Wie uns Pfeiler, Werte und Prinzipien agiler Methoden bei Projekten unterstützen können

Transform2Agile

Kanban

- 1947 von Taiichi Ohno bei Toyota erfunden.
- Pull statt Push Prinzip
- David J. Anderson nahm es dann für die Wissensarbeit im Unternehmen
- Kanban hat 4 Grundprinzipien und 6 Praktiken

- (Kanban bedeutet wörtlich übersetzt Signalkarte)

Kanban Grundprinzipien

- Beginne mit dem, was du jetzt tust (Start where you are)
- Inkrementelle, evolutionäre Veränderungen verfolgen
- Aktuelle Prozesse, Rollen & Verantwortlichkeiten berücksichtigen
- Zu Führungsverantwortung auf allen Ebenen ermutigen. Kontinuierliche Verbesserung betrifft alle Ebenen (KAIZEN)

Kanban Praktiken

- Den Workflow visualisieren (Kanban Board)
- Laufende Arbeit begrenzen (Work in Progress Limits (WIP Limit)) setzen) =>Stop starting, start finishing !
- Workflow Management (Arbeit als steter Fluss)
- Prozessrichtlinien ausformulieren (nur was bekannt ist, kann auch verbessert werden)
- Feedbackschleifen

Crystal

- Methodengruppe entwickelt von Alistair Cockburn bei IBM/2001 Unterzeichner agiles Manifest
- Die Methoden sind mit Farben benannt: Crystal Clear, Crystal Yellow, Crystal Orange, Crystal Orange Web, Crystal Red, Crystal Magenta/Maroon, Crystal Diamond (optional), Crystal Blue, Crystal Sapphire (optional).
- Die Farbe spiegelt im Wesentlichen die Personenanzahl wider. So wird die einfachste Variante, Crystal Clear für Teamgrößen von zwei bis sechs Personen empfohlen.

Programmdefekte bedeuten Gefahr für	Anzahl Beteiligte			
	1–6	6–20	20–40	60–100
Leben	-	-	-	-
Kritische Gelder	-	E20	E40	E100
Verfügbare Gelder	D6	D20	D40	D100
Komfort	C6	C20	C40	C100
verwendete Methodik	Crystal Clear	Crystal Yellow	Crystal Orange	Crystal Red

Crystal Prinzipien 1

- Passiver Wissenstransfer
- Durch räumliche Nähe und Freiräume für Gespräche wird informeller, "passiver" Wissenstransfer gefördert.
- Persönliche Sicherheit (Safety Culture)
- Kritik und Befürchtungen können ohne Repressalien geäußert werden.
- Laufende Kritik und Verbesserung
- Es werden laufend Verbesserungsvorschläge gesucht, gesammelt und die Wichtigkeit ihrer Umsetzung bewertet.
- Fokussiertes Arbeiten

Crystal Prinzipien 2

- Die Mitarbeiter wissen genau, was ihr Ziel ist, und werden nicht abgelenkt oder für andere Projekte abgezogen.
- Häufige Releases
- Durch häufige Herausgabe von Zwischenversionen an den Kunden oder andere Projektbeteiligte wird vermieden, dass Erwartungen angestaut werden und größerer Erklärungsbedarf entsteht. Gleichzeitig kann eine höhere Sicherheit für das Team durch Zwischenabnahmen entstehen.
- Zugang zu kundigen Benutzern

  - Dadurch, dass ständig ein erfahrener Benutzer des künftigen Produktes erreichbar ist, können Detailfragen schnell und formlos geklärt werden.

Crystal Prinzipien 3

-Dies vermeidet unter anderem, dass Missverständnisse zu Problemen auswachsen.

- Automatisiertes Testen
- Durch Unit Testing wird für dauerhaft stabilen Programmcode gesorgt, was auch das Vertrauen des Teams in die eigene Arbeit stärkt.
- Häufige Integration
- Nicht nur der Programmcode wird getestet, es wird auch regelmäßig (z. B. täglich und automatisiert) eine lauffähige Testversion erstellt.
- Konfigurationsmanagement
  - Verwendung von Konfigurationsmanagement, oder zumindest einer Versionsverwaltung.

Lean Management

- Taiichi Ono bei Toyota
- Arbeitsplatz steht im Mittelpunkt
- Prozesse sind so abzustimmen, dass Kundenbedürfnisse bei gleichzeitiger Rentabilität sichergestellt werden
- Fokus auf die Wertschöpfung
- Verantwortung wird vermehrt an die Mitarbeiter übertragen
- Vermeidung von Verschwendung

Lean Management

- Verbesserte Kommunikation intern und extern
- Konzentration auf das Wesentliche
- Kundenorientierung

Lean Prinzipien und Ziele

- Kundenorientierung
- Wertstrom identifizieren
- Fluss Prinzip
- Pull statt Push
- Kontinuierliche Verbesserung(Kaizen)
- Vermeidung von Verschwendung unterteilbar in Muda(Verschwendung(7Arten)), Mura(Unausgeglichenheit und Muri(Überlastung)
- Überflüssige Materialbewegung,hohe Lagerbestände,Schlechte Ergonomie,überflüssige Wartezeiten,Verarbeitung,Überproduktion,Ausschuss und Korrekturen

Lean Werkzeuge

- Kanban
- PDCA Cycle (Plan Do Act Check) auch als Deming Cycle bekannt
- Value Stream Mapping (VSM) (Prozessdarstellung entlang des Wertstroms)
- Und weitere die hier zu weit führen würden

- https://www.it-agile.de/agiles-wissen/agile-entwicklung/was-ist-testgetriebene-entwicklung/#:~:text=Test%2DDriven%20Development%2C%20TDD),dass%20der%20Test%20erfolgreich%20durchl%C3%A4uft.

VUCA Welt

- V = Volatile (wechselhaft)
- U = Uncertainty (unsicher)
- C = Complexity (komplex)
- A = Ambigious (mehrdeutig)

- Wie können wir damit umgehen?

Lean UX

- Mehr ein Mindset als ein Framework
- Methodensammlung
- Iterativ

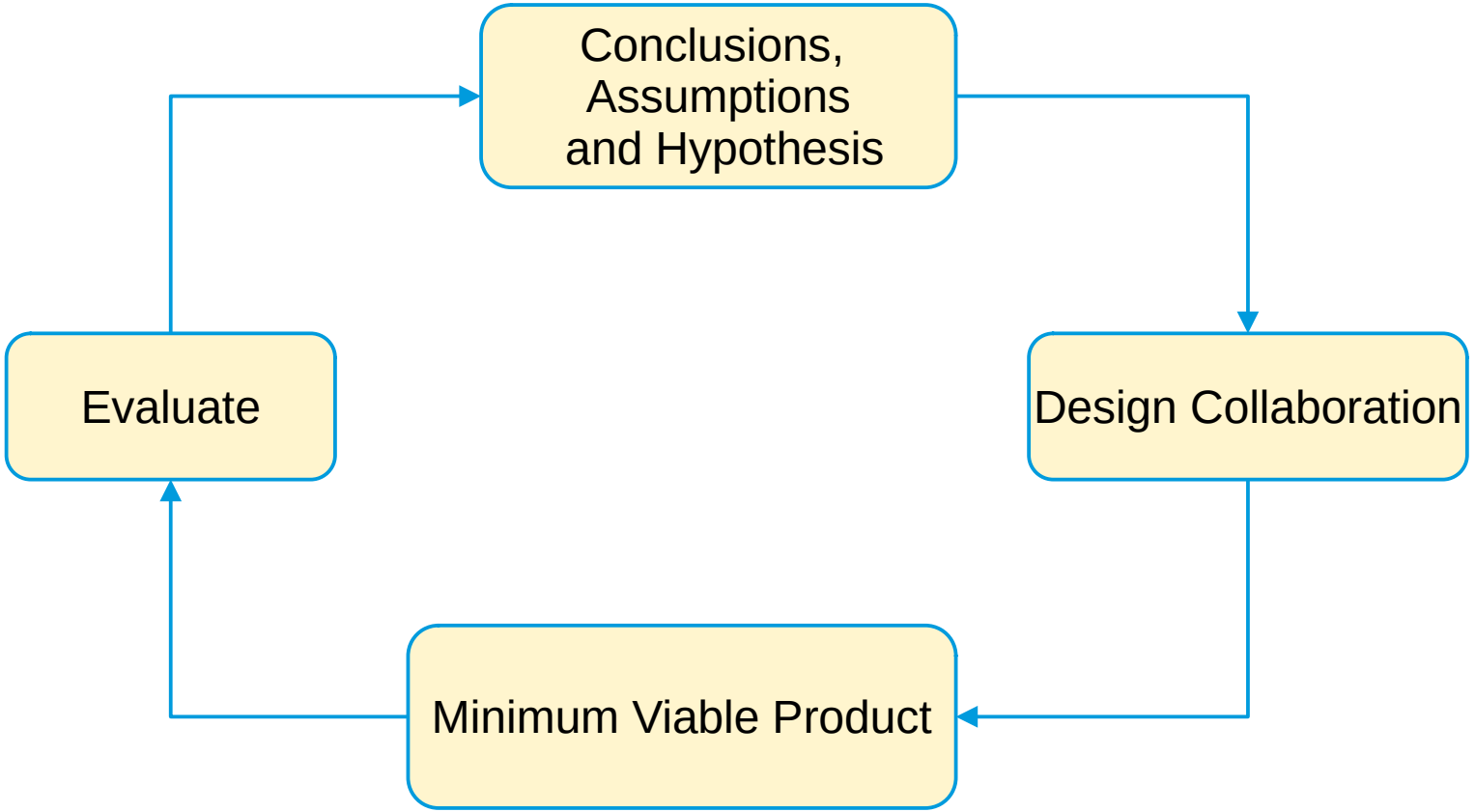
Lean UX Grundprinzipien

- Nutzerzentriertes Vorgehen
- Interdisziplinäre Teams
- Verschwendung minimieren
- Kleine Arbeitspakete
- Ständiges Testen und Lernen

Lean UX Grundprinzipien Teil 2

- Lernen vor Wachstum
- Misserfolge sind erlaubt
- Machen statt analysieren
- Think big, start small !

Lean UX Prozessphasen

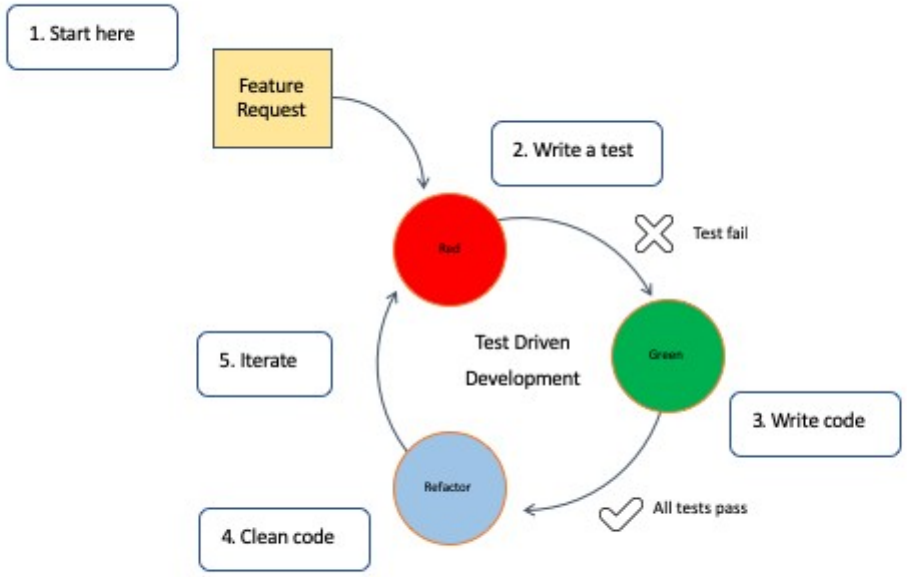


Test Driven Development

- Bei der testgetriebenen Entwicklung (Test-Driven Development, TDD) werden Tests dazu benutzt, um die Softwareentwicklung zu steuern. Der Ablauf dieser Programmierung ist zyklisch:

- Ein Test wird geschrieben, der zunächst fehlschlägt.
- Genau so viel Produktivcode wird implementiert, dass der Test erfolgreich durchläuft.
- Test und Produktivcode werden refaktorisiert.

Test Driven Development



Test Driven Development TDD

- Auf Basis des Red-Green Testings wird die Basis für CI/CD geschaffen, da folgender Code damit automatisch bei der Freigabe getestet wird

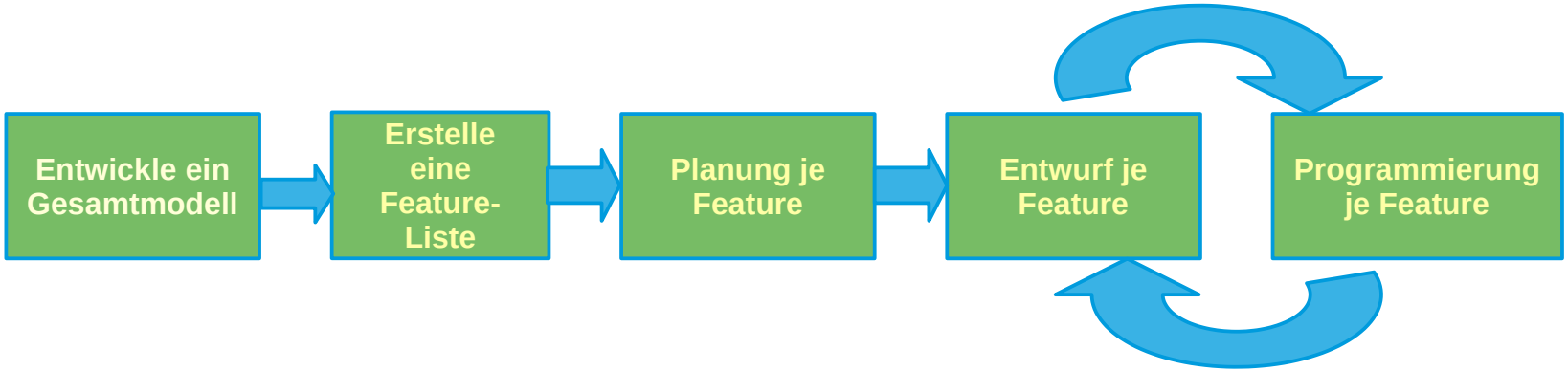
Feature Driven Development

- Definiert von Jeff de Luca 1997 für ein großes zeitkritisches Projekt (50 Entwickler)
- FDD-Projekte durchlaufen fünf Prozesse.

FDD Prozesse

- Prozess #1: Entwickle ein Gesamtmodell (Rollen: alle Projektbeteiligten)
- Prozess #2: Erstelle eine Feature-Liste (Rollen: in der Regel nur die Chefprogrammierer)
- Prozess #3: Planung je Feature (Rollen: Projektleiter, Entwicklungsleiter, Chefprogrammierer)
- Prozess #4: Entwurf je Feature (Rollen: Chefprogrammierer, Entwickler)
- Prozess #5: Programmierung je Feature (Rollen: Entwickler)

FDD Prozessübersicht



Die ersten drei Prozesse werden innerhalb weniger Tage durchlaufen. Die Prozesse 4 und 5 werden in ständigem Wechsel durchgeführt, weil jedes Feature in maximal zwei Wochen realisiert wird.

FDD Rollen

- Project Manager
- Chief Architect
- Development Manager
- Chief Programmer
- Class Owner (Klasse ist die kleinste Einheit)
- Domain Experts

FDD Hilfsrollen

- Dazu kommen noch weitere Supportrollen
- Domain Manager
- Release Manager
- Language Guru
- Build Engineet
- Toolsmith
- System Administrator
- Testers
- Developers
- Technical writers

FDD Gedanken

- Zuerst muss man verstehen was ist das Feature
- Dann wie können wir es auf das Feature hin entwickeln
- Dazu braucht es die einzelnen Funktionen, die wir herausfinden müssen, damit das Feature auch das macht was es soll.
- Bild ändern Feature in einer Präsentationssoftware benötigt die Funktionen, hinzufügen, ändern, löschen, Farbpalette, Resize, evtl Hintergrund ersetzen.
- Wenn man das weiß, kann man die Class Owner , wenn ihre Class an der Reihe ist, mit dem Feature Team verbinden

DSDM (Dynamic System Development Method)

- 1994 begründet
- DSDM fordert die Verwendung von verschiedenen erprobten Handlungsweisen, die unter anderem beinhalten:
- Workshops ermöglichen
- Modelle und interative Entwicklung
- MoSCoW Priorisierung
- Time boxing (d.h. Zeitbeschränkungen)

MoSCoW- war das nicht in Russland?

- MoSCoW ist eine Abkürzung für
- Must
- Should
- Could
- Would (not at this moment)
- Bei der Abstimmung mit Stakeholdern ist klar, was die Konsequenz ist, wenn ein Punkt entsprechend kategorisiert wurde
- Hilft nicht bei Begeisterungsfaktoren (siehe z.B. Kano Modell)

DSDM 8 Prinzipien

Fokus auf den Business Bedarf

- Liefere pünktlich
- Zusammenarbeit
- Keine Abschwächung der Qualitätsziele
- Baue inkrementell auf stabilen Fundamenten
- Entwickle iterativ
- Kommuniziere regelmäßig und klar
- Zeige dass du das Projekt steuerst (demonstrate control)

Extreme Programmng (XP)

- Ursprung Chrysler Comprehensive Compensation (c3) program
- Mitte der 90er Jahre
- Kent Breck und Ron Jeffries
- Zusammenstellung von Engineering Methoden mit dem Ziel von angewandter Exzellenz, Lieferung hoher Qualität, bei gleichzeitig hoher Lebensqualität für die Entwickler
- Kent Beck veröffentlichte unter anderem ein Bucht mit dem Titel „Extreme Programming Explained: Embrace Change“

Extreme Programming Rollen

- Kunde
- Entwickler
- Tracker (KPI tracker)
- Coach

Extreme Programming Werte

- Kommunikation
- Feedback
- Courage (Mut)
- Respect
- Einfachheit (was ist das Einfachste, das den Zweck erfüllt?)

Extreme Programming Methoden 1st Edition

- The Planning Game
- Small Releases
- Metaphor
- Simple Design
- Testing
- Refactoring
- Pair Programming

- Collective Ownership
- Continuous Integration
- 40-hour week
- On-site Customer
  Coding Standard

Extreme Programming Practices 2nd Edition

- Sit Together
- Whole Team
- Informative Workspace
- Energized Work
- Pair Programming
- Stories
- Weekly Cycle = eine Iteration

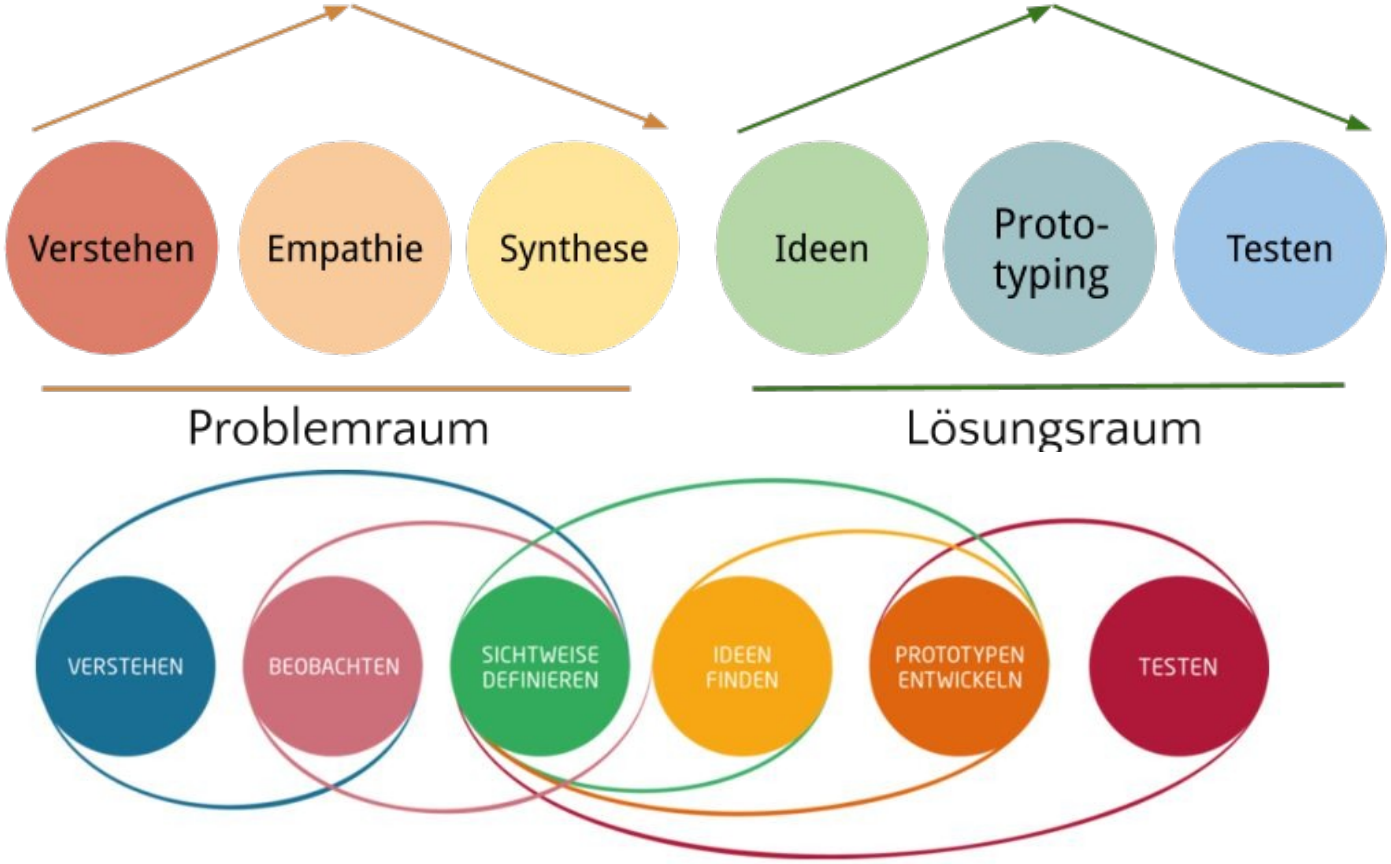
Extreme Programming 2nd Edition

- Quarterly Cycle = Release
- Slack (Jede Art von Tätigkeit die zurückgestellt werden kann ist eine Slack Practice.)
- Ten-Minute Build
- Continuous Integration
- Test-First Programming (Write failing automated test -> Run failing test -> develop code to make test pass -> run test -> repeat)
- Incremental Design

Design Thinking

- Von Larry Leifer (Stanford), Terry Winograd (Ausbilder von Larry Page) und David Kelley im Jahre 1991 begründet
- Menschenzentrierter Ansatz
- Iterativ
- Direkter Kundenkontakt

Design Thinking Phasen



Design Thinking Methode und Werte

- Starte mit einem Beginner Mind(=nichts voraussetzen)
- Erst das Problem verstehen, dann Lösungen entwickeln
- Der Kunde steht im Fokus
- Du bist bereit nicht tragfähige Lösungen zu entwerfen
- Du bist erst fertig, wenn deine erfolgversprechende Idee materialisiert und implementiert ist

Agiles Manifest

- Im Jahr 2001 von 17 Entwicklern verfasst
- Ziel schneller und besser fertige Software zu liefern
- „Wir entdecken bessere Wege zur Entwicklung von Software, indem wir es tun und anderen helfen, es zu tun!“

Agiles Manifest 4 Werte

- Individuen und Interaktionen sind wichtiger als Prozesse und Werkzeuge
- Funktionierende Produkte sind wichtiger als umfassende Dokumentation
- Zusammenarbeit mit dem Kunden ist wichtiger als Vertragsverhandlungen
- Reagieren auf Veränderungen ist wichtiger als das Befolgen eines Plans

Agiles Manifest 12 Grundprinzipien Teil1

- Kundenzufriedenheit durch frühzeitige und kontinuierliche Softwarebereitstellung
- Anpassung an sich ändernde Anforderungen während des gesamten Entwicklungsprozesses
- Häufige Lieferung von Arbeitssoftware
- Zusammenarbeit zwischen den geschäftlichen Interessengruppen und den Entwicklern während des gesamten Projekts
- Unterstützung, Vertrauen und Motivation der beteiligten Personen
- Ermöglichen Sie Interaktionen von Angesicht zu Angesicht
- Funktionierende Software ist der primäre Maßstab für den Fortschritt

Agiles Manifest 12 Grundprinzipien Teil2

- Agile Prozesse zur Unterstützung eines konsistenten Entwicklungstempos
- Aufmerksamkeit für technische Details und Design erhöht die Agilität
- Einfachheit – Entwickeln Sie gerade genug, um die Aufgabe im Moment zu erledigen.
- Selbstorganisierende Teams fördern großartige Architekturen
- Regelmäßige Überlegungen, wie man effektiver werden kann – Selbstverbesserung, Prozessverbesserung, Weiterentwicklung von Fähigkeiten und Techniken

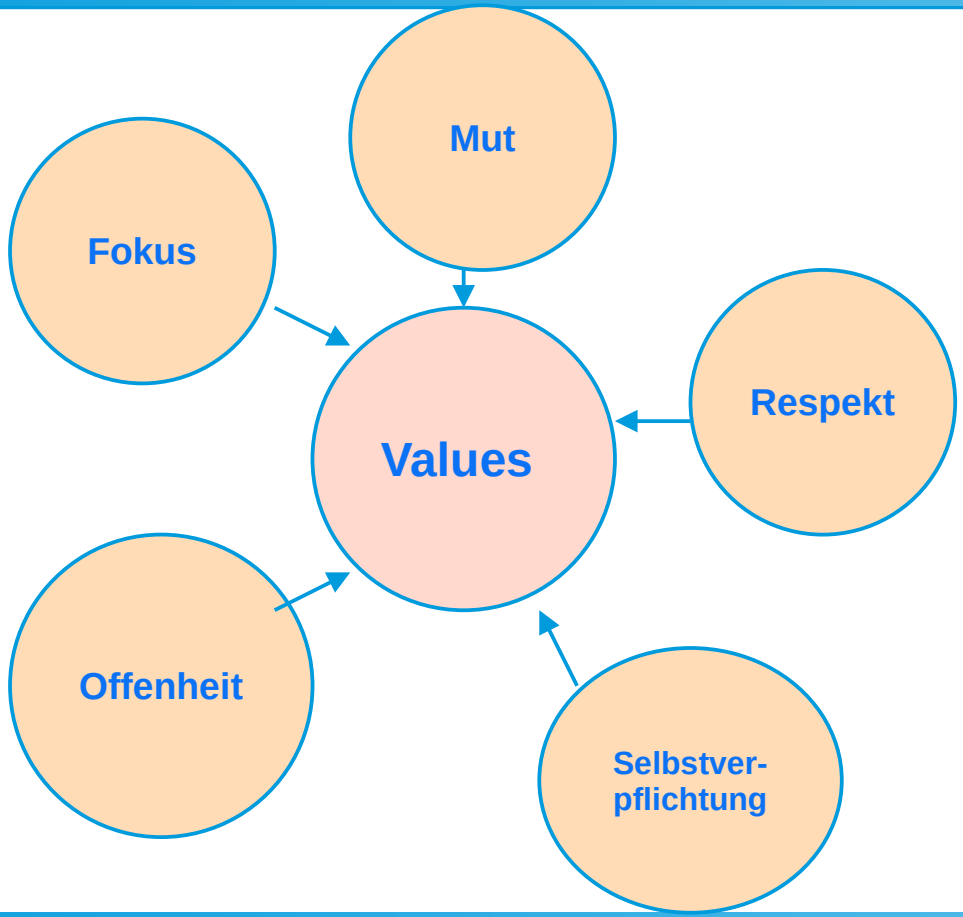
Scrum

- Ken Schwaber / Jeff Sutherland ab 1993, erste Veröffentlichung 2004
- Empirisch, inkrementell und iterativ
- Teamgröße 3-9 Personen
- 1 Team
- 1 Produkt
- 4 Events (Veranstaltungen, mit Zeitlimit (time boxed)
- 3 Artefakte
- 3 Rollen

Scrum 3 Säulen

- Transparenz
- Überprüfung
- Anpassung

Scrum Werte



4 Scrum Events

- Sprint Planning
- Daily Scrum
- Review
- Retrospective

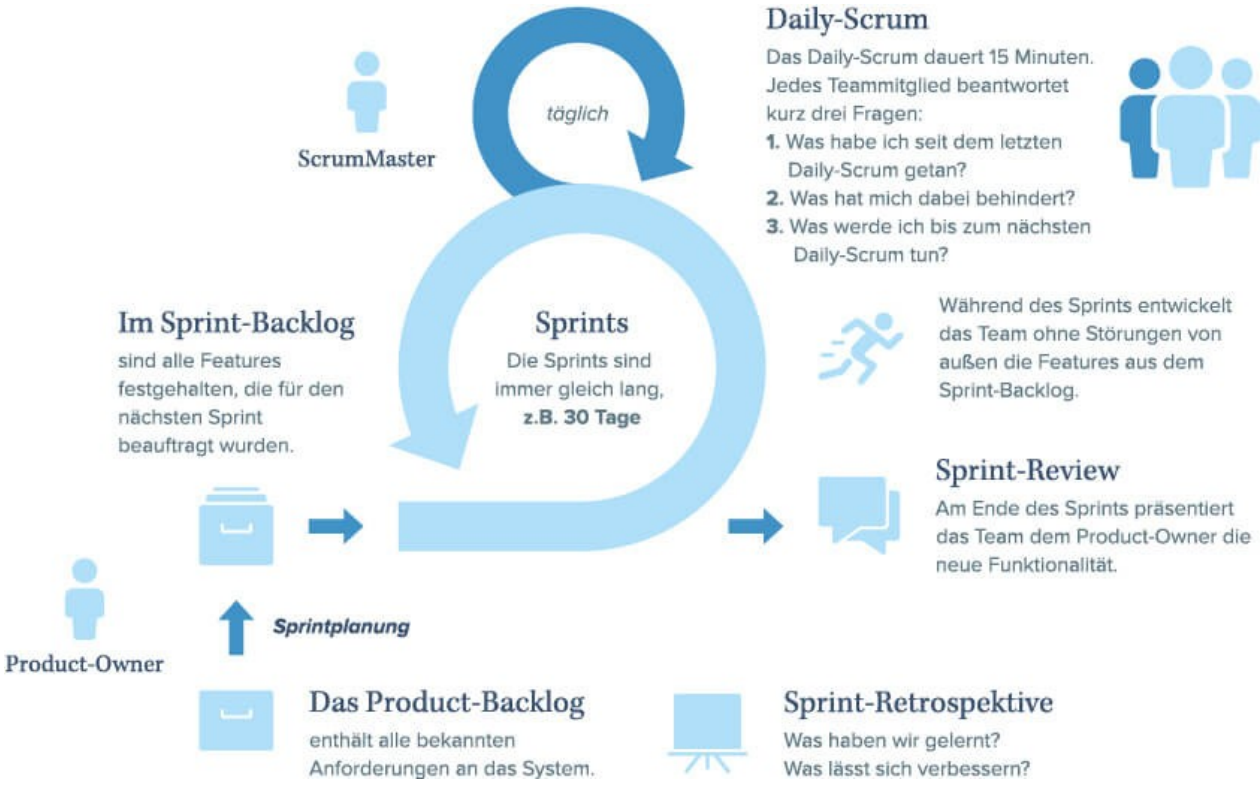
Scrum Rollen

- Product Owner
- Scrum Master
- Developer Team
- Alle zusammen sind das Scrum Team

Scrum Artefakte

- Product Backlog
- Sprint Backlog
- Increment (MVP= minimum viable product)
- Zusätzlich gibt es eine DOR (definition of ready) und eine DOD(Definition Of Done)

Scrum Prozess



Nexus

- Ken Schwaber
- Werte und Prinzipien wie bei Scrum
- 1 Produkt
- Mehrere Teams (3-9 Teams)

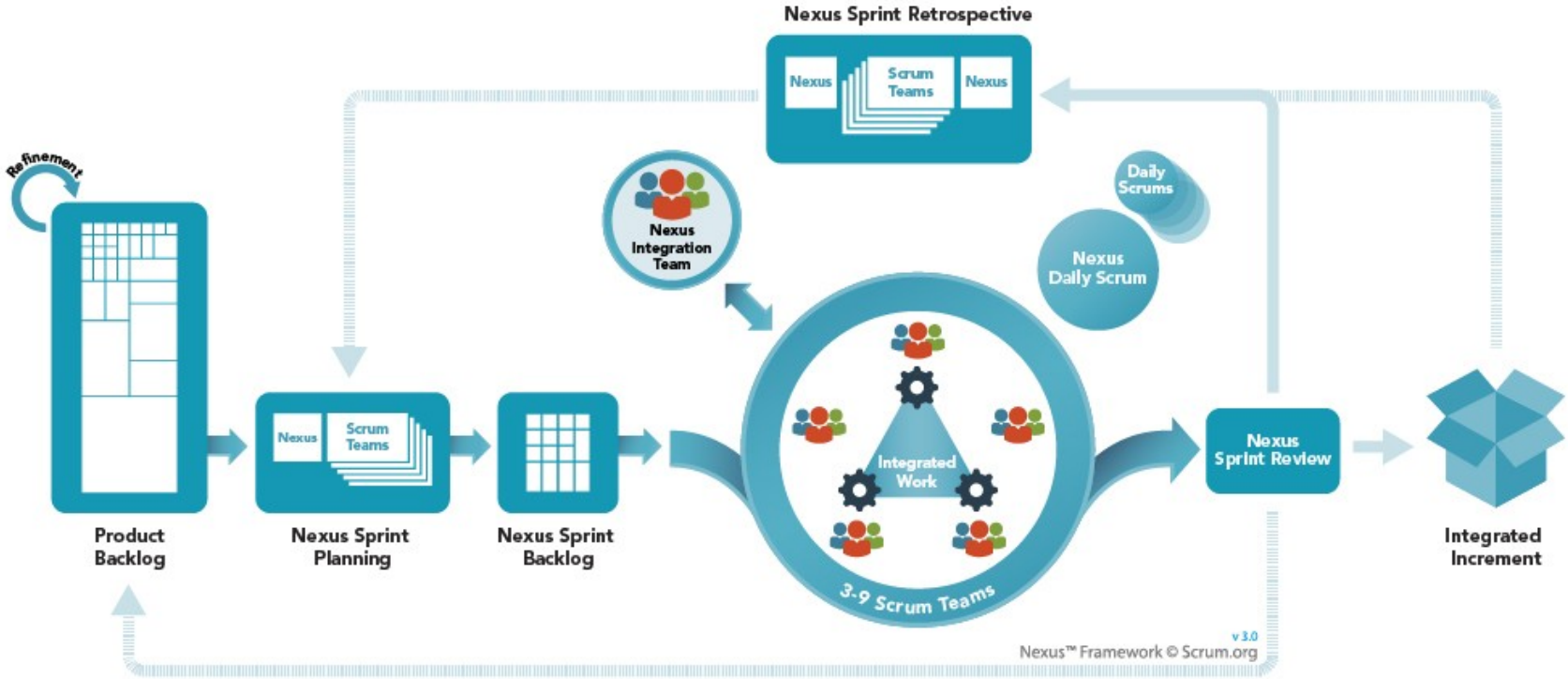
Nexus Rollen

- Neue Rolle Nexus Integration Team
- Ein Team mit 3-9 Personen, Scrum Master, Product Owner, Developer Team
- Dazu dann 3-9 Scrum Teams

Nexus Events

- Wie bei Scrum
- Neu : Nexus Daily Scrum

Nexus Prozess



LeSS (Large Scale Scrum)

- Entstehung durch Craig Larman und Bas Vodde 2005
- LeSS: 1 Produkt und bis zu 8 Teams (pro Team maximal 8 Personen)
- LeSS Huge: 1 Produkt und bis zu 2000 Personen

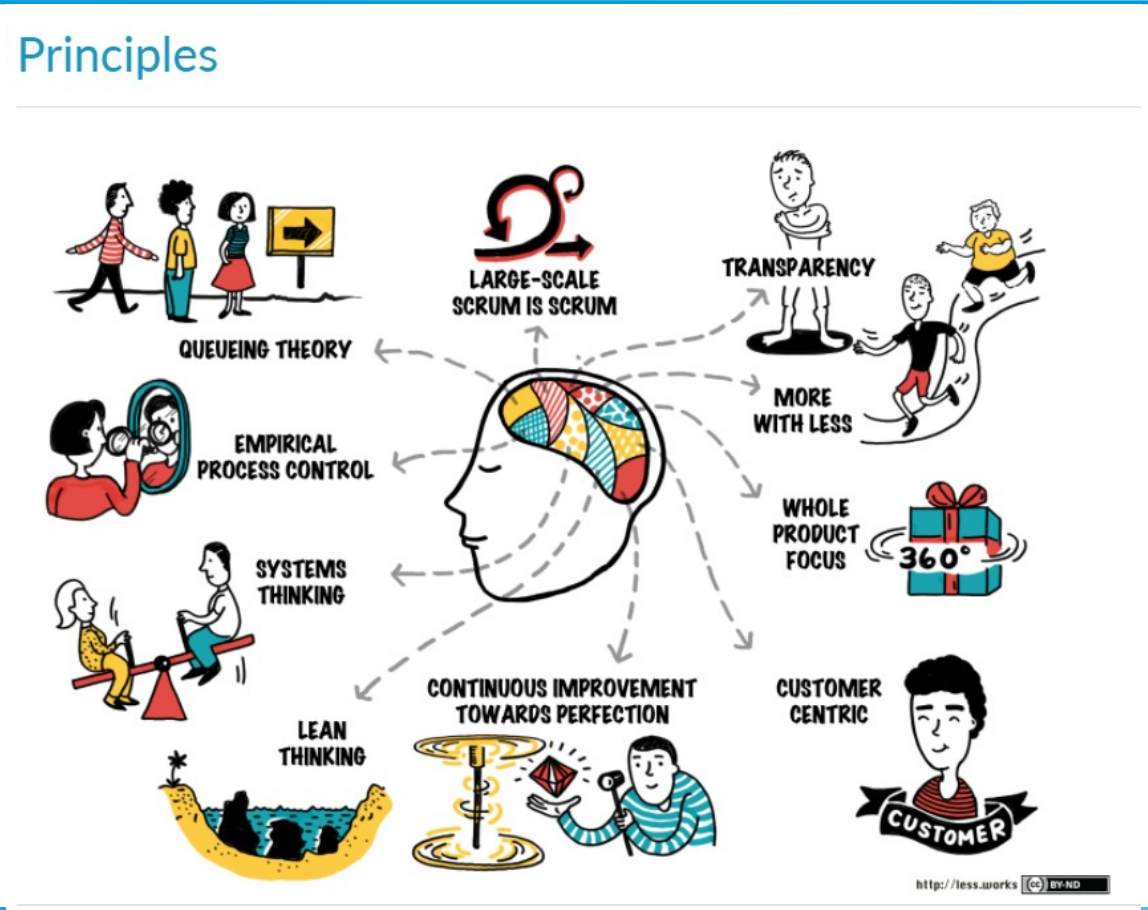
Less

- LeSS hat 1 Product Backlog
- LeSS hat 1 DOD (Definition Of Done)
- 1 potentiell lieferbares Produkt am Ende des Sprints
- 1 Product Owner
- 1 Sprint
- Viele cross-functional Leute, keine Spezialisten

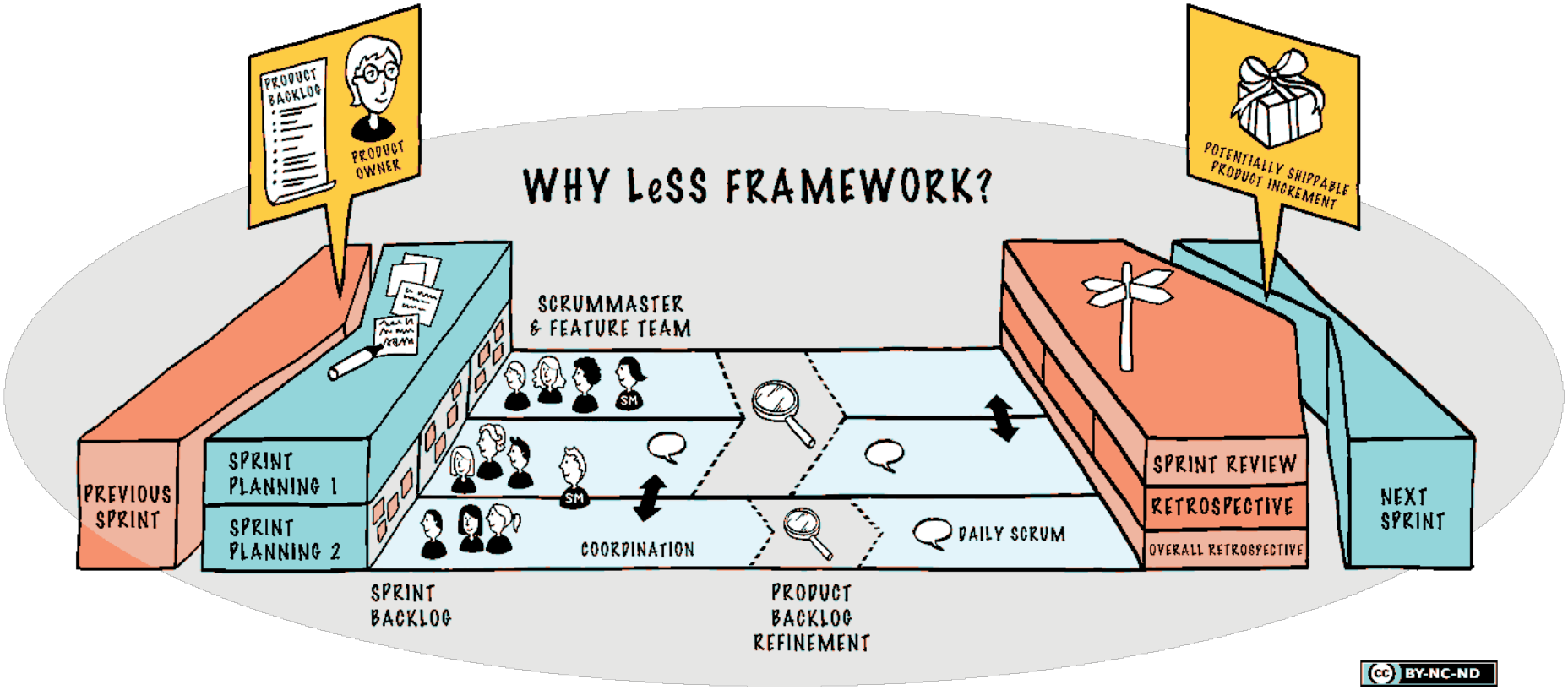
LeSS Spezialitäten

- Sprint Planning Part 1
- Sprint Planning Part 2
- Daily Scrum
- Coordination
- Overall PBR
- Product Backlog Refinement(PBR): The only requirement in LeSS is single-team PBR, the same as in one-team Scrum.
- Sprint Review
- Overall Retrospective

LeSS Prinzipien



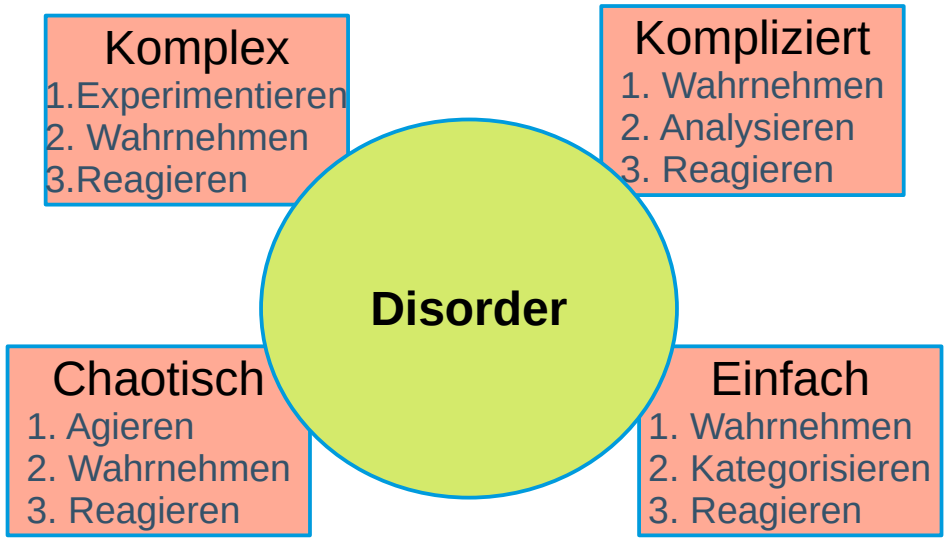
LeSS Prozess



Cynefin

- David Snowden 2020
- Wissensmanagement Modell mit der Aufgabe Probleme, Situationen und Systeme zu beschreiben
- Hintergrund: Verhalten beruht oft auf Erfahrungen aus der Vergangenheit, wurden aber in anderen Domänen gemacht und daher ist der Verhaltensansatz in einer anderen Domäne nicht passend.
- Voraussetzung: Ermitteln in welcher Domäne man sich befindet

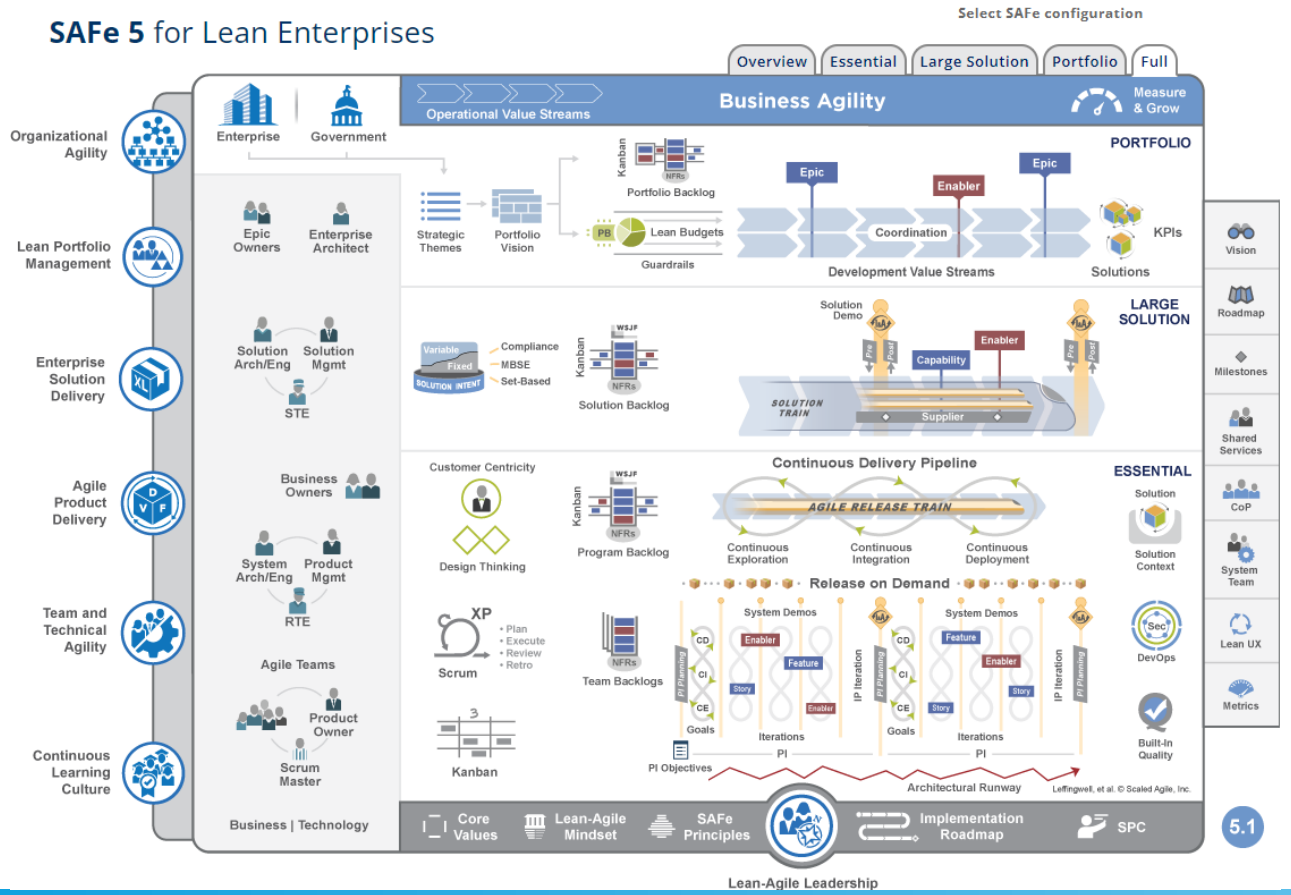
Cynefin Modell



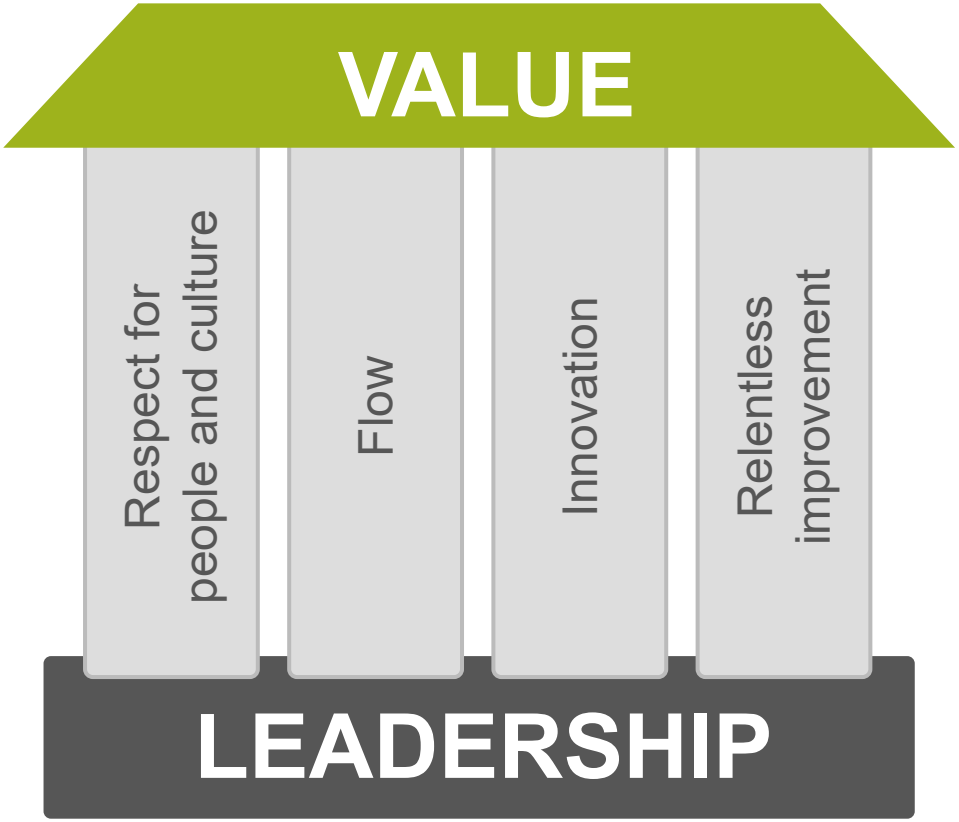
SAFe (Scaled Agile Framework)

- Dean Leffingwell und Drew Jemilo 2011
- Portfolio und Enterprise Level
- Basierend auf Mindset, Rollen, Werten,Prinzipien und Methoden

SAFe Framework



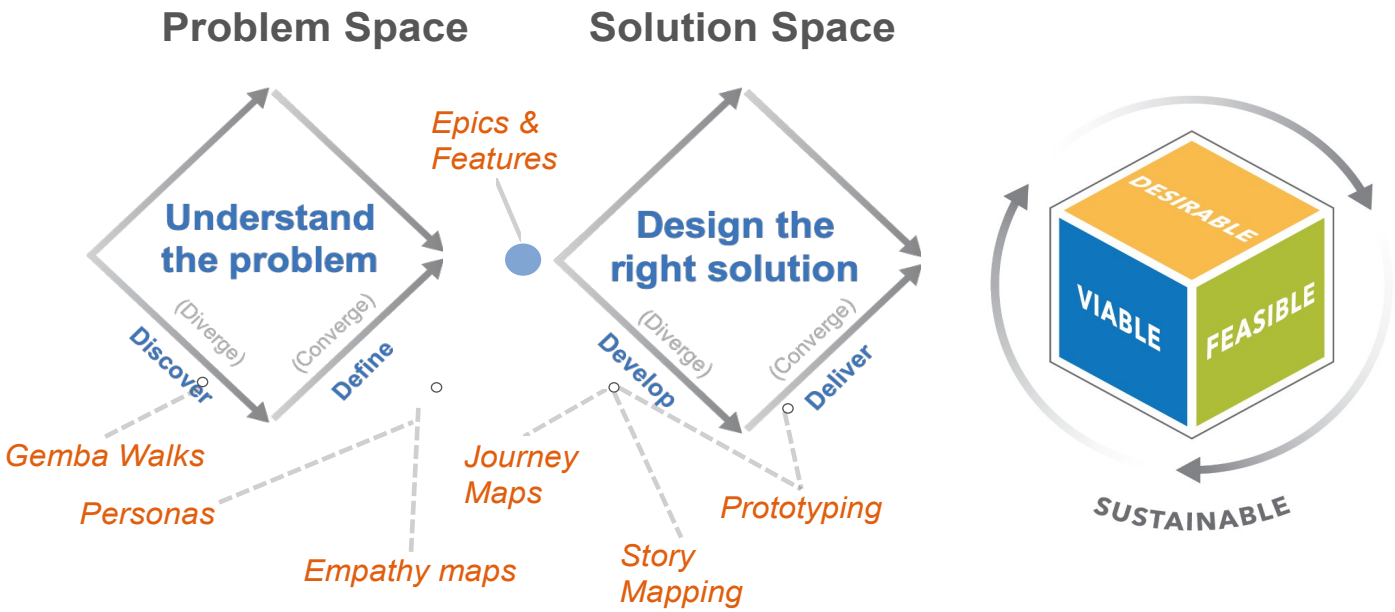
SAFe Werte


VALUE
Respect for people and culture
Flow
Innovation
Relentless improvement
LEADERSHIP

SAFe Lean-Agile Prinzipien

- Habe eine wirtschaftliche Sicht
- Wende systemisches Denken an
- Schätze Vielseitigkeit;bewahre Optionen
- Baue inkrementell mit schnellen, integrierten Lernzyklen
- Meilensteine basieren auf der Zielentwicklung von Arbeitssystemen
- Visualisiere und begrenze WIP, reduziere die Batch Größen und verwalte die Queue Länge
- Wende Kadenz und synchronisiere diese mit dem Cross-Domänen Plan
- Entfessele die intrinsische Motivation der Knowledge Worker
- Denzentralisiere die Entscheidungsfindung
- Organisiere alles um den Wert

Customer Centricity and Design Thinking

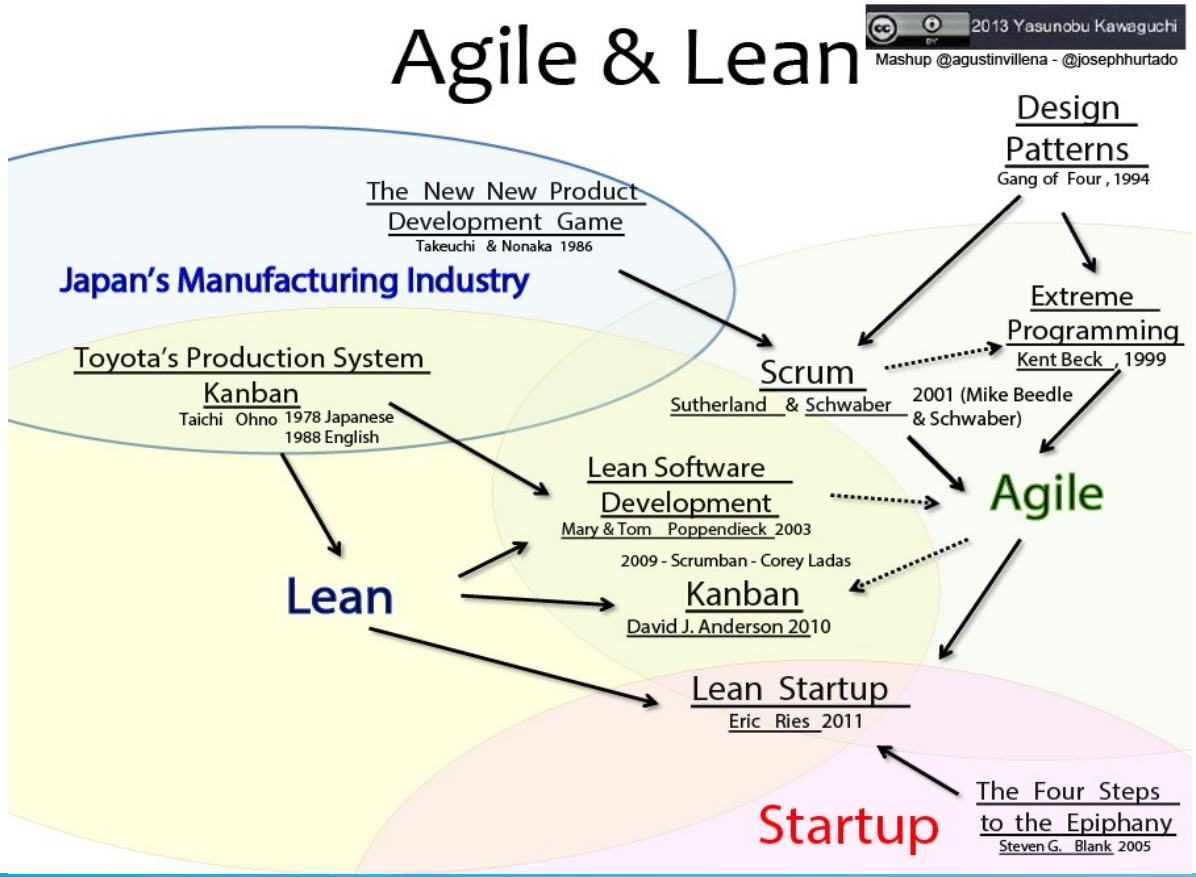


A clear and continuous understanding of the target market, customers, the problems they are facing and the jobs to be done

SAFe DevOps Prinzipien

- **C**ulture of shared responsibility
- **A**utomate everything
- **L**ean flow
- **M**easure flow
- **R**ecovery

Wie hängt alles zusammen?



Essenz

Wichtige Punkte

- Respektvolle und regelmäßige Kommunikation im Team
- Kommunikation mit den Nutzern und Stakeholdern (Feedback!)
- Aufgaben in kleine Portionen aufteilen
- Iterativ vorgehen
- Dokumentation nicht vergessen, aber nicht übertreiben
- Arbeit begrenzen (WIP Limit!)
- Fokus beibehalten
- Wert schaffen im Auge behalten

Essenz

- Regelmäßig testen
- Nicht nur das Produkt, sondern auch die eigene Arbeitsweise immer wieder überprüfen und verbessern (Inspect & Adapt)
- Respektvoller Umgang mit den anderen Teammitgliedern

Das Wichtigste zum Schluss

- Nicht den Menschen vergessen
- Nicht in Spielerei verlieren
- Wert im Auge behalten

Links und Quellen

- https://deming.org/wp-content/uploads/2020/06/PDSA_History_Ron_Moen.pdf
- https://www.it-agile.de/agiles-wissen/agile-entwicklung/was-ist-testgetriebene-entwicklung/#:~:text=Test%2DDriven%20Development%2C%20TDD),dass%20der%20Test%20erfolgreich%20durchl%C3%A4uft
- https://www.agilebusiness.org/page/whatisdsdm
- https://t2informatik.de/wissen-kompakt/moscow-methode/
- https://www.toolsqa.com/agile/feature-driven-development/
- https://de.wikipedia.org/wiki/Feature_Driven_Development

Links und Quellen

- http://agilelion.com/agile-kanban-cafe/agile-and-lean-influences-where-did-kanban-scrum-scrumban-come-from
- https://digitaleneuordnung.de/blog/design-thinking-methode/#:~:text=Design%20Thinking%20ist%20eine%20kundenzentrierte,Kundensicht%20%C3%BCberlegene%20L%C3%B6sung%20zu%20entwickeln

  https://teamentwicklung-lab.de/wp-content/uploads/2019/05/Design-Thinking-Prozess-6-Phasen.png
- https://www.scrum.org
- https://mission-mobile.de/files/2018/05/6-Phasen-des-Design-Thinkings.jpg

Links und Quellen

- Soonersaferhappier.com (Cartoon)

Links und Quellen

- https://www.agilest.org/scaled-agile/large-scale-scrum-less/
- Marc Löffler. Scrum Master Journey S. 180-184
- https://www.scaledagileframework.com/
- https://explainagile.com/agile/xp-extreme-programming/practices/slack/