

Das Agile Development Framework im Überblick

Golo Roden

www.goloroden.de

Über mich

- > Wissensvermittler und Technologieberater
 - > .NET, Codequalität und agile Methoden
 - > MVP für C# in den Jahren 2010 und 2011
 - > Agile Development Framework (ADF)
- > Autor, Sprecher und Trainer
 - > dotnetpro, heise Developer
 - > prio.conference, .NET DevCon
- > Kontakt
 - > www.goloroden.de



Eine Herausforderung – keine Kunst?

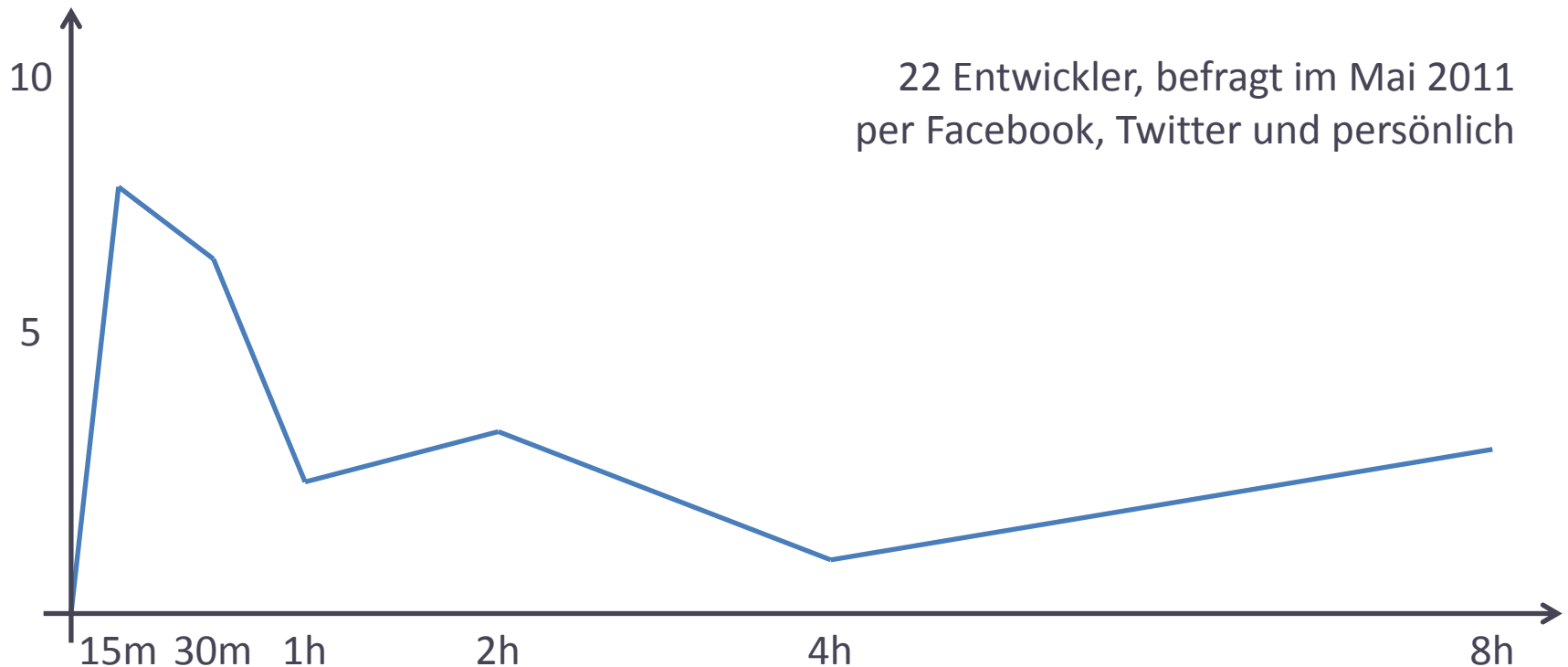
- > Herausforderung
 - > Blasen Sie ein rohes Ei aus
 - > Bemalen Sie es mit einer Wiese, einem Osterhasen, blauem Himmel und ein paar Wolken
 - > Binden Sie das Garn an das Streichholz und fädeln Sie es ein
- > Voraussetzungen
 - > Alle Materialien sind gegeben
- > Wie lange brauchen Sie?



(Quelle: <http://www.pitopia.de>)

Keine Kunst – wirklich?


- > Eine Umfrage unter Entwicklern hat Schwankungen zwischen 15 Minuten und 8 Stunden ergeben



- > Das entspricht einer Varianz von 4.800 % (!)

Verwunderlich?

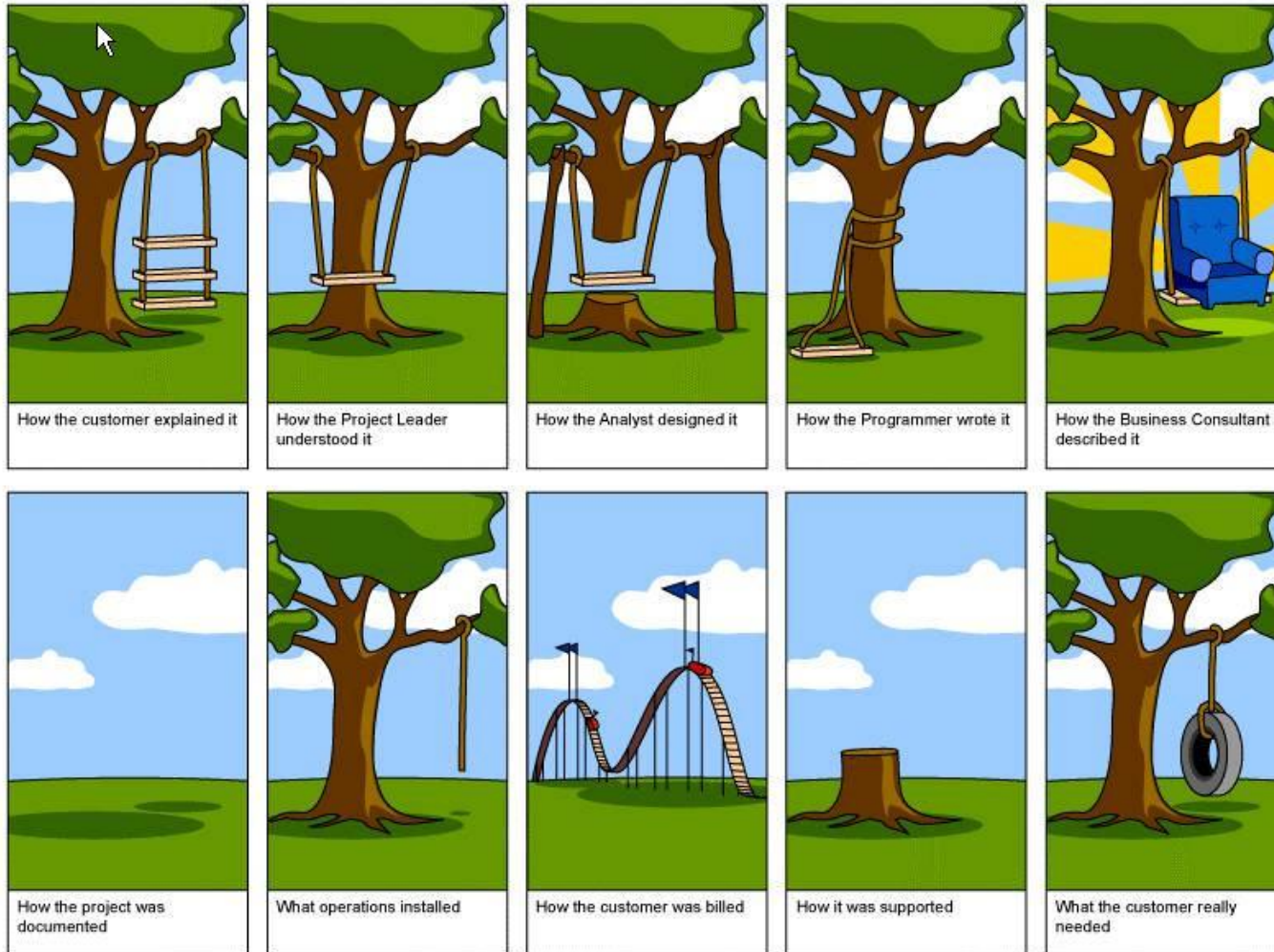
- > Herausforderung 2.0
 - > Welche Codezeile werden Sie am 17. Februar 2015 um 9:37 Uhr schreiben?
- > Unmöglich?
 - > Der Kunde will es aber wissen ...
- > A propos „wollen“ ...
 - > Was will der Kunde überhaupt?



```
1  /**-
2  sample javascript from xui
3  */
4
5  var undefined,
6      xui,
7      window    = this,
8      string     = new String('string'),
9      document   = window.document,
10     simpleExpr = /^#?([\w-]+)$/,
11     idExpr     = /^#/,
12     tagExpr    = /<([\w:]+)/,
13     slice     = function (e) { return [].slice.call(e, 0); };
14     try { var a = slice(document.documentElement.childNodes)[0].nodeType; }
15     catch(e){ slice = function (e) { var ret=[]; for (var i=0; e[i]; i++)
16         ret.push(e[i]); return ret; }; }
17
18 window.x$ = window.xui = xui = function(q, context) {
19     return new xui.fn.find(q, context);
20 };
```

(Quelle: <http://ethanschoonover.com/solarized>)

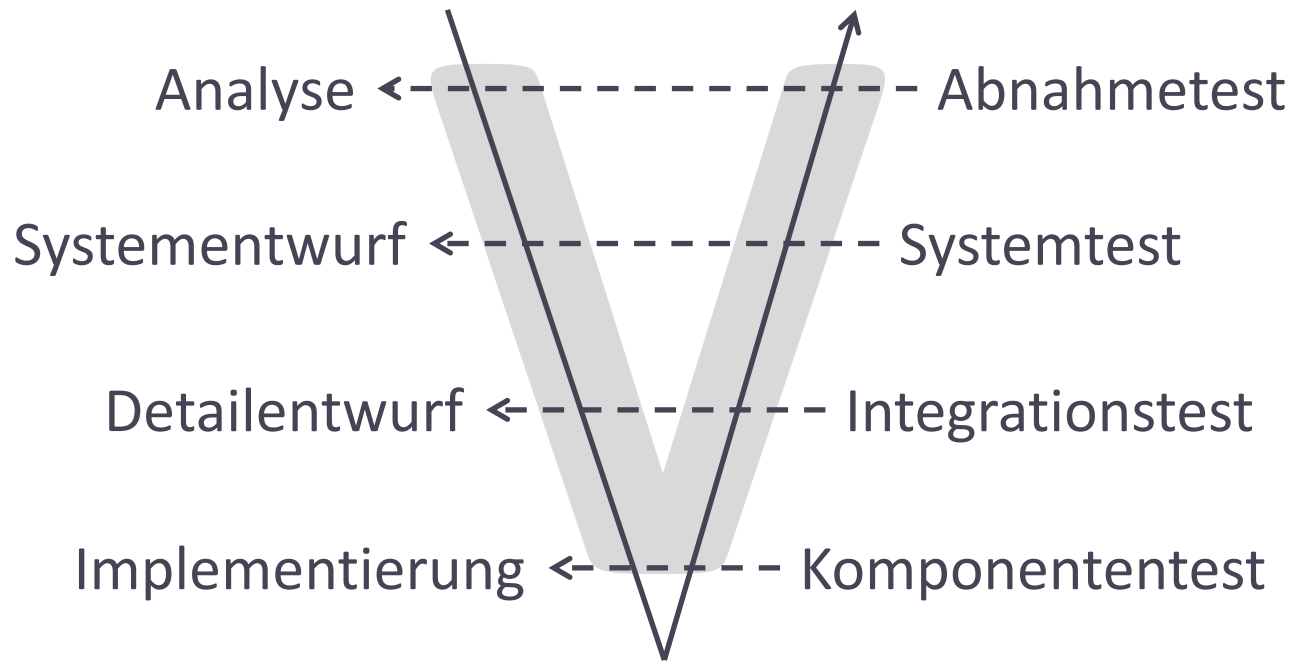
Was der Kunde will ...



(Quelle: <http://www.d80.co.uk/?tag=/NFR>)

Professionelle Entwicklung – wie?

> Das V-Modell



Das V-Modell – professionell?



(Quelle: <http://www.sportberg-goldeck.at/erlebnis-am-berg/geocaching-gps-schatzsuche-am-goldeck/>)

Big design up-front? *

* Scott Belware: The Problem with Big Design Up Front is the „Big“, not the „Up Front“
(siehe <http://www.dotnetpro.de/SL1103ADF1>)

Planen + Testen = Erfolg?



(Quelle: http://www.welt.de/motor/article1280688/Mercedes_und_der_Elch_Die_perfekte_Blamage.html)

Wenn nicht das V-Modell – wie dann?



(Quelle: <http://dilbert.com/strips/comic/2007-11-26/>)

Agilität und Anarchie – identisch?

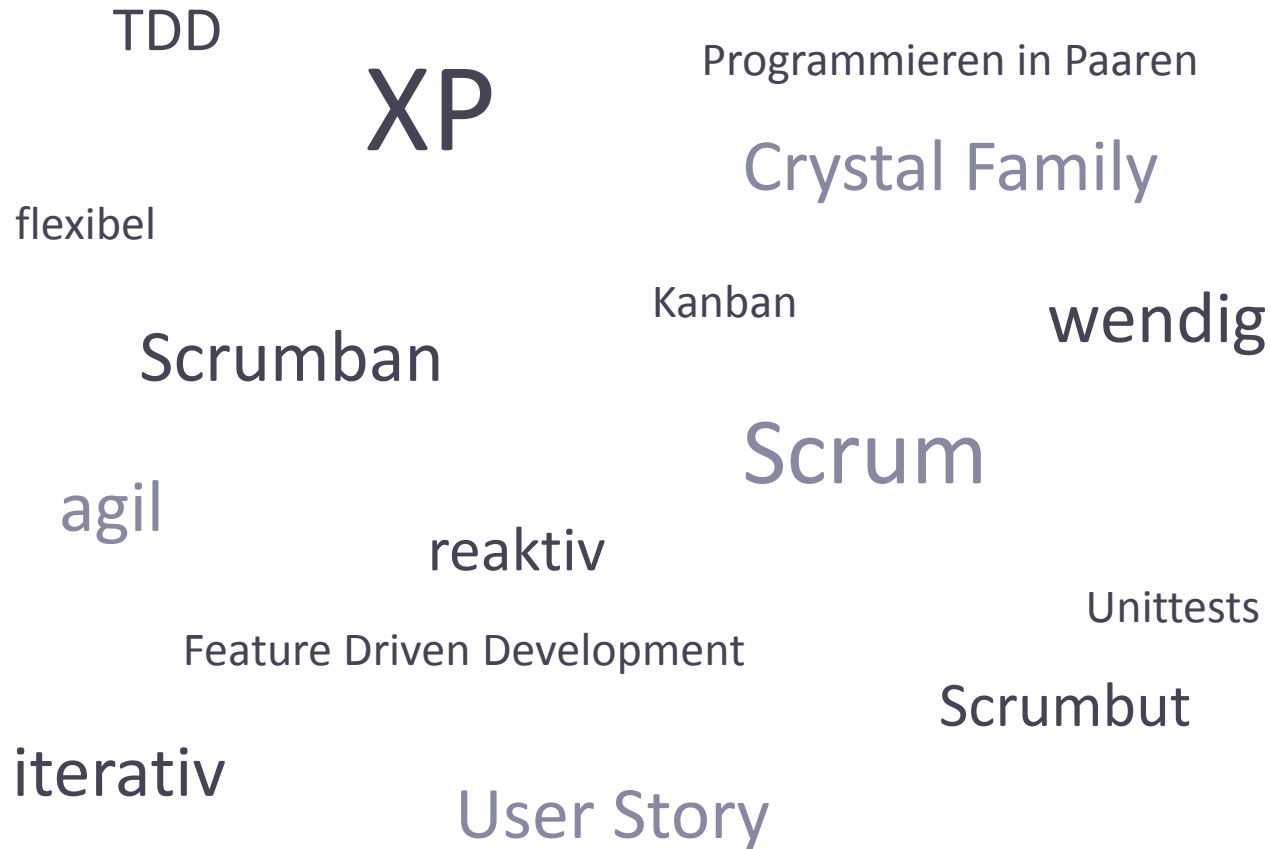
> Agile Manifesto

- > We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:
 - > Individuals and interactions over processes and tools
 - > Working software over comprehensive documentation
 - > Customer collaboration over contract negotiation
 - > Responding to change over following a plan
- > That is, while there is value in the items on the right, we value the items on the left more.

> Siehe

- > <http://www.agilemanifesto.org>

Von A wie „agil“ bis X wie „XP“ ...



Agile Landschaft – zersplittert?



(Quelle: <http://www.sueddeutsche.de/muenchen/muenchen/pavillon-auf-dem-marstallplatz-zersplittert-in-ewigkeit-1.959470>)

Agil 2.0 – quo vadis?

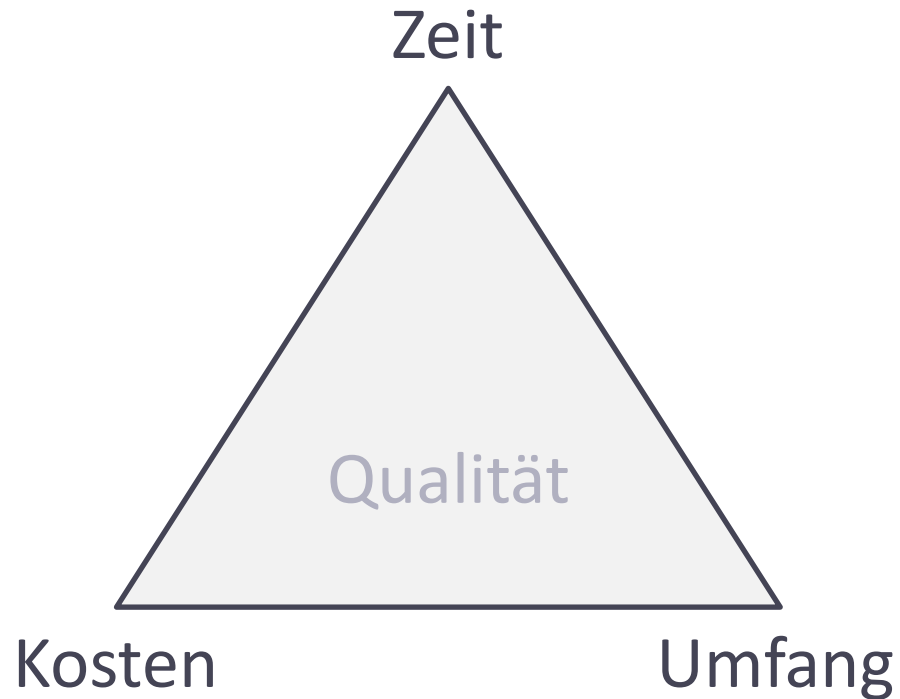
- > Das *Agile Development Framework* vereint die bestehenden agilen Methoden auf konsistente Art, passt fragliche Praktiken an oder eliminiert diese gänzlich, und ergänzt bislang fehlende Praktiken.

Quo vadis – warten auf Godot?

- > „Wer vom Ziel nicht weiß
kann den Weg nicht haben,
wird im selben Kreis
all sein Leben traben.“

(von Christian Morgenstern)

Ein Projekt – verschiedene Ziele?



Alles ist diskutabel – auch die Qualität?

- > „... zur flexiblen, konstruktiven und evolutionären Entwicklung qualitativ hochwertiger Software.“

(aus der Vision von ADF, siehe <http://www.agile-development-framework.net/>)

Als Basis – vier Maximen?

Evolution



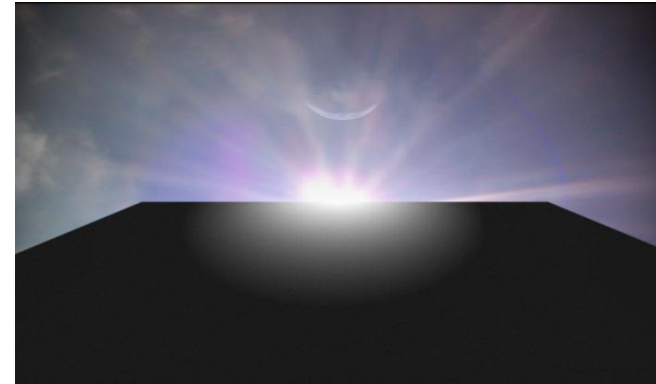
(Quelle: <http://www.kayakquixotica.com/2008/09/12/operetta/>)

Kommunikation



(Quelle: <http://www.scene-stealers.com/blogs/1-year-100-movies-15-2001-a-space-odyssey-1968/>)

Eleganz



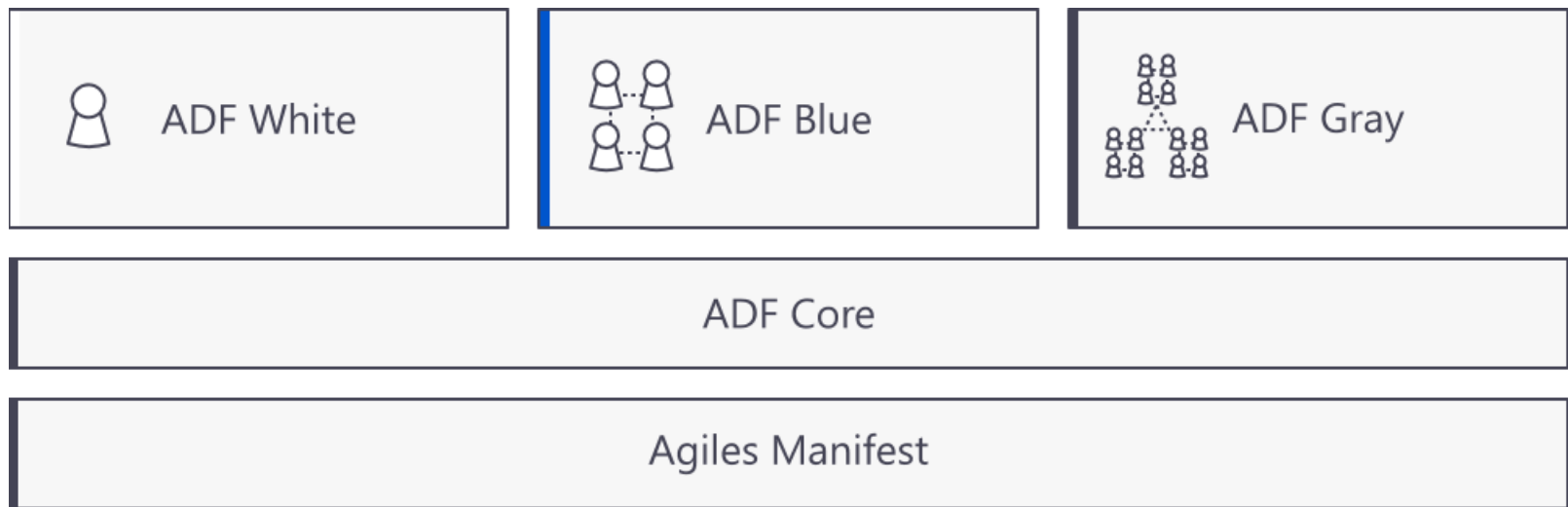
(Quelle: <http://libbyteal10.blogspot.com/2010/08/monolithic.html>)

Vertrauen



(Quelle: <http://vigilantgrandpa.blogspot.com/2010/07/hal-9000-to-handle-your-retirement.html>)

Drei Farben – weiß, blau, grau?



ADF White

 ADF White

> ADF White – für Individuen

Freiraum – vergeudete Zeit?

- > Wissensmanagement
 - > Wissen muss erworben, ausgebaut und gepflegt werden
 - > Kreativität und Ideen müssen sich entfalten können
- > Google als Vorbild
 - > 20% Innovation
Time-Off
- > Beispielsweise
 - > Jeden Freitag oder
täglich 90 Minuten
 - > Allein oder im Paar



(Quelle: <http://www.flickr.com/photos/yermom/3995203341/>)

Zeit messen – oder Ergebnisse?

- > Arbeitsweise
 - > Arbeit in Zeit messen – ein Relikt des Industriezeitalters
 - > Results-Only Work Environment (ROWE)
- > Verstreute Teams
 - > Koinzidenz verringern, Kompetenz erhöhen
 - > Wohlfühlen in der Arbeitsatmosphäre
- > Beispielsweise
 - > 100% Offsite
 - > Zusammenarbeit per TeamViewer und Skype



(Quelle: <http://auszeit-lanzarote.de/internet-und-arbeiten-im-urlaub.html>)

Null-Fehler-Richtlinie – null?!?

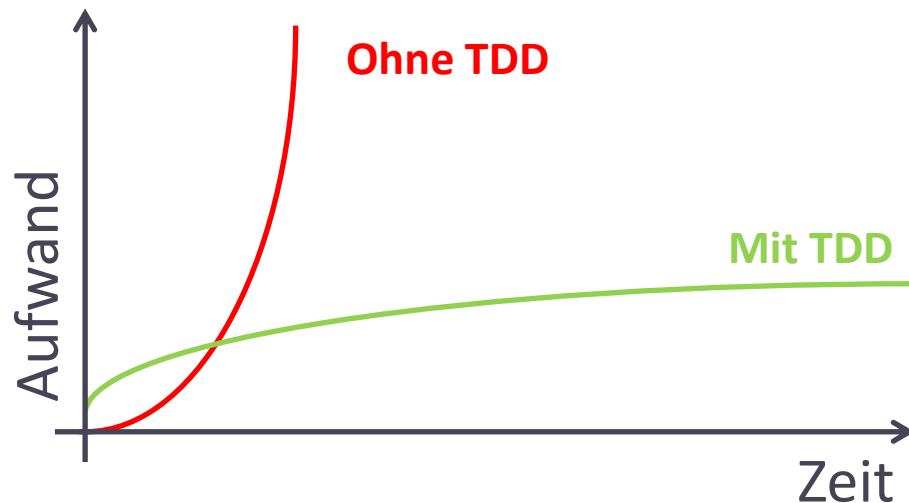
- > Qualitätssicherung
 - > Fehler werden desto teurer, je später sie behoben werden
 - > Pendant zur „Broken Window“-Theorie
- > Fehler beheben
 - > „Vor langer, langer Zeit – da war es nur eins.“
- > Beispielsweise
 - > Bewährte Vorgehensweisen verletzt
 - > Pixelfehler in der UI
 - > Schreibfeeler



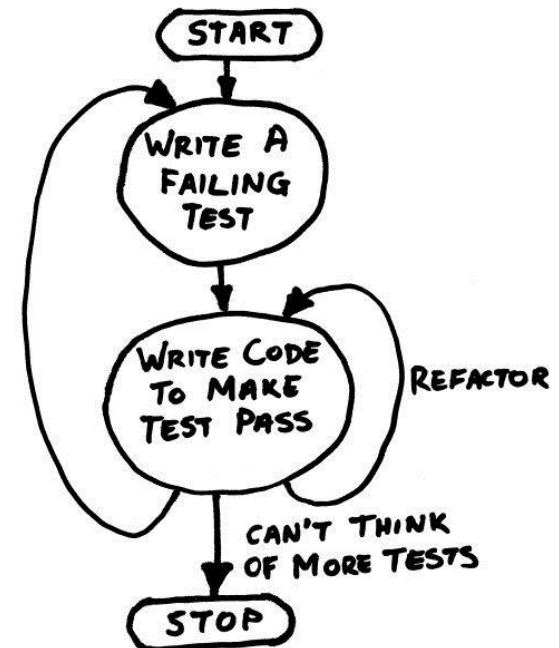
(Quelle: <http://www.skalduggery.com/2010/01/15/friends%E2%80%A6-on-crime-part-3/>)

Testen – aber wann?

- > Code
 - > Testgetriebene Entwicklung (TDD)
 - > 100% Testabdeckung



- > Beispielsweise
 - > Tests erzeugen Architektur
 - > Fehler erzeugen Tests



(Quelle: <http://test.ical.ly/2010/03/31/testgetriebene-entwicklung-tdd-einer-funktion-meines-symfony-plugins/>)

Fertig – wirklich?

- > Artefakte & Werkzeuge
 - > „Definition of Done“
- > Auch Features sind Projekte
 - > Es gilt – Zeit, Kosten oder Umfang?
 - > „It’s done when it’s done.“
- > Beispielsweise
 - > Paarweise entwickelt
 - > Testgetrieben entwickelt
 - > Refaktoriert und rearrangiert



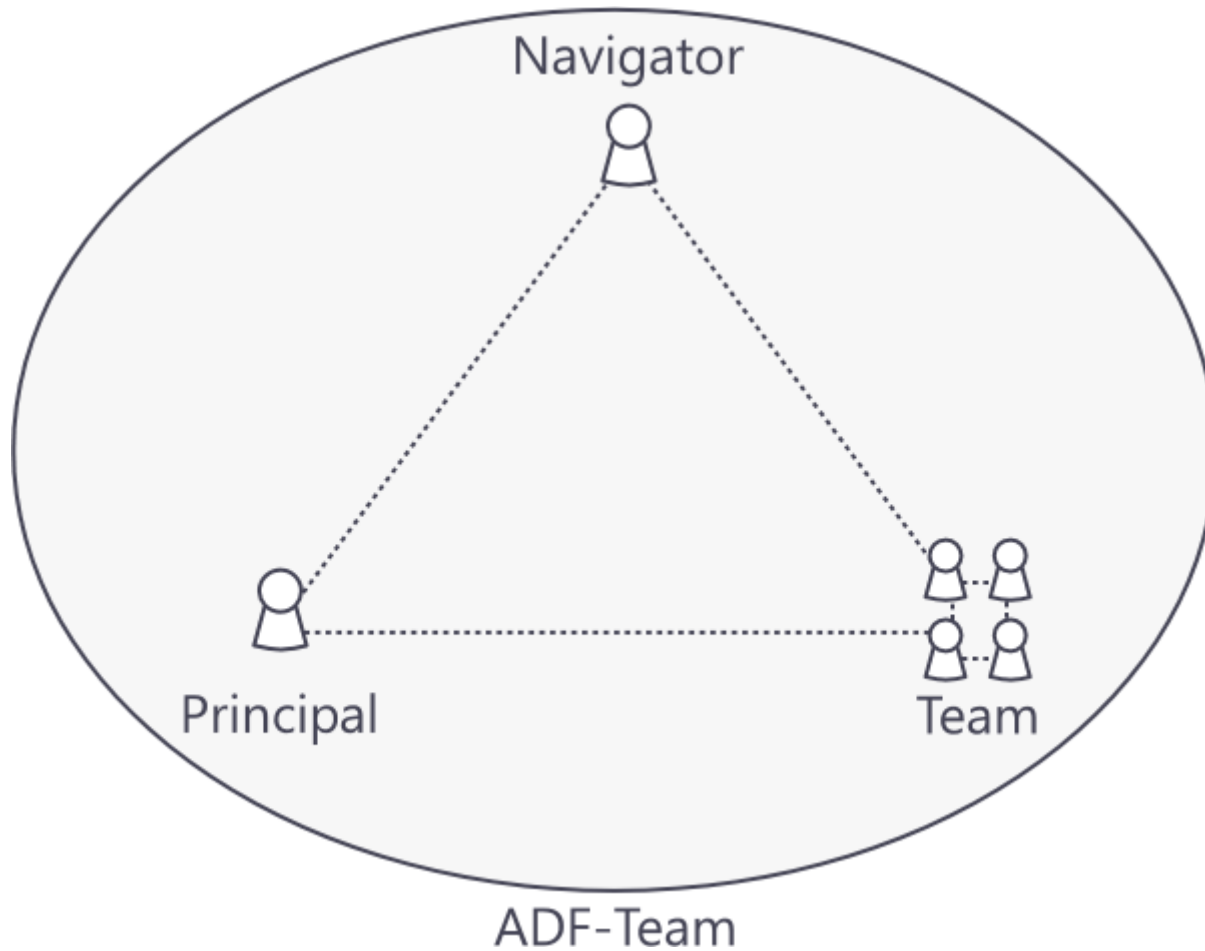
(Quelle: <http://www.noop.nl/2009/07/commit-to-sprint-planning-or-definition-of-done-not-both.html>)

ADF Blue



> ADF Blue – für Teams

Das Team – die Rollen?



Das Team – die Größe?



... und warum gerade 4?

Ich weiß – Du auch?

- > Wissensmanagement
 - > Wissen muss im gesamten Team geteilt werden
 - > Gegenseitiges Voranbringen und Helfen
- > Seiteneffekte
 - > Ausfälle kompensieren – selbst wenn Lemminge im Team sind
 - > Über den Tellerrand blicken
 - > Voneinander lernen
- > Beispielsweise
 - > Programmieren in Paaren
 - > Codereviews



(Quelle: <http://www.nichtlustig.de/toondb/050528.html>)

Es war einmal – eine Geschichte?



- > Artefakte & Werkzeuge
 - > Anforderungen wandeln sich im Lauf der Zeit
 - > Kein „Big design up-front“
- > Storycards
 - > Als Download im PDF-Format verfügbar
 - > Dienen als virtueller Knoten im Taschentuch
- > Beispielsweise
 - > ADF wird mit sich selbst verwaltet und entwickelt

Feature

ADF-Storycards als Download im PDF-Format verfügbar machen

Priorität **1**

Geschätzte Zeit **4**

Tatsächliche Zeit **6,5**

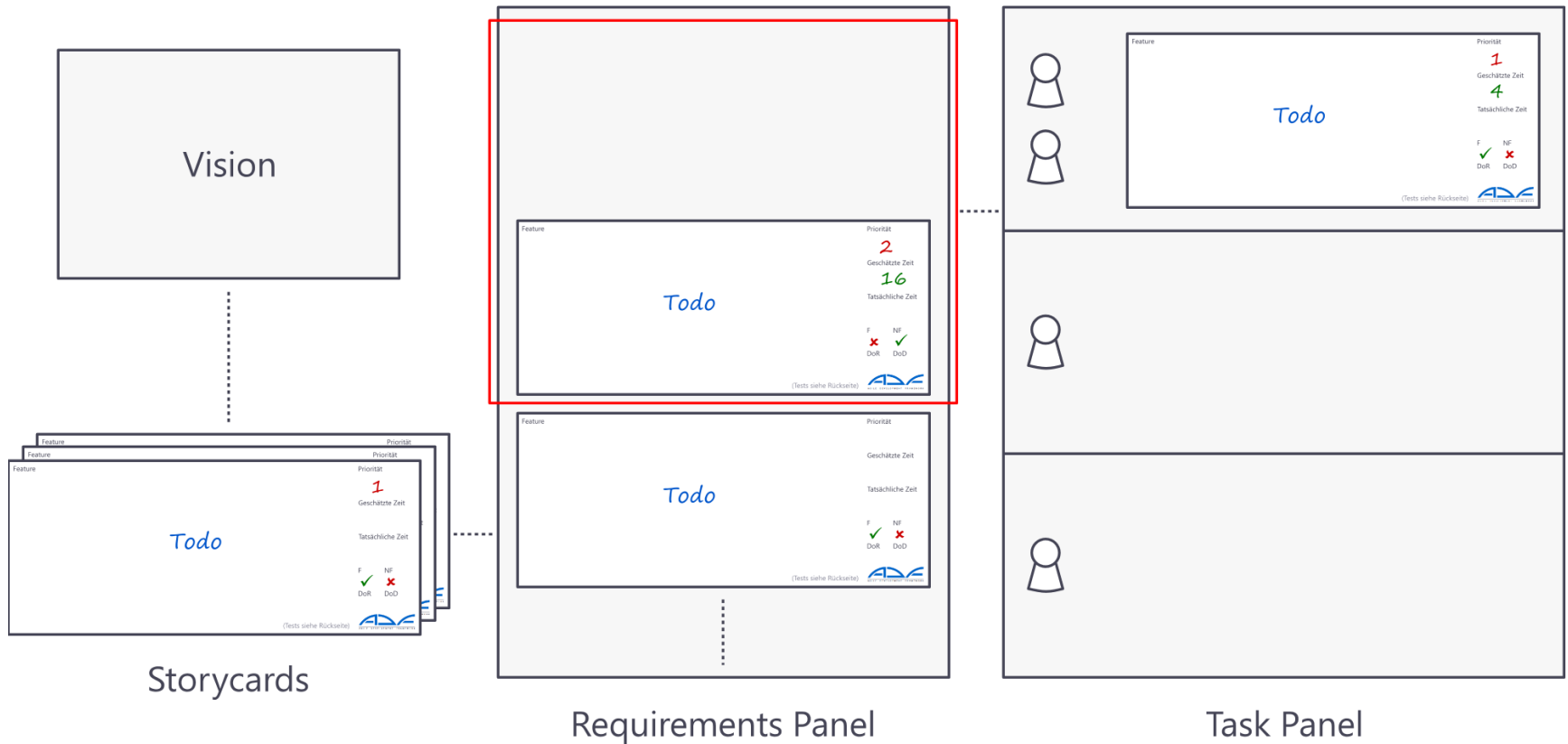
F ✓ NF ✗

DoR ✓ DoD ✓

(Tests siehe Rückseite)

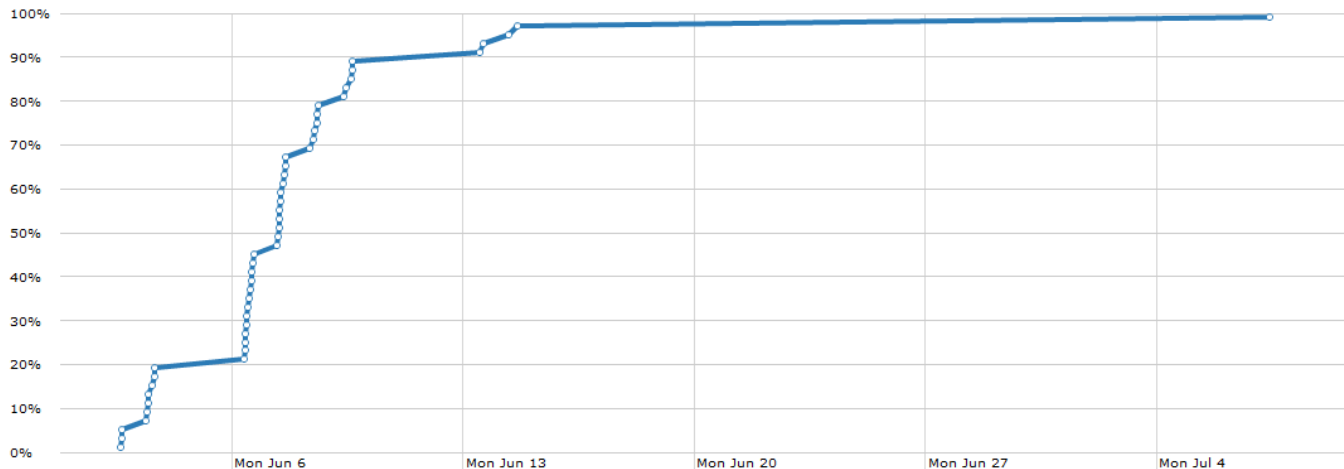


Der Prozess – Stream-basiert?



Aufwand schätzen – raten?

- > Evidence-Based Scheduling
 - > Features zerkleinern
 - > Tasks dürfen nicht größer als 16 Stunden sein
 - > Zeit messen
 - > Einschließlich Unterbrechungen, Pausen & Co
 - > Velocity = geschätzte Zeit / tatsächlich benötigte Zeit
 - > Zukunft simulieren
 - > Monte Carlo-Simulation über die verschiedenen Velocity-Werte



ADF Gray



> ADF Gray – für Unternehmen

Vereint, verstreut – verteilt?

- > Typische Probleme verteilter Teams
 - > Koordination
 - > Synchronisation
- > Logische Konsequenz
 - > Isolierte Teams als Spezialeinheiten
 - > Kernkompetenzen und Wissensinseln
- > Arbeitsweise
 - > Featurebezogene Teams
 - > Keine Colocation fordern



(Quelle: <http://www.haz.de/Nachrichten/Politik/Deutschland-Welt/Fusion-von-Bundespolizei-und-BKA-vom-Tisch>)

Der Prozess – verteilt?



- > Prozess von ADF Gray
 - > Leichtgewichtige Variante des Prozesses von ADF Blue
- > Prime Principal
 - > Prime Requirements Panel
 - > Prime Requirements Selection
- > Core Team
 - > Ubiquitous Language
 - > Blueprints und Styleguides
 - > Technologiestrategie



(Quelle: <http://www.grixartig.de/herausragend.html>)

Fazit

- > Erkenntnisse
 - > Zeit, Kosten und Umfang sind diskutabel
 - > Qualität als einziger Faktor nicht
- > Agile Development Framework (ADF)
 - > Konsistente agile Methode für qualitativ hochwertige Softwareentwicklung
 - > Evolution, Eleganz, Kommunikation und Vertrauen
- > Drei aufeinander aufbauende Stufen
 - > ADF White – für Individuen
 - > ADF Blue – für Teams
 - > Stream-basierter Prozess
 - > Evidence-Based Scheduling (EBS)
 - > ADF Gray – für Unternehmen

Weiterführende Informationen

> Webseite

> <http://www.agile-development-framework.net>

> Soziale Netzwerke

> http://www.twitter.com/adf_net

> <http://www.facebook.com/agiledevelopmentframework>

> Zeitschriften

> dotnetpro

> 03.2011 – Agil 2.0

> 04.2011 – Im Kleinen Großes bewirken

> 05.2011 – Vertrauen ist gut, Kontrolle ist schlechter

> 06.2011 – Vereint, verstreut, verteilt

> 07.2011 – Mit weniger mehr erreichen

> 08.2011 – Mut zum Anfang

Feedback

- > Fragen, Anregungen, Lob oder Kritik?
 - > www.goloroden.de