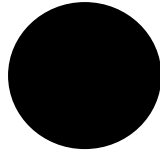


Was ist agile Softwareentwicklung?



Ziel der Softwareentwicklung

10.10.2007 - 3



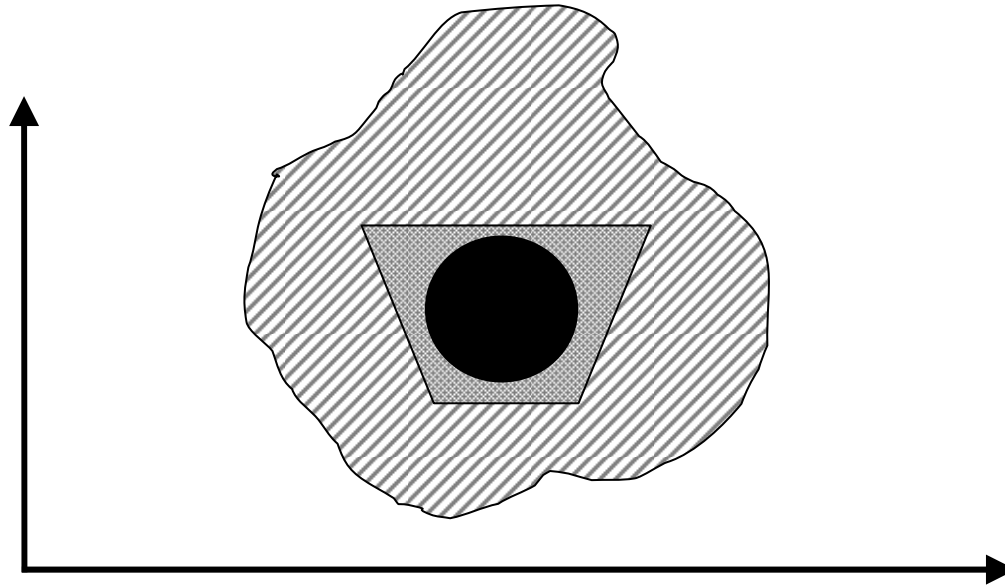
- Ziel der Softwareentwicklung ist, dem User eine nutzenbringende Anwendung zur Verfügung zu stellen:
 - Diese Anwendung muss die Funktionen bereitstellen, die der Anwender benötigt.
→ **Korrektheit**
 - Diese Anwendung muss dem Anwender in ausreichendem Umfang zur Verfügung stehen.
→ **Stabilität**
 - Nutzen bringt vornehmlich die Anwendung.
 - Der Nutzen von Fachkonzepten, IT-Konzepten, Dokumentation, etc. ist nur für spezielle Aufgaben vorhanden.
 - Das Ergebnis zählt, nicht der Weg dorthin.

Agilität



Unsicherheit der Softwareentwicklung

10.10.2007 - 4



Agilität

- **Zu Beginn eines Projektes ist die Unsicherheit über das Ziel groß!**
 - **Weder die Gestalt des Ziels noch die Lage des Ziels sind wirklich bekannt.**

Risiken, Unsicherheit, ...

10.10.2007 - 5



- **Der Kunde weiß nicht genau, was er wann braucht.**
- **Nicht alle Anforderungen und Stakeholder sind bekannt.**
- **Der Kunde hat widersprüchliche Anforderungen.**
- **Der Auftragnehmer versteht nicht genau, was der Kunde will.**
- **Es treten Änderungen in den Prioritäten, Geschäftsprozessen, etc. während des Projektes auf.**
- **Der Auftragnehmer unterschätzt/überschätzt den Aufwand.**
- **Das Projekt ist in eine komplexe Projektlandschaft eingebunden.**
- **Es gibt technische Risiken, z.B. die Infrastruktur hält nicht was sie verspricht.**

Agilität



Das herkömmliche Vorgehen in Projekten

10.10.2007 - 6



- **Der Lösungsansatz des klassischen Software Engineering ist am Anfang präzise zu spezifizieren.**
 - Erst fachlich, dann technisch.
 - Das hilft nicht, weil es lediglich das genauer aufschreibt, was man nicht weiß.
- **Es gibt vier grundsätzliche Faktoren mit denen Problemen in Projekten entgegnet werden kann:**
 - Time – man kann sich mehr Zeit nehmen (und damit mehr Budget).
 - Budget – man kann mehr Budget aufwenden.
 - Qualität – man kann eine geringer Qualität in Kauf nehmen.
 - Umfang der Funktionalität – man kann weniger als geplant realisieren.
- **Häufig wird – eher unbewusst – die Qualität vernachlässigt.**
 - Bewusst entscheidet man sich eher für die Minderung des Funktionsumfangs.
- **Einer dieser vier Faktoren wird aber sehr häufig verändert.**

Agilität



Die Lösung – Agile Softwareentwicklung

10.10.2007 - 7



- **Die Lösung ist die Agile Softwareentwicklung:**
 - Agile Softwareentwicklung stellt das Ergebnis, die anwendbare Anwendung, in den Mittelpunkt.
 - Dafür werden wenige flexible Regelungen aufgestellt, die den bürokratischen Overhead minimieren.
 - Basis ist eine zyklische Entwicklung, die die Veränderung der Ziele ermöglicht.

- **Die grundlegenden Ideen sind:**
 - kurze Releasezyklen mit verwendbarem Ergebnis
 - evolutionäre Entwicklung, einfaches Design
 - Kommunikation im Team mit dem Kunden im Team und räumlicher Nähe
 - Verantwortung des Einzelnen, Qualitätsarbeit leisten
 - Praktiken, Kultur und Adaptivität statt feste Prozesse
 - Automatisierte Tests als zentraler Bestandteil der Entwicklung
 - Dokumentation soweit nötig
 - Vertrauen und Werte
 - Änderungen sind Normalität
 - Refactoring

Agilität

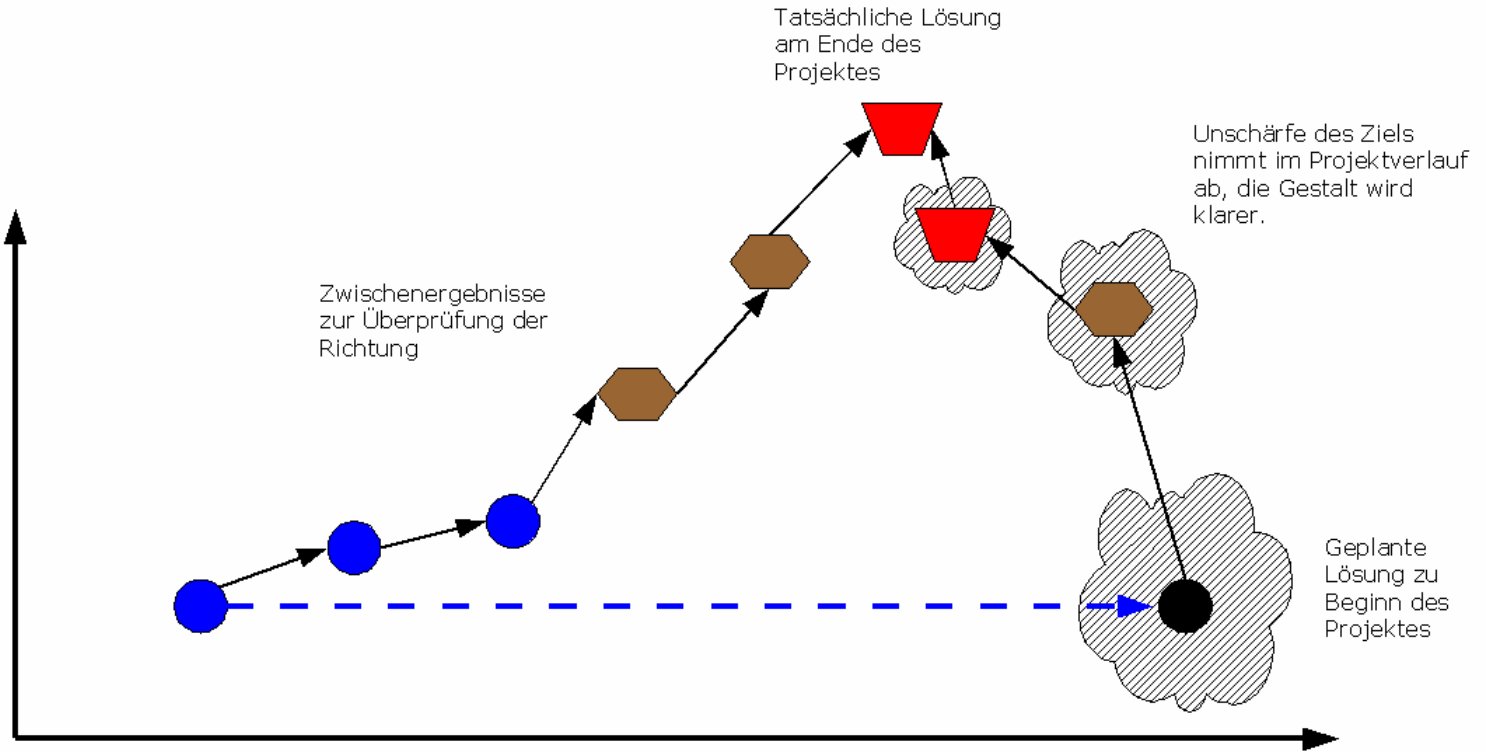


Moving Target

10.10.2007 - 8



Agilität



Idee der Grafik aus verschiedenen Quellen, ursprünglich wohl K. Eberhardt, Fa. Iteratec, aber auch bei Starke und Beneken



Agiles Manifest

10.10.2007 - 9



Agilität

Wir entdecken bessere Wege zur Entwicklung von Software, indem wir Software entwickeln und anderen bei der Entwicklung helfen. Durch diese Tätigkeiten haben wir gelernt, dass uns

- Individuen und Interaktion **wichtiger** sind als Prozesse und Werkzeuge.
- Funktionierende Software **wichtiger** ist als umfangreiche Dokumentation.
- Kooperation mit Projektbetroffenen **wichtiger** ist als Vertragsverhandlungen.
- Reaktion auf Änderungen **wichtiger** ist als Festhalten an einem starren Plan.

Wichtiger heißt nicht, dass das Andere **unwichtig** ist!

17 unabhängige Berater in den USA, Februar 2001 in Utah, siehe <http://www.agilemanifesto.org/>



Voraussetzungen

10.10.2007 - 10



Agilität

- **Verantwortung erfordert Qualifizierung (nur qualifizierte Leute „wissen“, was zu tun ist)**
 - Können, Wissen, Disziplin, ...
- **mit kleinen Teams ist es leichter**
 - auch aufgrund der Kommunikation
- **kurze Releases um so besser, je mehr**
 - Akzeptanz und Zeit in der Fachabteilung vorhanden ist
 - die Anforderungen in Partitionen unterteilt werden können
- **Evolutionäre Architektur, nur wenn Änderungen billig**
 - „Gute“ Start-Architektur (Erfahrener Architekt)
 - konsequenter Einsatz automatischer Tests (Test-Driven Development)
 - Refactoring - Entwicklungsumgebung macht Änderungen leicht, z.B. mit Eclipse (COBOL-Refactoring auf dem Host?)
- **Die richtige Dokumentation, Unwichtiges weglassen**
 - Für die Wartung aber muss wegen des Know-How-Transfers Dokumentation vorhanden sein
 - Anforderungen in geeigneter Tiefe/Vollständigkeit beschreiben
 - Also die nötige Dokumentation während der Entwicklung erstellen – nicht vorher
- **Akzeptanz bei Management, Einkauf, ...**
 - Wiederholbarkeit der Ergebnisse nicht über Prozess gewährleistet
 - Planbarkeit, Verträge bei änderbarem Prozess?
 - Abrechnung von Ergebnissen?



Was ist CafE?



Vielzahl von Steuergerätevarianten

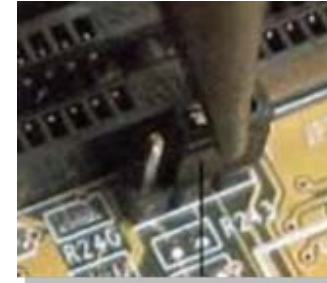
10.10.2007 - 12



Produktion



Z.B.: 308 Fahrzeugauftragselemente,
über 2 Mio. theoretische Varianten



Entwicklung



Z.B.: 3.500 Funktionen in 64
Steuergeräten

- Steuergeräte realisieren eine hohe Zahl an Funktionen für unterschiedlichste Fahrzeuge.
- Dadurch entsteht eine sehr große Vielzahl von Steuergerätevarianten, die zu beherrschen ist.

Minimierung der Varianten durch Codierung

- Die Codierung ermöglicht die Verwendung einer Hardware- bzw. Software-Variante eines Steuergeräts statt vieler Varianten.
- Die Steuergeräte sind für alle Funktionen parametrisierbar ausgelegt und werden per Codierung für jedes einzelne Fahrzeug funktionstüchtig gemacht.

10.10.2007 - 13



CafE



Spezifikation der Codierung als CAF mit CafE

- **Spezifikation der Codierbarkeit eines Steuergeräts als CAF**
 - Coding Application File (CAF)
 - XML-basiertes Standardaustauschformat
 - Modulare Entwicklungsmöglichkeit
 - Leichte, lokale Änderung und Pflege
 - Lokale Absicherbarkeit
 - Schutz der Vertraulichkeit durch Verschlüsselung
 - Schutz der Integrität durch Signatur
- **Erstellung und Bearbeitung von CAFs mit CafE**
 - CAF-Editor (CafE)
 - Effizientes und sicheres Erstellen von Codierspezifikationen
 - Ausgefeiltes Regelwerk für korrektes Editieren
 - Editor für Variantenbedingungen
 - Zugriffsschutz durch persönliches Entwickler-Softtoken
 - Weltweite Verwendung bei Steuergeräteherstellern
 - BMW-interne Verwendung bei Absicherungsstellen

10.10.2007 - 14



CafE





Agilität im Projekt CafE?

Auswahl der richtigen Methode

10.10.2007 - 16



- **Es gibt verschiedene agile Methoden**
 - Extreme Programming, Die Crystal Familie, Scrum, Lean Development, DSDM (Dynamic Software Development Method), ASD (Adaptive Software Development)

- **Es gibt keine richtige Methode – es gibt nur die angemessene Methode**
 - Angemessen dem Projekt – also der Größe, der Kritikalität und weiteren Randbedingungen

- **Daher muss die richtige Methode zusammengestellt werden**
 - Einzelteile der verschiedenen Methoden werden in einen sinnvollen Prozess gefasst.
 - Dazu eignen sich Einzelteile aus einigen Methoden mehr andere weniger.

- **Wichtig ist der Metaprozess**
 - Regelmäßige Retrospektiven und daraus resultierende Anpassung der Arbeitsweise sind in agilen Verfahren unabdingbar.

Agilität und CafE

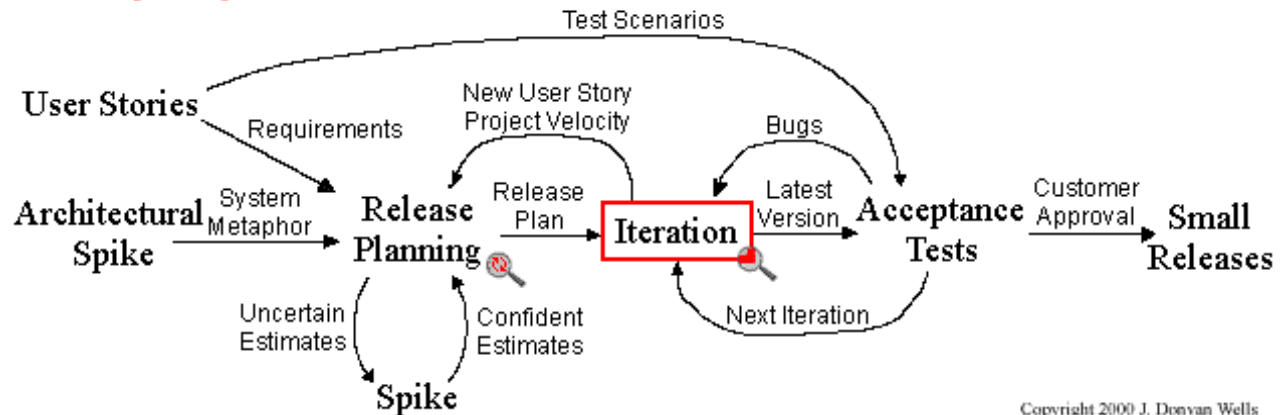


Ursprung unseres Vorgehen

- Das unten aufgezeigte Vorgehen ist ein Ursprung unseres Prozesses.
 - Aufgrund der Ausgangssituation und der gegebenen Randbedingungen wurde das Modell angepasst.



Extreme Programming Project

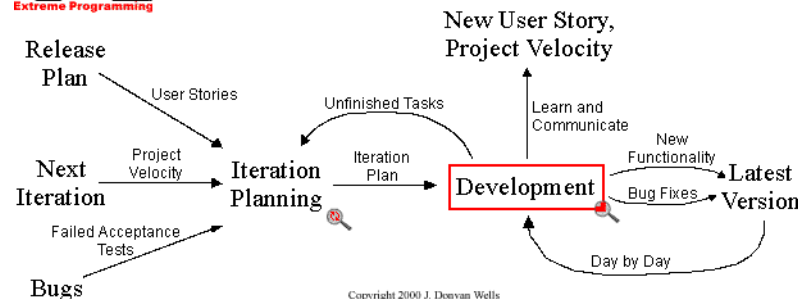


Copyright 2000 J. Donovan Wells



Iteration

Zoom Out



Copyright 2000 J. Donovan Wells

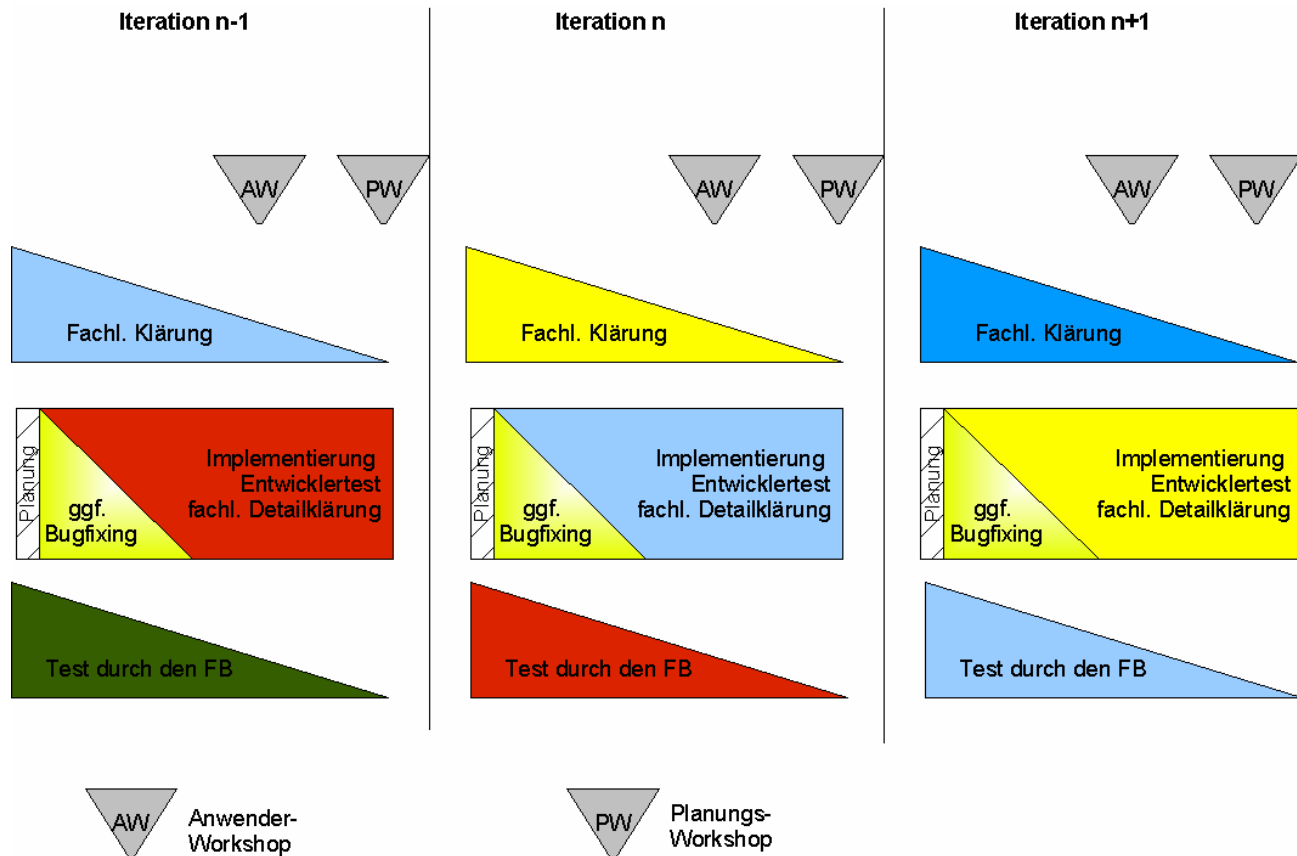


Das Vorgehen im Projekt CafE

10.10.2007 - 18



- In vierwöchigen Iterationen werden die Anforderungen implementiert.
- In zwei Workshops vor jeder Iteration werden die jeweiligen Anforderungen für eine Iteration festgelegt.
- Der Fachbereich testet die Ergebnisse der vorherigen Iteration.



Agilität und CafE



Details des Vorgehens im Projekt CafE

10.10.2007 - 19



● Randbedingungen

- Am Ende einer Iteration steht lauffähige Software zur Verfügung.
- Die Entwicklung der Funktionen erfolgt zeitnah und so kann entwickelt werden, was der Kunde wirklich benötigt.
- Die Entwicklung erfolgt meist vor Ort beim Kunden in unmittelbarer Nähe zum Fachbereich.

● Artefakte im CafE

- Es werden die Konzepte aus der Leistungsstufe I so weit nötig fortgeschrieben.
- Die Releases werden mit Release-Notes verteilt.
- Benutzerdokumentation entsteht am Ende einer Leistungsstufe.

Agilität und CafE



Bisherige Erfahrungen im Projekt CafE

10.10.2007 - 20



- **Vorgehen für Kunden und Entwicklungsteam ungewohnt:**
 - Enge Zusammenarbeit schon während der Implementierung.
 - Beschreibung der Testfälle parallel zur Implementierung.
 - Entscheidungsfindung auf den Workshops musste geübt werden.
- **Hohe Flexibilität gegenüber sich ändernden Anforderungen.**
 - Zu den ursprünglich angedachten großen Arbeitspaketen kamen eine Vielzahl kleiner Anforderungen (Bugfixes und Verbesserungen aus den ausgerollten Releases)
 - Es wurden 75% der großen Arbeitspakete umgesetzt und zusätzlich 85% der kleinen Anforderungen.
- **Kunde bekommt schneller Feedback über die umgesetzten Anforderungen.**
- **Es war möglich auf sehr stark veränderte inhaltliche und zeitliche Vorgaben zu reagieren.**
 - Es war einfach möglich auf eine Verschiebung eines wichtigen Termins in der Fahrzeugentwicklung zu reagieren.

Agilität und CafE





Ausblick



Ausblick - CafE wird weiter entwickelt

10.10.2007 - 22



- **Umstellung auf die Eclipse Rich Client Plattform:**
 - Als Basis einer Integrationsplattform ist RCP gewählt.
 - Dafür wird CafE zur Zeit in diese Umgebung migriert.
 - Ohne agiles Vorgehen wäre das sicherlich sehr schwierig geworden.
- **Änderungen am Vorgehen:**
 - Nur noch ein Workshop vor jeder Iteration zur Minimierung des Overhead.
 - Ggf. Verkürzung der Iteration auf drei Wochen.

Ausblick



Zusammenfassung

10.10.2007 - 23



- **Agile Softwareentwicklung ist ein probates Mittel, um**
 - die Unsicherheiten und Risiken im Laufe eines Softwareprojektes in den Griff zu bekommen,
 - sich ändernde Ziele und Randbedingungen zu beherrschen und
 - effizient Software zu entwickeln.
- **Dabei steht der Nutzen und die Qualität der Anwendung im Vordergrund.**
 - Es wird früh und intensiv Feedback eingeholt.
 - Die frühe und intensive QS helfen eine hohes Maß an Wartbarkeit und Erweiterbarkeit zu erreichen.

Ausblick

