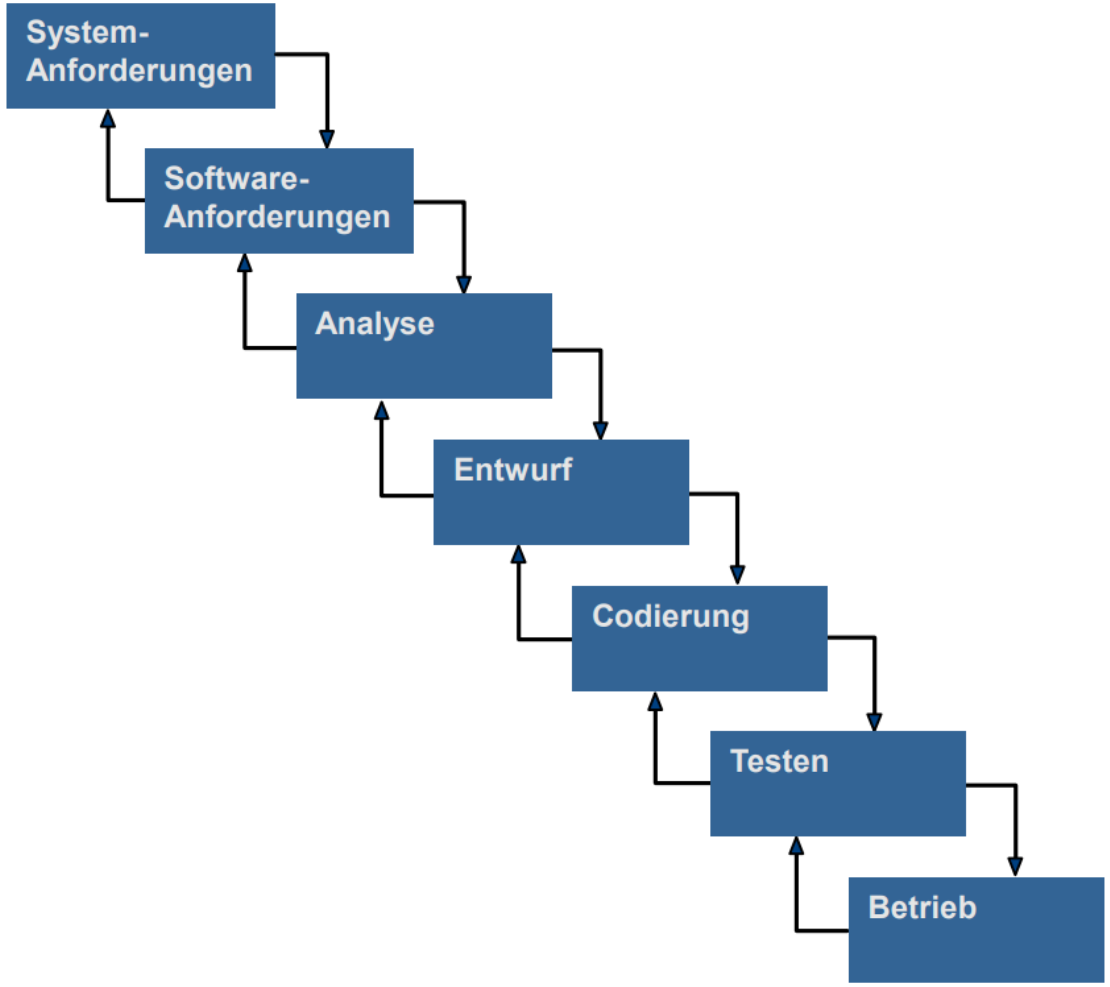
VORLESUNG 2

„SOFTWARE-ENGINEERING / SOFTWARE-TECHNIK"
IM WS 2022/2023

Dozent:
Oliver Bleisinger (extern)

DAS WASSERFALL-MODELL

- Software wird in sukzessiven Stufen (Phasen) entwickelt
- Name Wasserfall-Modell
  - Ergebnisse einer Phase fallen wie bei einem Wasserfall in die nächste Phase
- Charakteristika
  - Jede Aktivität ist in der richtigen Reihenfolge und in der vollen Breite vollständig durchzuführen
  - Am Ende jeder Aktivität steht ein fertiggestelltes Dokument
  - Dokumenten-getriebenes Modell
  - Der Entwicklungsablauf ist sequentiell
  - Jede Aktivität muss beendet sein, bevor die nächste anfängt
  - Orientierung am top-down-Vorgehen
  - Einfach, verständlich und benötigt nur wenig Managementaufwand
  - Benutzerbeteiligung ist nur in der Definitionsphase vorgesehen



DAS WASSERFALL-MODELL

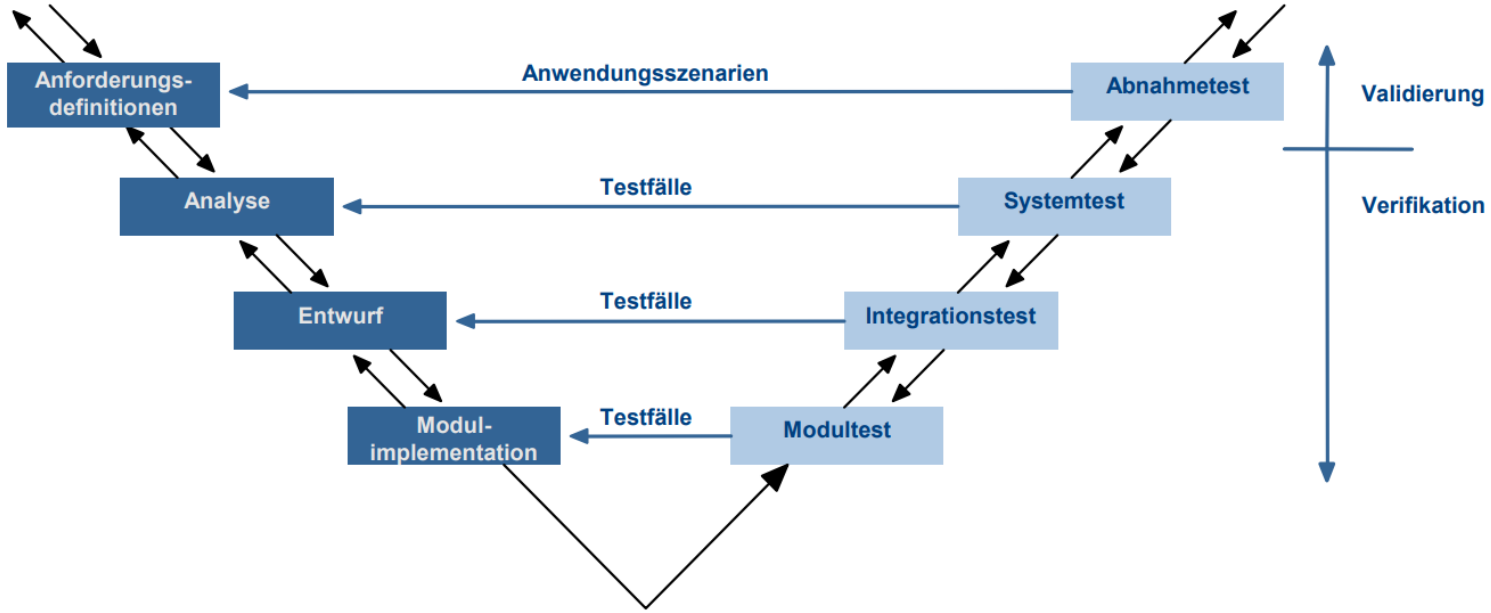
- Beispiel „Seminarorganisation“
  1. Unter Einbeziehung des Auftraggebers/Benutzers wird eine Produktdefinition für alle Anforderungen des Auftraggebers ermittelt.
     - Nach Abnahme des Produktmodells durch den Auftraggeber ist die Definitionsphase abgeschlossen.
  2. In der Entwurfsphase wird ausgehend vom Produktmodell eine Produkt-architektur für das gesamte Produkt entwickelt
     - Sind Anforderungen des Produktmodells nicht realisierbar, wird ein Änderungsdokument erstellt
     - Anschließend beginnt wieder die Entwurfsphase
     - Als Ergebnis der Entwurfsphase wird die Produktarchitektur an die Implementierungsphase übergeben
  3. Die Produktarchitektur wird implementiert
     - Es entsteht das fertige Produkt

DAS WASSERFALL-MODELL

- **Nachteile**
  - Nicht immer sinnvoll
    - alle Entwicklungsschritte in der vollen Breite und vollständig durchzuführen
    - alle Entwicklungsschritte sequentiell durchzuführen
  - Dokumentation wird ggf. wichtiger wird als das entwickelte System
  - Risikofaktoren werden ggf. nicht ausreichend betrachtet, da nur die initial festgelegten Anforderungen bestehen bleiben

DAS V-MODELL

- Erweiterung des Wasserfall-Modells
- Integriert die Qualitätssicherung in das Wasserfall-Modell
- Verifikation der Teilprodukte und Validation des Gesamtprodukts sind Bestandteile des V-Modells
  - Verifikation
    - Überprüfung der Übereinstimmung zwischen einem Software-Produkt und seiner Spezifikation (nicht den Anforderungen!)
  - Validation
    - Eignung bzw. der Wert eines Produktes bezogen auf seinen Einsatzzweck (Anwendungskontext)
- Vorsicht: In der Praxis Begriffe "Verifikation" und „Validierung“ häufig falsch verwendet



DAS PROTOTYPEN-MODELL (1)
PROBLEME TRADITIONELLER PROZESSMODELLE

- Auftraggeber ist oft nicht in der Lage, die Anforderungen an ein neues System explizit und/oder vollständig zu formulieren
- Während der Entwicklung ist oft eine wechselseitige Koordination zwischen Entwicklern und Anwendern sinnvoll
  - Traditionelle Prozessmodelle beenden die Feedback-Schleife zwischen Entwickler und Kunden, wenn die Anforderungen final dokumentiert sind
- Software-Entwicklungsabteilungen ziehen sich nach der Definitions-Phase vom Auftraggeber zurück
  - Präsentation des Ergebnisses erst nach der Fertigstellung

DAS PROTOTYPEN-MODELL
PROBLEME TRADITIONELLER PROZESSMODELLE

- Manchmal gibt es unterschiedliche Lösungsmöglichkeiten.
  - Diese können experimentell erprobt und mit dem Kunden diskutiert werden.
- Die Realisierbarkeit lässt sich theoretisch nicht garantieren
  - Beispiel: Echtzeitanforderungen und nicht-funktionale Anforderungen
- In der Akquisitionsphase muss der Auftraggeber von der prinzipiellen Durchführbarkeit einer Idee oder der Handhabung überzeugt werden

→ Probleme können teilweise durch das Prototypen-Modell gelöst werden

DAS PROTOTYPEN-MODELL
SOFTWARE-PROTOTYP VS. PROTOTYP

- **Unterschiede**
  - Ein Software-Prototyp ist nicht das erste Muster einer großen Serie von Produkten
  - Ein Software-Prototyp zeigt ausgewählte Eigenschaften des Zielproduktes im praktischen Einsatz
- **Gemeinsamkeiten**
  - Anforderungen oder Entwicklungsprobleme klären
  - Diskussionsbasis
  - Entscheidungshilfe
  - Verwendung für experimentelle Zwecke
  - Sammeln von praktischen Erfahrungen

DAS PROTOTYPEN-MODELL

- Unterstützt systematisch die frühzeitige Erstellung ablauffähiger Modelle (Prototypen) des zukünftigen Produkts, um die Umsetzung von Anforderungen und Entwürfen in Software zu demonstrieren und mit ihnen zu experimentieren
- Vorgehensweise
  - Prototyping
- Vier Arten von Prototypen
  - Demonstrationsprototyp
  - Prototyp im eigentlichen Sinne
  - Labormuster
  - Pilotsystem

DAS PROTOTYPEN-MODELL
DEMONSTRATIONSPROTOTYP

- Dient zur Auftragsakquisition
- Soll dem potentiellen Auftraggeber einen ersten Eindruck vermitteln, wie ein Produkt für das vorgesehene Anwendungsgebiet aussehen kann
- In der Regel werden solche Prototypen schnell aufgebaut
  - Rapid Prototyping

DAS PROTOTYPEN-MODELL
BEISPIEL DEFINITION MINIMAL VIABLE PRODUCT (MVP)

Backend

- Ermöglichen des Ausführens EINES festgelegten Workflows (Parametrierung Fahrdynamikmodell)
  - Datenimport
  - Datenbereinigung/-aufbereitung
  - Datentransformation/-vorbereitung (Feature Engineering)
  - Modelltraining, Modellevaluation, Visualisierung Endergebnis

Frontend

- Webanwendung ohne Rollen, Visualisierung der Einzelschritte und Trainingsergebnisse
- Vereinfachte Version des Wizard/Assistenten, der durch die einzelnen Schritte des Workflows führt

Sonstiges

- Einzelne Schritte des CRISP-Prozess in Code implementieren, die für Use Case wichtig sind
- Dedizierter Server für Hintergrundaufgaben (Modelltraining, Datentransformation etc.)
- Mehraufwand wie Laden von Konfigurationsdateien für MVP nicht notwendig

DAS PROTOTYPEN-MODELL
PROTOTYP IM EIGENTLICHEN SINNE

- Wird parallel zur Modellierung des Anwendungskontext erstellt
- Soll Aspekte der Benutzungsschnittstelle oder Teile der Funktionalität veranschaulichen
- Trägt dazu bei, den Anwendungskontext zu analysieren
- Provisorisches, ablauffähiges Software-System

DAS PROTOTYPEN-MODELL
LABORMUSTER

- Soll konstruktionsbezogene Fragen und Alternativen beantworten
- Demonstriert die technische Umsetzbarkeit des Produktmodells
- Nicht für Endbenutzer bestimmt
- Sollte technisch mit dem späteren Produkt vergleichbar sein

DAS PROTOTYPEN-MODELL
PILOTSYSTEM

- Ist Kern eines Produkts
- Unterscheidung zwischen dem Prototyp und dem Produkt verschwindet
- Pilotsystem ist für die Benutzung in der Einsatzumgebung entworfen und nicht nur unter Laborbedingungen

DAS PROTOTYPEN-MODELL
BEWERTUNG

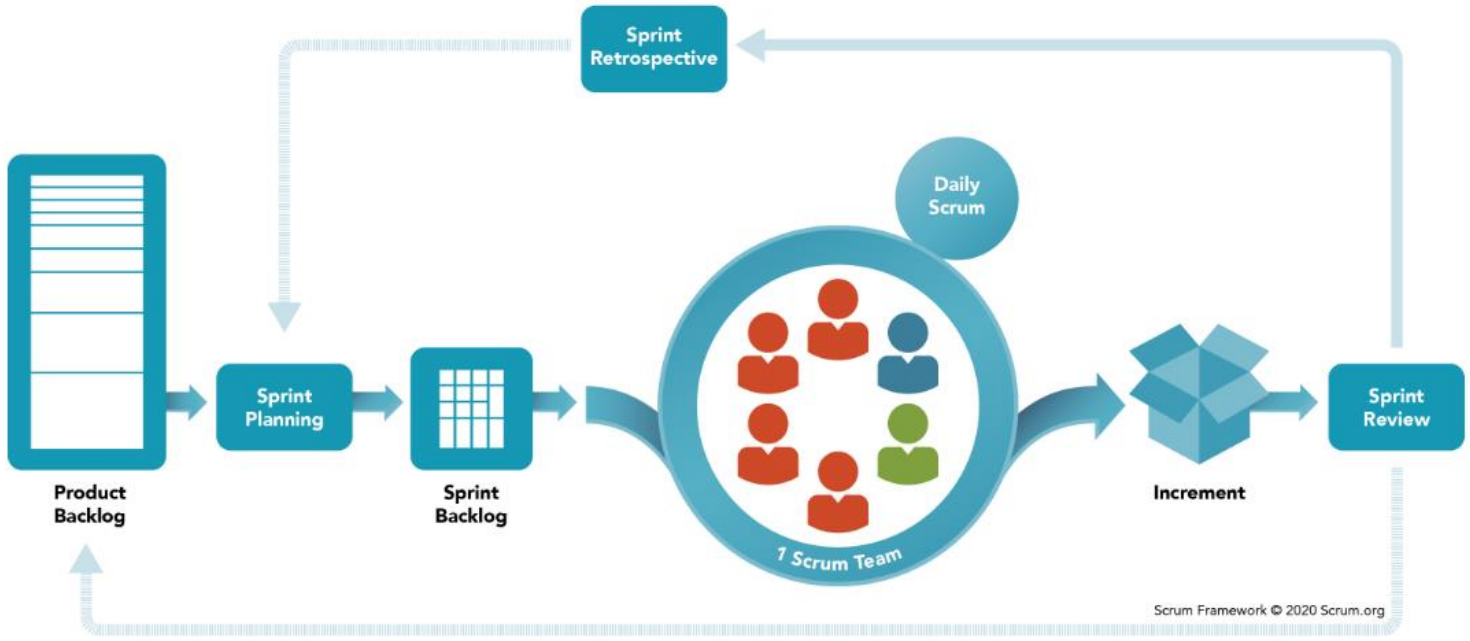
- Reduzierung des Entwicklungsrisikos durch frühzeitigen Einsatz von Prototypen
- Prototypen können sinnvoll in andere Prozessmodelle integriert werden.
- Prototypen können heute durch geeignete Werkzeuge schnell erstellt werden.
- Prototyping verbessert die Planung von Software-Entwicklungen.
- Labormuster fördern die Kreativität für Lösungsalternativen.
- Starke Rückkopplung mit dem Endbenutzer und dem Auftraggeber
- Höherer Entwicklungsaufwand, da Prototypen zusätzlich erstellt werden
- Gefahr, dass ein „Wegwerf“-Prototyp Teil des Endprodukts wird.
- Verträge für die Software-Erstellung berücksichtigen noch nicht das Prototypen-Modell.
- Prototypen werden oft als Ersatz für die fehlende Dokumentation angesehen.
- Die Beschränkungen und Grenzen von Prototypen sind oft unbekannt.

DAS PROTOTYPEN-MODELL
VORAUSSETZUNGEN

- Ausreichendes Wissen über das Anwendungsgebiet muss vorhanden sein.
- Nur auf der Basis schriftlicher Dokumente kann kein Prototyp erstellt werden.
  → Die Entwickler müssen Zugang zu den Benutzern haben.
- Die Endbenutzer müssen am Prototypingprozess beteiligt werden.
- Die Benutzerbeteiligung ersetzt nicht die kreativen Ideen der Entwickler.
- Alle beteiligten Personengruppen müssen in direktem Kontakt stehen.
- Prototypen müssen dokumentiert werden.
- Die Vorgehensweise hängt von der untersuchten Fragestellung ab.
- Geeignete Werkzeuge müssen verfügbar sein.
  → Motto: „Redo until Right“

SCRUM

- Scrum ist ein Framework, das Menschen, Teams und Organisationen hilft, durch adaptive Lösungen für komplexe Probleme Werte zu schaffen.
- Scrum erwartet, dass ein Scrum Master eine Umgebung fördert, in der:
  - Ein Product Owner ordnet die Arbeit für ein komplexes Problem in einem Product Backlog an.
  - Das Scrum Team setzt eine Auswahl der Arbeit während eines Sprints in ein Wertzuwachs (Increment) um.
  - Das Scrum Team und seine Stakeholder prüfen die Ergebnisse und passen sie für den nächsten Sprint an.
  - Wiederholen



https://www.scrum.org/resources/what-is-scrum/

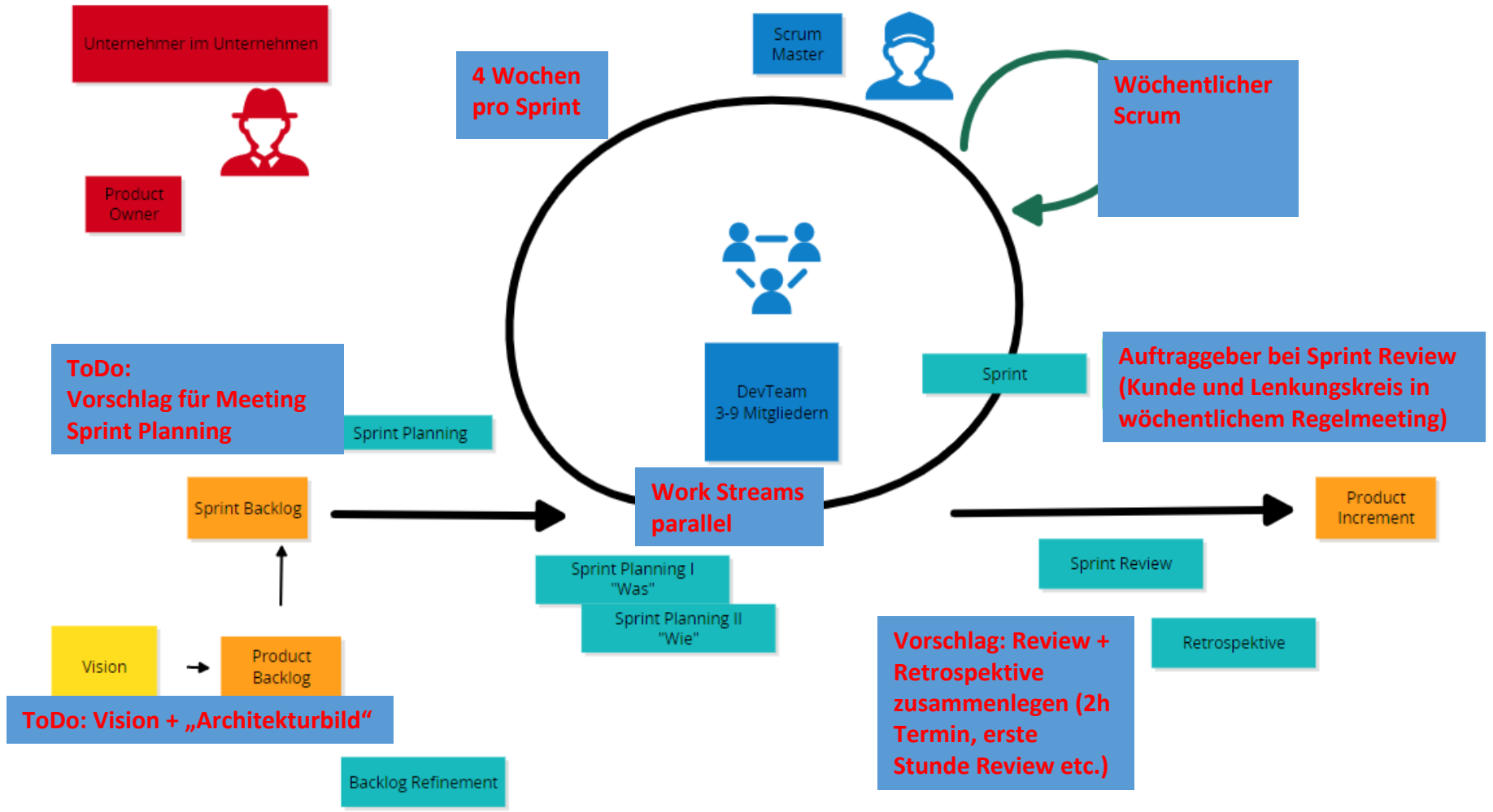
SCRUM GLOSSARY

Element	Beschreibung
Daily Scrum	Scrum Event, ein 15-minütiges Timebox-Event, das jeden Tag für die Entwickler stattfindet. Das Daily Scrum findet an jedem Tag des Sprints statt. Dabei planen die Entwickler die Arbeit für die nächsten 24 Stunden. Dies optimiert die Zusammenarbeit und die Leistung des Teams, indem die Arbeit seit dem letzten Daily Scrum überprüft und die anstehende Arbeit im Sprint prognostiziert wird. Das Daily Scrum wird jeden Tag zur gleichen Zeit und am gleichen Ort abgehalten, um die Komplexität zu reduzieren.
Increment	Scrum-Artefakt, das die vollständige und wertvolle Arbeit definiert, die von den Entwicklern während eines Sprints produziert wird. Die Summe aller Inkremente bildet ein Produkt.
Product Backlog	Scrum-Artefakt, das die vollständige und wertvolle Arbeit definiert, die von den Entwicklern während eines Sprints produziert wird. Die Summe aller Inkremente bildet ein Produkt
Sprint Backlog	Scrum-Artefakt, das einen Überblick über die Entwicklungsarbeit zur Erreichung des Sprint-Ziels gibt, typischerweise eine Vorhersage der Funktionalität und der zur Bereitstellung dieser Funktionalität erforderlichen Arbeit. Wird von den Entwicklern verwaltet.
Sprint Planning	Scrum Event, das auf 8 Stunden oder weniger begrenzt ist, um einen Sprint zu starten. Es dient dem Scrum-Team dazu, die Arbeit aus dem Product Backlog zu inspizieren, die als Nächstes erledigt werden sollte, und diese Arbeit in das Sprint Backlog zu integrieren.

SCRUM GLOSSARY

Element	Beschreibung
Sprint Retrospective	Scrum Event, das auf ein Zeitfenster von 3 Stunden oder weniger festgelegt ist, um einen Sprint zu beenden. Es dient dem Scrum Team zur Überprüfung des vergangenen Sprints und zur Planung von Verbesserungen, die in zukünftigen Sprints umgesetzt werden sollen.
Sprint Review	Scrum Event, das auf eine Timebox von 4 Stunden oder weniger festgelegt ist, um die Entwicklungsarbeit eines Sprints abzuschließen. Es dient dem Scrum Team und den Stakeholdern dazu, das aus dem Sprint resultierende Produktinkrement zu inspizieren, die Auswirkungen der geleisteten Arbeit auf den Gesamtfortschritt in Richtung des Produktziels zu bewerten und das Product Backlog zu aktualisieren, um den Wert der nächsten Periode zu maximieren.

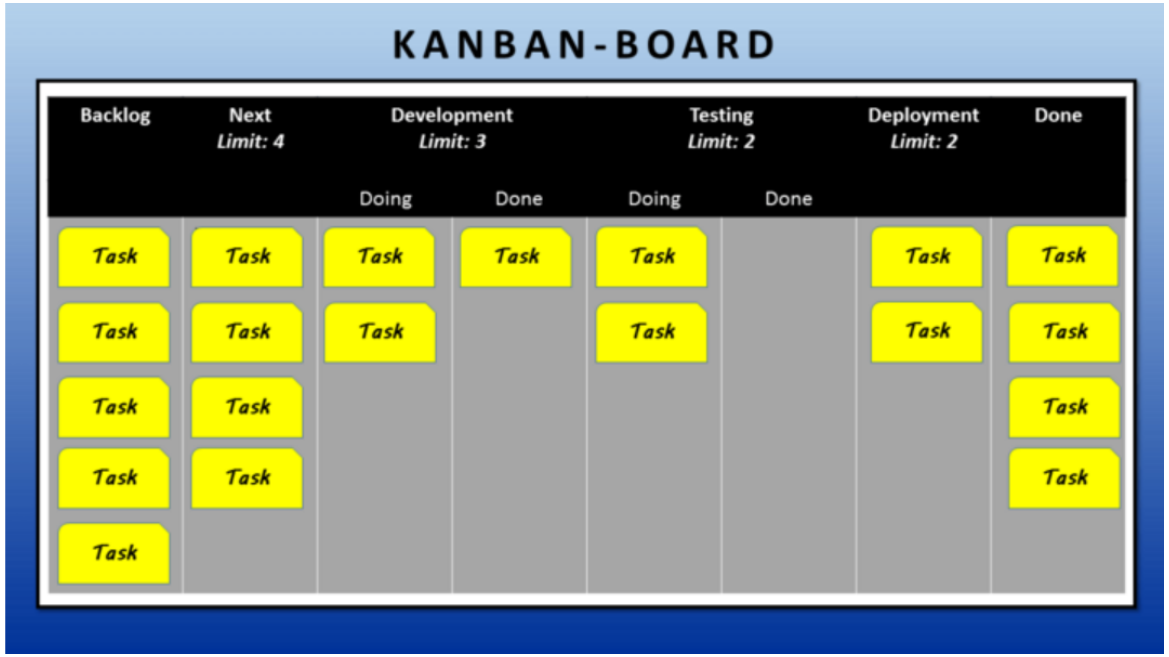
SCRUM-PRAXISBEISPIEL BEI MEHREREN PARALLEN PROJEKTEN DES DEVTEAMS (NEGATIVBEISPIEL)



KANBAN

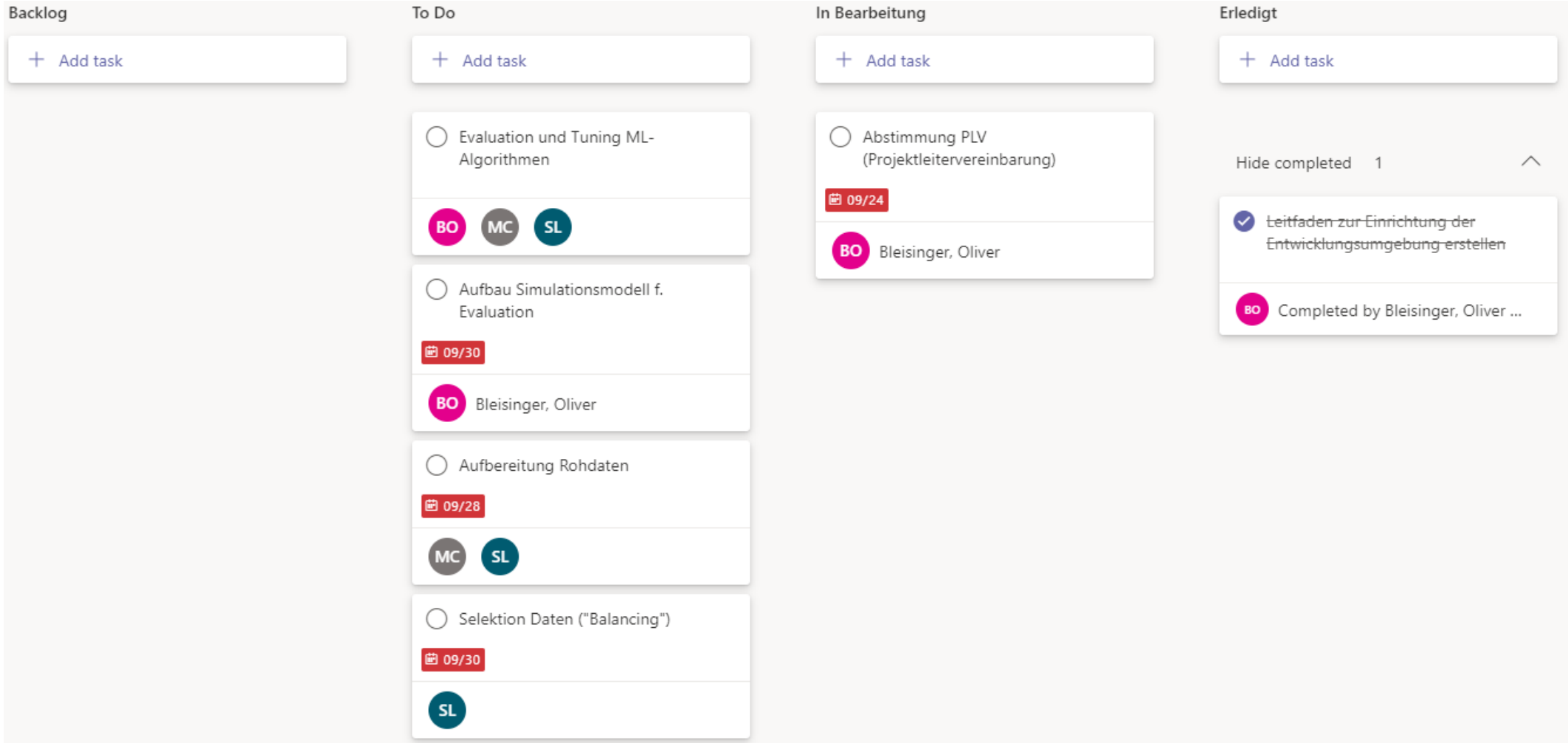
Insgesamt lassen sich sechs verschiedene Praktiken von Kanban ausmachen:

- **Visualisierung**: Das Kanban-Board ist eine Visualisierung der Arbeitsabläufe. Die Gestaltung selbst bleibt aber relativ offen. Wichtig ist nur, dass Stationen klar sind und für jede Spalte das entsprechende Limit angezeigt wird.
- **Limitierung**: Jede Spalte darf nur eine maximale Anzahl an Aufträgen enthalten. Erst wenn eine Auftragskarte weiter nach rechts wandert, darf sich das Team eine neue Karte von links nehmen. Dies führt zwangsläufig zu einem effizienteren Workflow.
- **Management**: Während des Arbeitsprozesses kann es zu Blockaden und Engpässen kommen. In solchen Situationen ist es notwendig, den Fokus des Teams darauf zu legen, diese Störungen aus dem Weg zu schaffen. Außerdem kann die Beobachtung des Workflows dafür sorgen, Kapazitäten langfristig korrekt zu verteilen.
- **Regulierung**: Explizite Prozessregeln sind dafür gedacht, die Arbeitsabläufe transparenter und klarer zu gestalten. Zu solchen Regeln gehört z. B. die Festlegung der Limits, aber auch eine Definition, ab wann eine Aufgabe als erledigt gilt. Prozessregeln müssen ebenfalls ein sichtbarer und veränderbarer Teil des Kanban-Boards sein.
- **Feedback**: Rückmeldungen sind ein notwendiger Teil von Arbeitsabläufen, denn nur so lassen sich diese verbessern. Dafür sind regelmäßige Meetings vorgesehen, sogenannte Kadenzen. Anders als Scrum gibt Kanban aber kein starres Gerüst für solche Treffen.
- **Kaizen**: Prozesse im Team sollen mit Kanban kontinuierlich verbessert werden. Die Theorie geht somit davon aus, dass man kein Optimum erreichen kann, sondern dauerhaft an Verbesserungen arbeitet.

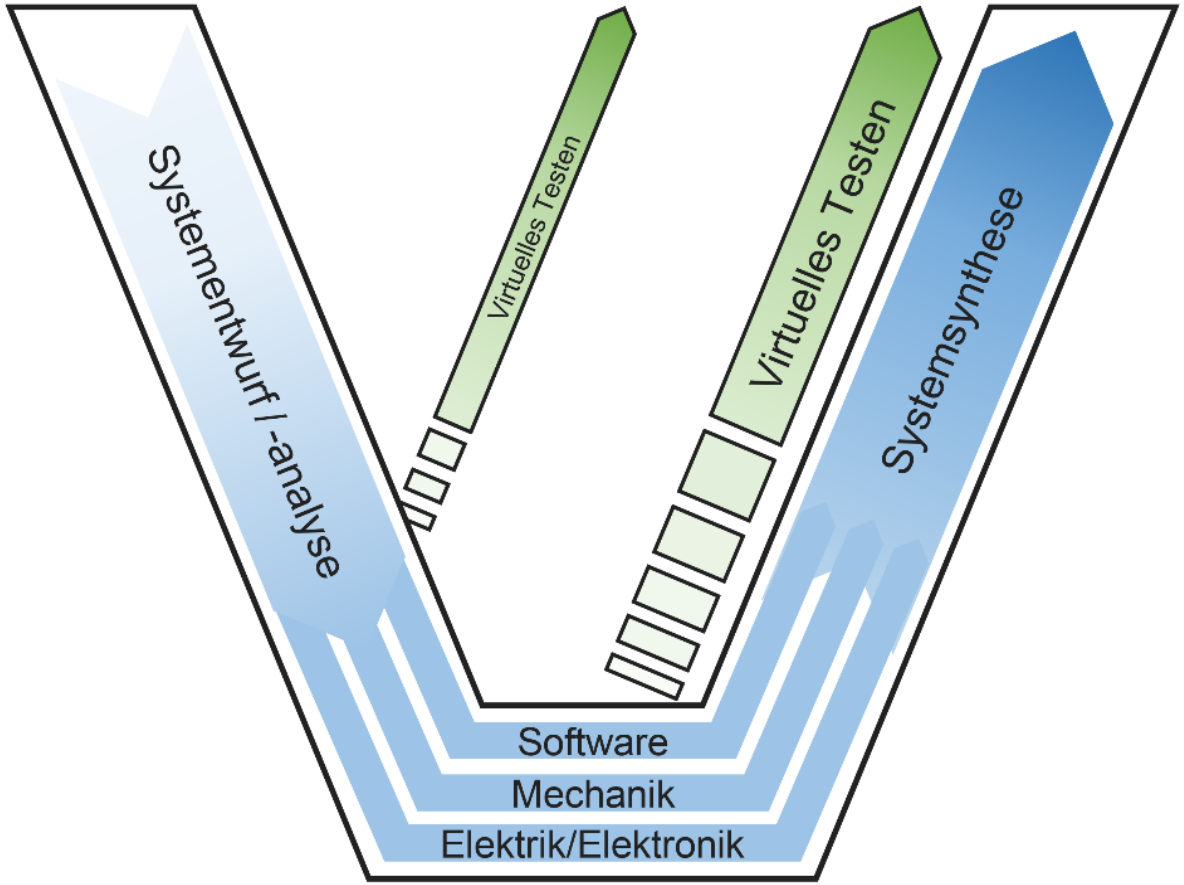


https://www.ionos.de/digitalguide/websites/web-entwicklung/kanban/

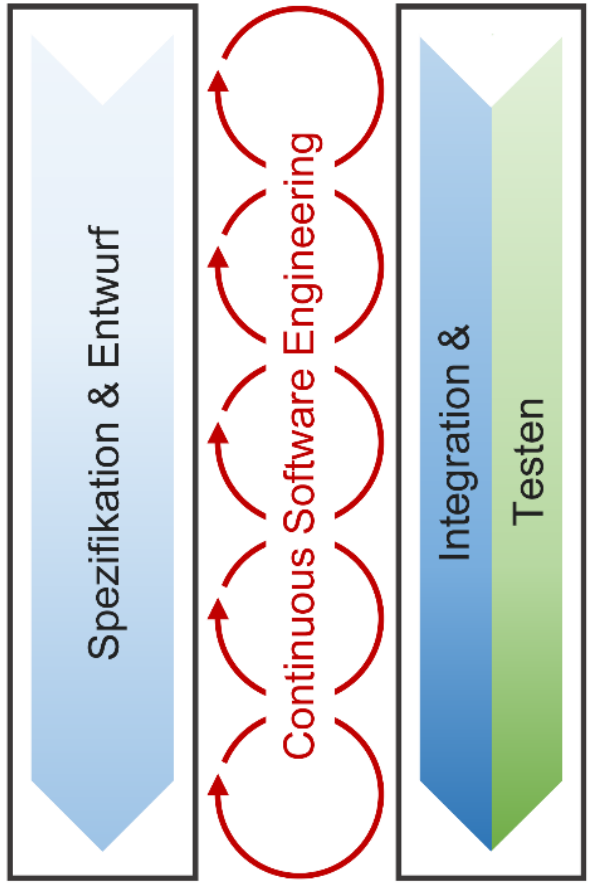
KANBAN-PRAXISBEISPIEL MIT MS TEAMS ALS TOOL OHNE WIP-LIMITS



VERBINDUNG DER DISZIPLINEN SOFTWARE UND SYSTEMS ENGINEERING
WEITERE VORGEHENSMODELLE

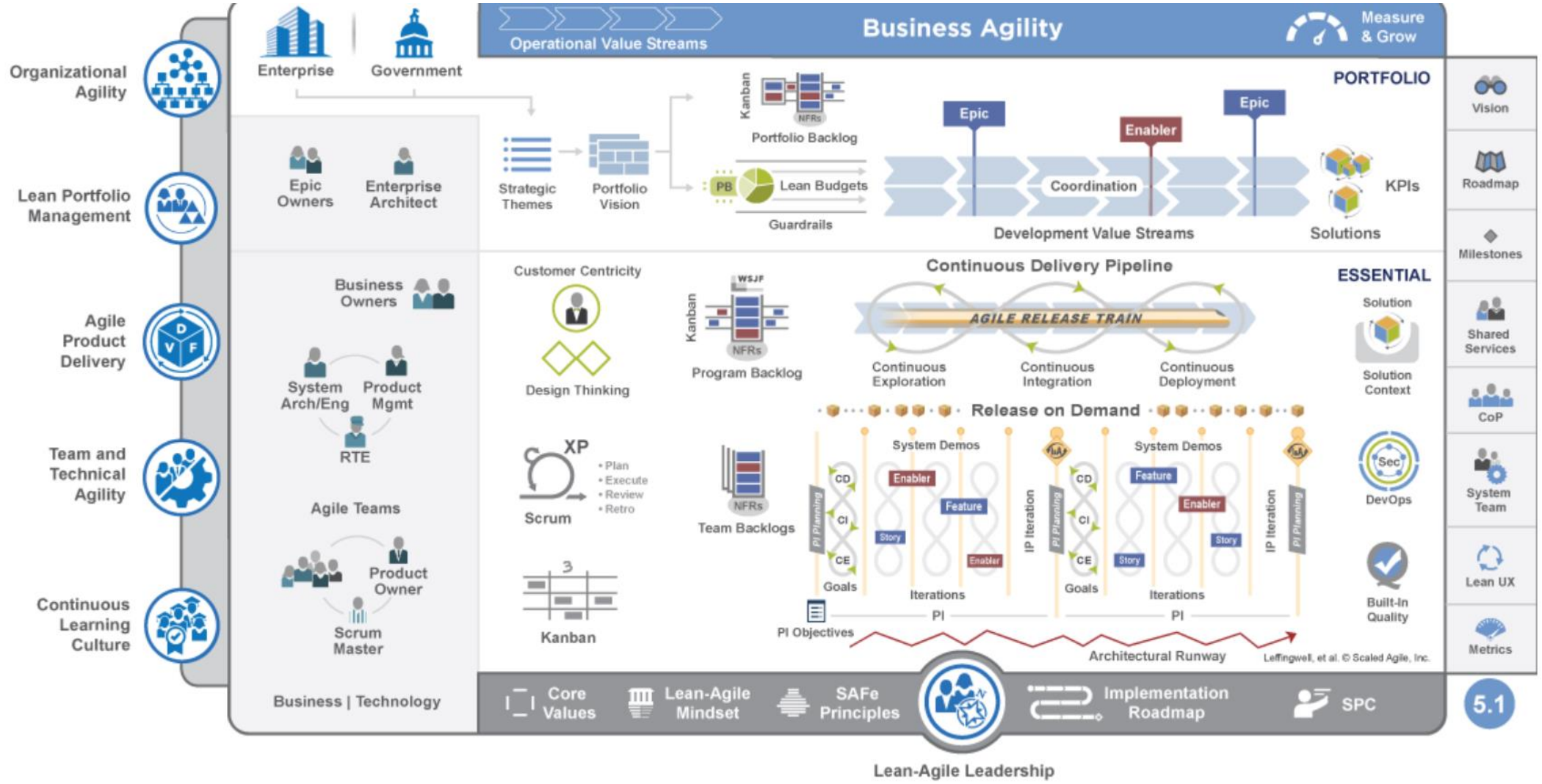


Quelle: V-Modell in Anlehnung an Martin Eigner et al. (2014) – "Modellbasierte virtuelle Produktentwicklung"



Quelle: II-Modell in Anlehnung an Pablo Antonino et al. (2019) – "Enabling continuous software engineering for embedded systems architectures with virtual prototypes"

SAFE – SCALED AGILE FRAMEWORK



SAFE
AUSBAUSTUFEN

- Essential SAFe
- Large Solution SAFe
- Portfolie SAFe
- Full SAFe

SAFE
ÜBERSICHT

Ausbaustufe	Ebenen	Ziel/Anforderung
Essential	Team und Programmebene	Geeignet für Unternehmen, die schnell starten wollen.
Portfolio	Team-, Programm- und Portfolioebene	Geeignet für Unternehmen, die ihre Programme mit der Unternehmensstrategie abgleichen wollen
Large Solution	Team-, Programm-, Portfolio-, und Large-Solution-Ebene	Für Unternehmen mit großem Skalierungsbedarf
Full	Team-, Programm-, Portfolio-, und Large-Solution-Ebene	Für Global Player mit großem Skalierungsbedarf – sowohl für Projekte als auch Mitarbeiter

SAFE
ESSENTIAL SAFE

- Essential SAFe enthält den minimalen Satz von Rollen, Ereignissen und Artefakten, die erforderlich sind, um als Team von Agilen Teams kontinuierlich Geschäftslösungen über einen Agile Release Train (ART) zu liefern
- **Drei Kernkompetenzen:**
  - **Team- und technische Agilität:** Beschreibt die entscheidenden Fähigkeiten und Lean-Agile-Prinzipien und -Praktiken, die leistungsstarke agile Teams und Teams agiler Teams einsetzen, um hochwertige Lösungen für ihre Kunden zu entwickeln.
  - **Agile Produktbereitstellung:** ist ein kundenorientierter Ansatz zur Definition, Erstellung und Freigabe eines kontinuierlichen Flusses wertvoller Produkte und Dienstleistungen für Kunden und Benutzer.
  - **Lean-Agile Leadership:** Beschreibt, wie Lean-Agile Leader organisatorische Veränderungen und operative Exzellenz vorantreiben und aufrechterhalten, indem sie Einzelpersonen und Teams befähigen, ihr höchstes Potenzial zu erreichen.

SAFE
LARGE SOLUTION SAFE

- Large Solution SAFe beschreibt zusätzliche Rollen, Praktiken und Anleitungen zum Aufbau und zur Weiterentwicklung der weltweit größten Anwendungen, Netzwerke und cyber-physischen Systeme.
- **Die Large Solution SAFe-Konfiguration umfasst die folgenden Konstrukte:**
  - Die grundlegende SAFe-Konfiguration
  - Eine zusätzliche Kompetenz, Enterprise Solution Delivery, die beschreibt, wie man Lean-Agile-Prinzipien und -Praktiken auf die Spezifikation, die Entwicklung, den Einsatz, den Betrieb und die Weiterentwicklung der weltweit größten und anspruchsvollsten Softwareanwendungen, Netzwerke und cyber-physischen Systeme anwendet.
  - Die großen Rollen, Artefakte und Ereignisse auf der Lösungsebene
  - Die vollständige übergreifende Palette
  - Eine Verbindung zu dem Unternehmen oder der staatlichen Einrichtung, die die Lösung unterstützt

SAFE
PORTFOLIO SAFE

- Portfolio SAFe richtet die Strategie auf die Ausführung aus und organisiert die Lösungsentwicklung um den Wertfluss durch einen oder mehrere Wertströme.
- Die Portfoliokonfiguration, die Essential SAFe enthält, ist die kleinste Konfiguration, die zur Erreichung von Business Agility eingesetzt werden kann, und besteht aus den folgenden Komponenten:
  - **Drei zusätzliche Kernkompetenzen:**
    - Die Kompetenz "Schlankes Portfoliomanagement" stimmt Strategie und Ausführung aufeinander ab, indem sie schlanke und systemorientierte Ansätze auf Strategie und Investitionsfinanzierung, agile Portfoliobetriebe und Governance anwendet
    - Die Kompetenz "Kontinuierliche Lernkultur" beschreibt eine Reihe von Werten und Praktiken, die den Einzelnen - und das Unternehmen als Ganzes - dazu ermutigen, Wissen kontinuierlich zu erweitern
    - Die Kompetenz „Organisatorische Agilität“ beschreibt, wie schlank denkende Menschen und agile Teams ihre Geschäftsprozesse optimieren, die Strategie mit klaren und entschlossenen neuen Verpflichtungen weiterentwickeln und die Organisation bei Bedarf schnell anpassen, um neue Gelegenheiten zu nutzen
  - Die Rollen, Ereignisse und Artefakte auf Portfolio-Ebene
  - Die vollständige übergreifende Palette

SAFE
FULL SAFE

- Beinhaltet alle Stufen
- Für globale Player mit großen Skalierungsbedarf

TEST-DRIVEN DEVELOPMENT (TDD)

- Testgetriebene Entwicklung (TDD) ist eine Philosophie und Praxis, die die Erstellung und Ausführung von Tests vor der Implementierung des Codes oder einer Komponente eines Systems beinhaltet.
- Durch die Validierung anhand einer Reihe von vereinbarten Tests verbessert TDD - eine agile Testpraxis - die Systemergebnisse, indem es sicherstellt, dass die Systemimplementierung die Anforderungen erfüllt.
- TDD ist zusammen mit Behavior-Driven Development (BDD) Teil des "test-first"-Ansatzes, mit dem Built-in Quality erreicht wird.
- Das Schreiben von Tests zuerst schafft ein ausgewogeneres Testportfolio mit vielen schnellen, automatisierten Entwicklungstests und weniger langsamen, manuellen End-to-End-Tests.

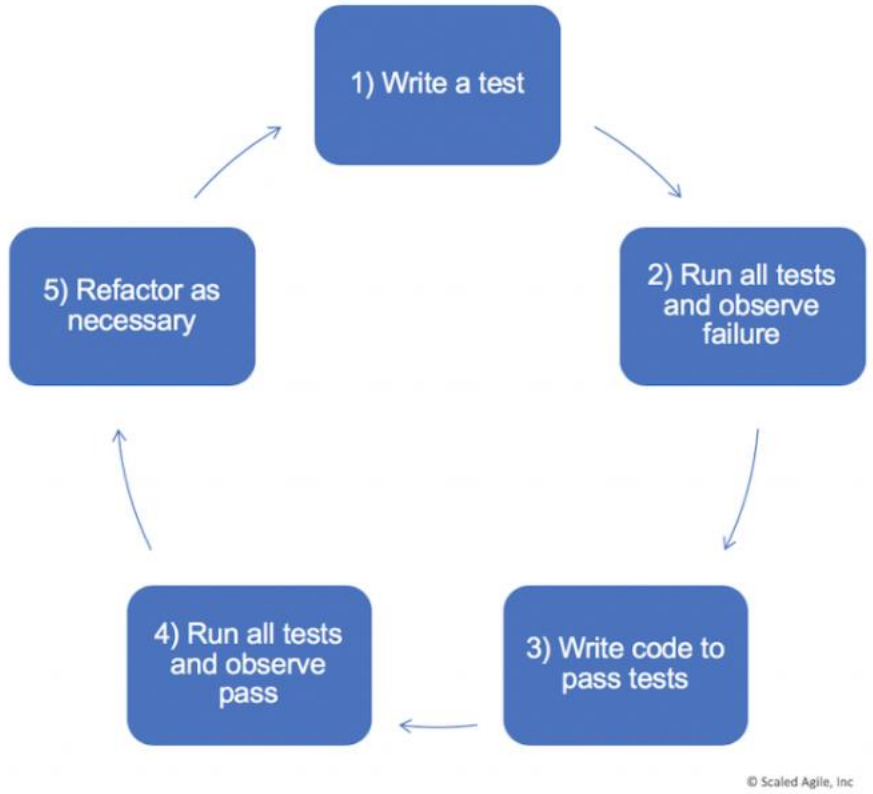


Figure 1. The Test-Driven Development process

https://www.scaledagileframework.com/test-driven-development/

Prozessmodell	Primäres Ziel	Antreibendes Moment	Benutzerbeteiligung	Charakteristika
Wasserfall	Minimales Management	Dokumente	Gering	Sequentiell, volle Breite
V-Modell	Maximale Qualität	Dokumente	Gering	Sequentiell, volle Breite, V&V
Prototypen	Risikominimierung	Code	Hoch	Nur Teilsysteme (horizontal pder vertikal)
Scrum/Kanban/SAFe	Höchstmögliche Transparenz, Überprüfung und Anpassung		Hoch	Pragmatisch, einfach, wenige involvierte Rollen
Test-Driven Development	Bestmöglichen Code schreiben	Code	Gering	Sequentiell, testbasiert

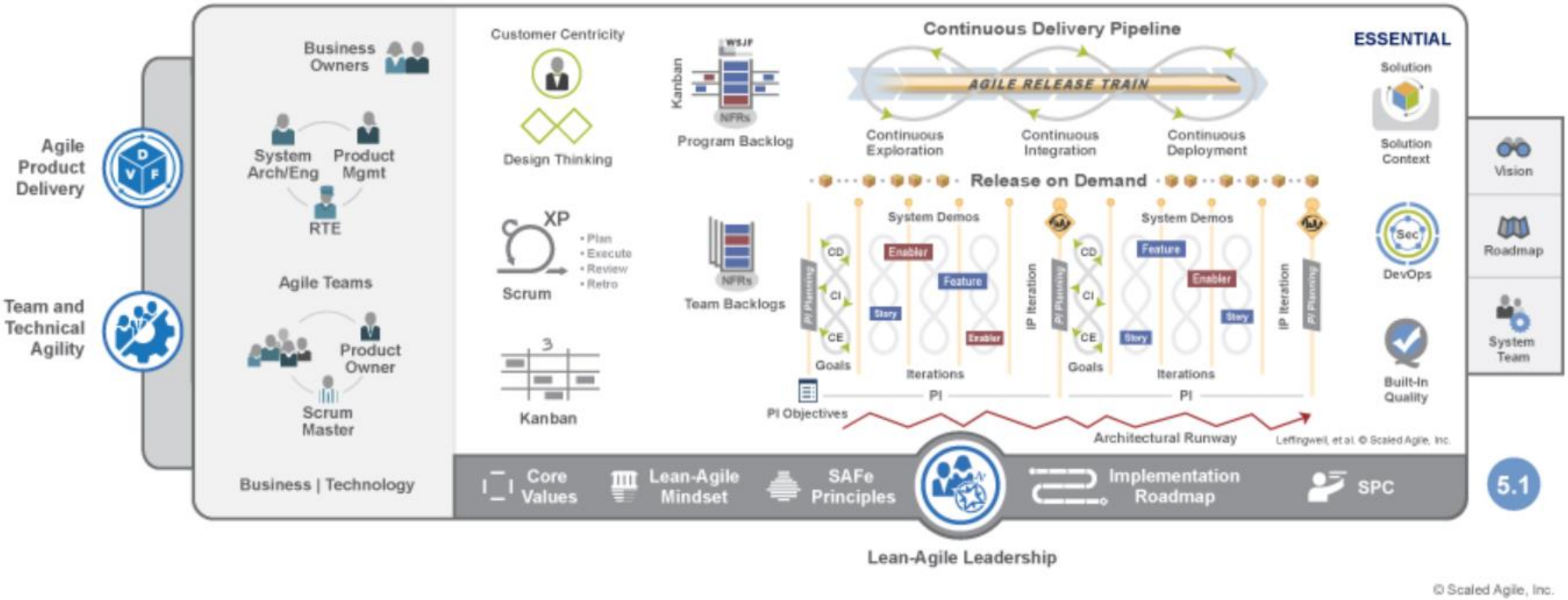


Figure 1. Essential SAFe

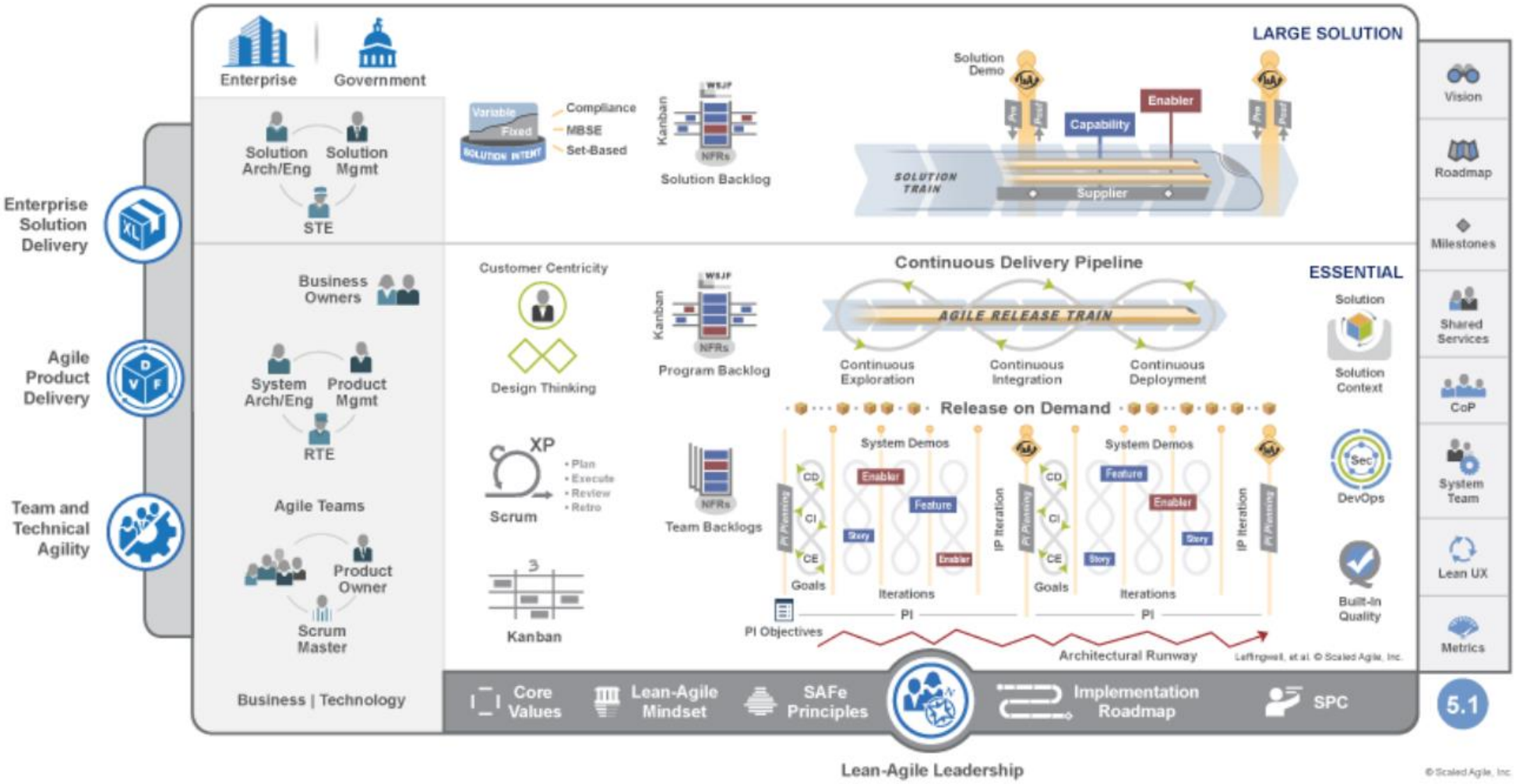


Figure 1. Large Solution SAFe

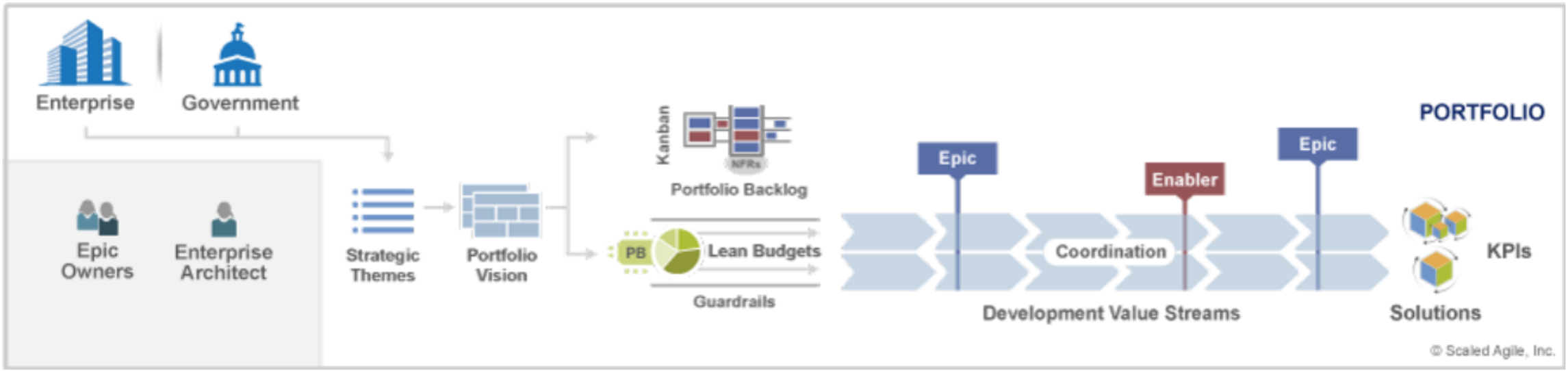


Figure 2. Portfolio level

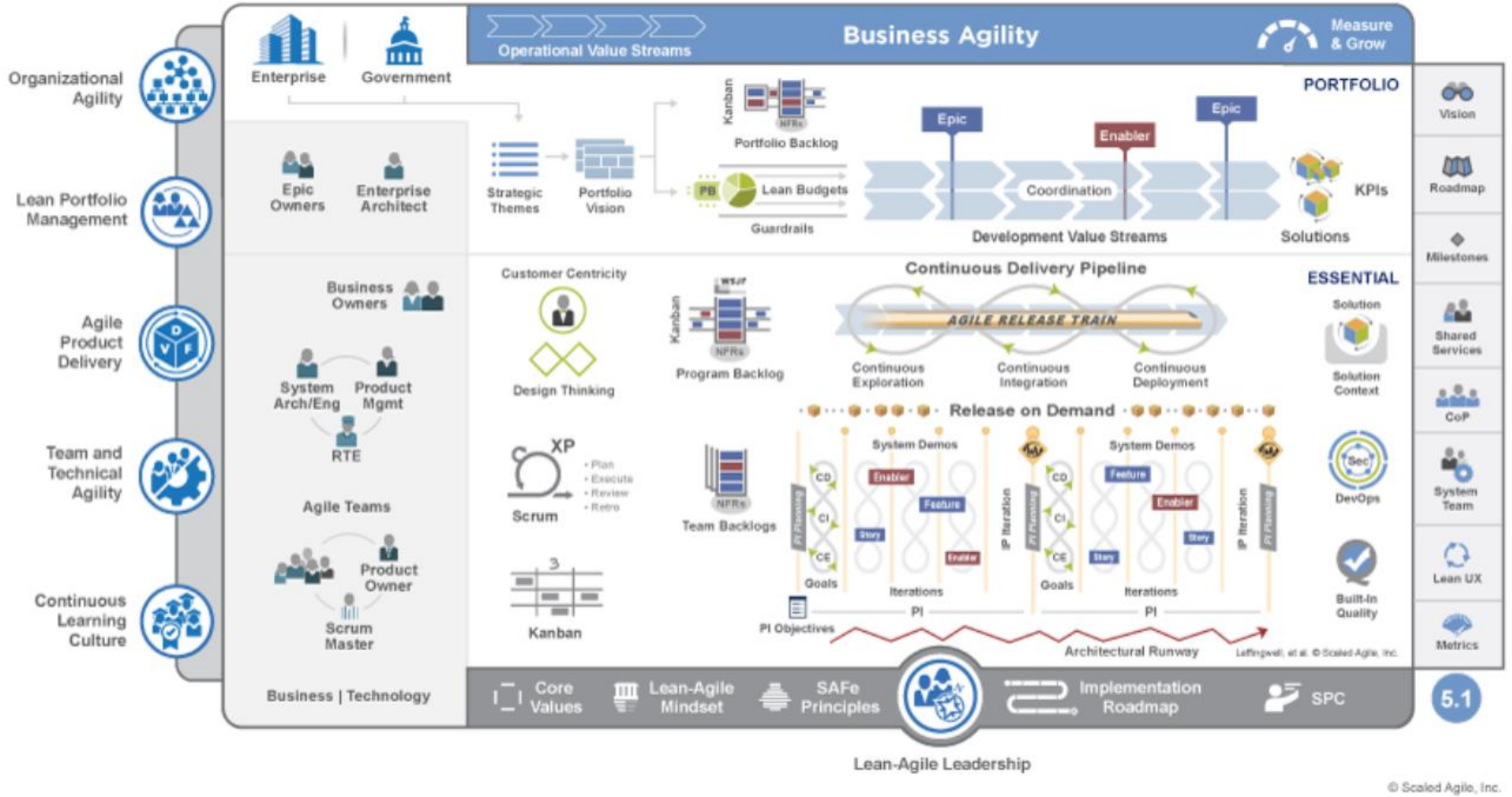


Figure 1. SAFe Portfolio

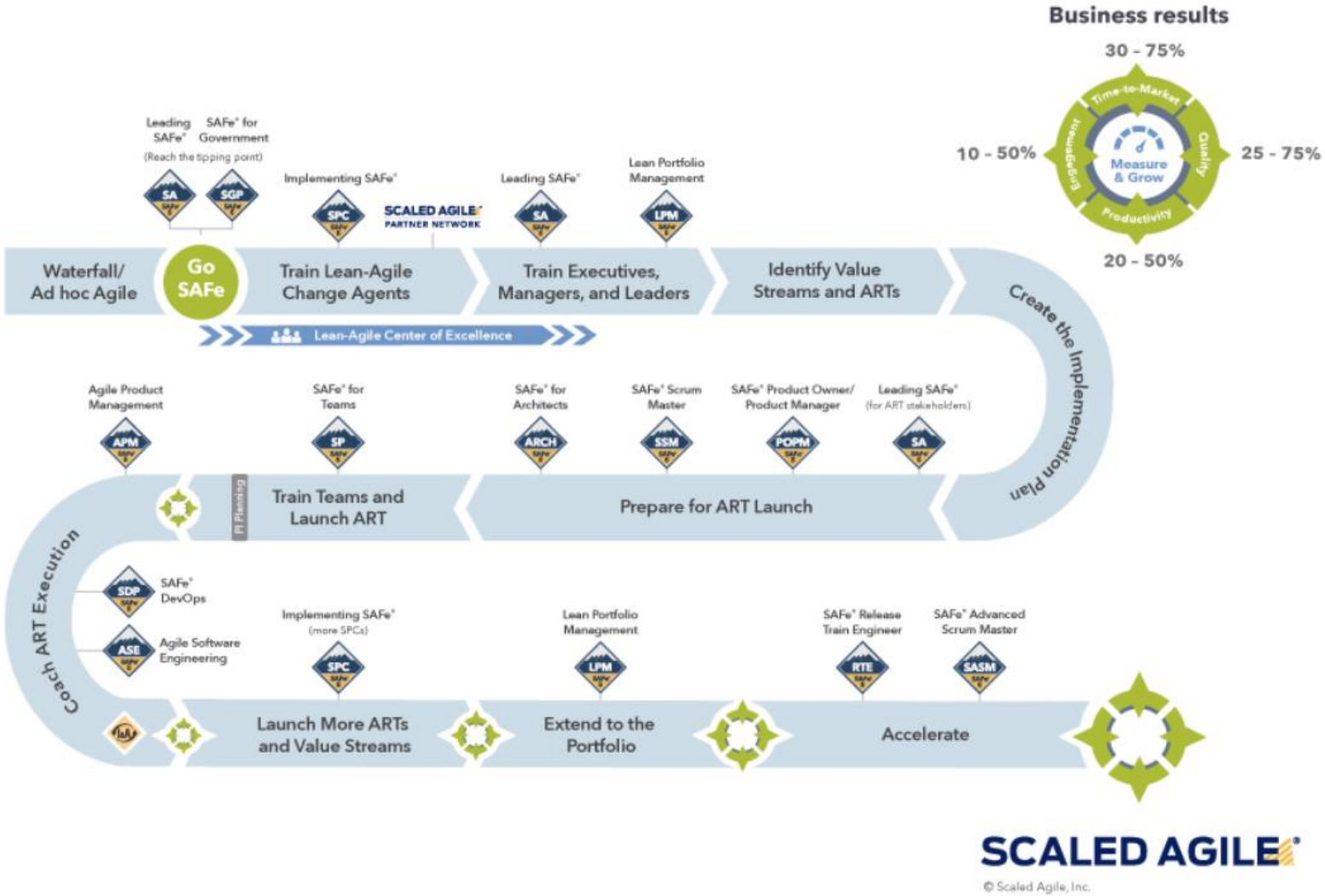


Figure 1. SAFe Implementation Roadmap