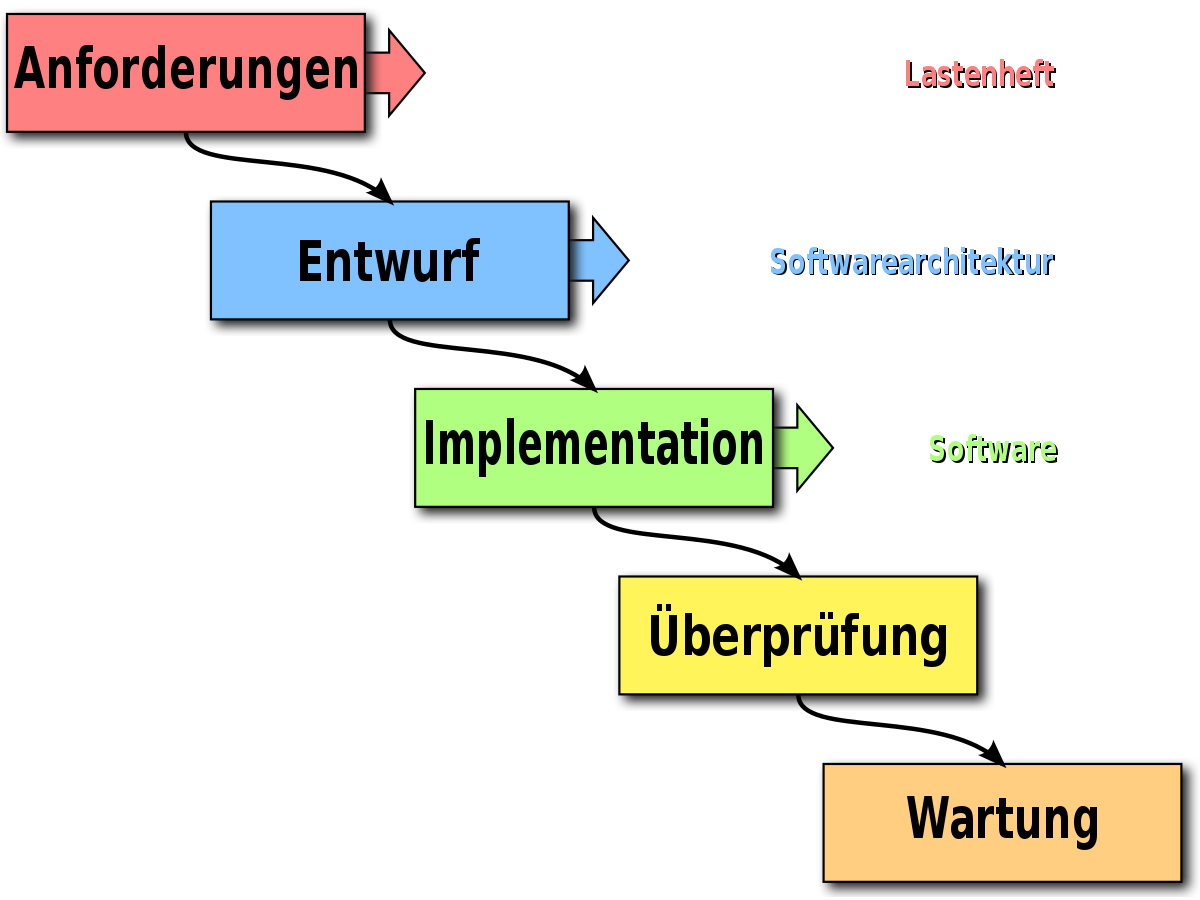
**Wasserfallmodel im Vergleich zu agilen Methoden**

Beginnen wir mit den Grundlagen, zum Beispiel mit der Frage, was **Agile** so einzigartig macht.

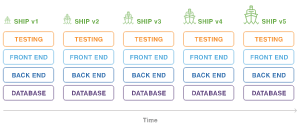
Bei herkömmlichen Projektmanagementverfahren wie dem Wasserfallmodell erfolgt die Entwicklung in Phasen. Unten sieht man eine Darstellung eines Standardprojekts im Wasserfallmodell. Bei dieser Art der Produktentwicklung setzen die Teams alles auf ein einziges, **hochriskantes "Urknall"-Release**. Wenn ein Projekt eine Phase durchlaufen hat, kann das Team nur schwer zu dieser zurückkehren, da sich die nächste Phase sehr schnell anschließt.



(Bildquelle: Wikipedia)

Herkömmliche Projektmanagementverfahren verursachen oft Engpässe, durch die das Projekt erst weitergeführt werden kann, wenn ein blockierendes Problem behoben ist. Um das Ganze noch zu verschlimmern, kann der Endkunde erst mit dem Produkt interagieren, wenn es komplett abgeschlossen ist. Daher werden wichtige Probleme im Produktdesign und Code erst nach dem Release entdeckt.

Vergleichen wir das mit dem agilen Ansatz, der auf einem iterativen Entwicklungsansatz mit regelmäßigen Feedbackintervallen basiert. Durch diese Iterationen kann das Team zu anderen Bereichen des Projekts umgeleitet werden (und dort produktiv sein), während ein blockierendes Problem gelöst wird.



Quelle: [https://www.atlassian.com](https://www.atlassian.com/de/agile/project-management/program)

**Engpässe vermeiden in Iterationen planen**

Neben der Vermeidung von Engpässen sorgen Iterationen dafür, dass man während der Entwicklung mit dem Produkt interagieren kann.

Damit erhält das Team wiederum ständige Gelegenheiten, um sich zu entwickeln, Lösungen bereitzustellen, zu lernen und sich anzupassen. Man ist auf Anforderungsänderungen als Team besser vorbereitet

**Zwei Punkte sind zu verinnerlichen um mit agiler Software-Entwicklung erfolgreich zu sein**

* Der Schwerpunkt des Produktinhabers liegt auf der Optimierung der vom Entwicklerteam erzielten Arbeitsergebnisse. Das Entwicklerteam verlässt sich darauf, dass der Produktinhaber den wichtigsten Aufgaben die höchste Priorität zuordnet.
* Das Entwicklerteam kann nur dann Aufgaben annehmen, wenn entsprechende Kapazitäten vorhanden sind. Der Produktinhaber vergibt keine Aufgaben an das Team und verpflichtet es auch nicht zu beliebigen Abgabeterminen. Das Entwicklerteam entnimmt Aufgaben aus dem Backlog, sobald es neue Arbeit annehmen kann.

**Wichtige Bestandteile agiler Methoden zur iterativen Abhandlung**

### **Roadmaps (der Projektplan)**

In einer **Roadmap** wird dargestellt, wie sich ein Produkt oder eine Lösung über die Zeit entwickelt. Roadmaps bestehen aus Initiativen, die große Funktionalitätsbereiche sind, und enthalten Zeitabläufe, mit denen die Verfügbarkeit eines Features kommuniziert wird. Änderungen der Roadmap während der Weiterentwicklung des Programms sind unabhängig vom Umfang der Änderung akzeptabel. Der Schwerpunkt der Roadmap sollte auf den aktuellen Marktbedingungen und langfristigen Zielen liegen.

## **Anforderungen (Features und Tasks)**

Jede Anforderung in der Roadmap wird in eine Sammlung von Anforderungen unterteilt. Bei Agile sind die Anforderungen nur kurze Beschreibungen der erforderlichen Funktionalität, nicht die bei herkömmlichen Projekten üblichen 100-Seiten-Dokumente. Sie werden über die Zeit weiterentwickelt und reflektieren das gemeinsame Verständnis des Teams für den Kunden und das gewünschte Produkt. Agile-Anforderungen bleiben kurz und knapp formuliert, solange die Teammitglieder durch fortlaufende Gespräche und Zusammenarbeit ein gemeinsames Verständnis entwickeln. Erst wenn die Implementierung beginnen soll, werden sie detailliert ausgearbeitet.

## **Backlog (Sammlung der Task, ...)**

Das **Backlog** legt die Prioritäten für das Agile-Programm fest. Das Team trägt alle Aufgabenelemente im Backlog ein: neue **Features**, **Bugs**, **Verbesserungen**, technische oder architekturbezogene **Tasks** usw. **Der Produktinhaber priorisiert** die Aufgaben im Backlog für das Entwicklerteam. Die Entwickler nutzen das priorisierte Backlog dann als zentrale Informationsquelle für alle anstehenden Aufgaben.

## **Agile Metriken**

Agile-Teams profitieren von Metriken. Work-in-Progress-Grenzen (WIP-Grenzen) sorgen dafür, dass Team und Unternehmen immer die Aufgaben mit der höchsten Priorität bearbeiten. Diagramme wie Burndown- und Kontrollcharts unterstützen das Team, den Lieferrhythmus zu prognostizieren. Engpässe können mithilfe eines kontinuierlichen Flussdiagramms erkannt werden. Diese Metriken und Artefakte sorgen dafür, dass sich alle auf die großen Ziele konzentrieren und das Team geplante Aufgaben mit größerer Zuversicht fertigstellen kann.

## **Agile Methoden basieren auf Vertrauen**

Ein agiles Programm kann ohne einen hohen Grad des Vertrauens unter den Teammitgliedern nicht funktionieren. Für die schwierigen Gespräche darüber, was richtig für das Programm und das Produkt ist, ist eine gewisse Offenheit erforderlich. Da Gespräche in regelmäßigen Intervallen stattfinden, werden auch regelmäßig Ideen und Bedenken vorgebracht. Teammitglieder müssen auch Vertrauen in die Fähigkeiten (und die Bereitschaft) der anderen haben, dass die während dieser Gespräche getroffenen Entscheidungen umgesetzt werden.

Unter dem Strich ist die Agile-Entwicklung ein strukturierter und iterativer Ansatz für die Softwareentwicklung. Du kannst damit auf Änderungen reagieren, ohne vom Weg abzukommen. Und das ist für jedes Programm von Vorteil

**Geschichte der agilen Softwareentwicklung**

Erste Ansätze zu agiler Softwareentwicklung sind Anfang der 1990er Jahre zu finden. Popularität erreichte die agile Softwareentwicklung erstmals 1999, als Kent Beck das erste Buch zu Extreme Programming veröffentlichte. Dies ebnete den Weg für andere agile Prozesse und Methoden. Zu Beginn war ***Extreme Programming*** die gängigste agile Methode,[[8]](https://de.wikipedia.org/wiki/Agile_Softwareentwicklung#cite_note-8) spätestens seit der ersten jährlichen Umfrage von VersionOne (2006) ist mit weitem Abstand Scrum die gängigste agile Methode.[[9]](https://de.wikipedia.org/wiki/Agile_Softwareentwicklung#cite_note-9)

Die Bezeichnung *agil* wurde im Februar 2001 bei einem Treffen in Utahauf Vorschlag von Mike Beedle ausgewählt, als Ersatz für das bis dahin gebräuchliche *leichtgewichtig* (engl. *lightweight*). Bei diesem Treffen wurde auch das *Agile Manifest* (siehe unten) formuliert.

2005 wurde von Forrester Research untersucht, dass 14 % der Unternehmungen in Nordamerika und Europa ihre Software mit agilen Prozessen entwickeln; weitere 19 % dachten über die Nutzung nach. VersionOne stellte 2013 fest, dass bereits 84 % aller Unternehmen agile Prozesse einsetzen,[[10]](https://de.wikipedia.org/wiki/Agile_Softwareentwicklung#cite_note-10) 2016 waren es 95 %.

### **Agile Leitsätze (Agiles Manifest)** Vier Leitsätze wurden im Februar 2001 als ***Agiles Manifest*** (englisch *Manifesto for Agile Software Development* oder kurz *Agile Manifesto*) formuliert:

„Wir erschließen bessere Wege, Software zu entwickeln, indem wir es selbst tun und anderen dabei helfen. Durch diese Tätigkeit haben wir diese Werte zu schätzen gelernt:

* **Individuen und Interaktionen** mehr als Prozesse und Werkzeuge
* **Funktionierende Software** mehr als umfassende Dokumentation
* **Zusammenarbeit mit dem Kunden** mehr als Vertragsverhandlung
* **Reagieren auf Veränderung** mehr als das Befolgen eines Plans

Das heißt, obwohl wir die Werte auf der rechten Seite wichtig finden, schätzen wir die Werte auf der linken Seite höher ein.“

– [Kent Beck](https://de.wikipedia.org/wiki/Kent_Beck), [Mike Beedle](https://de.wikipedia.org/wiki/Mike_Beedle), Arie van Bennekum, [Alistair Cockburn](https://de.wikipedia.org/wiki/Alistair_Cockburn), [Ward Cunningham](https://de.wikipedia.org/wiki/Ward_Cunningham), [Martin Fowler](https://de.wikipedia.org/wiki/Martin_Fowler), James Grenning, Jim Highsmith, [Andrew Hunt](https://de.wikipedia.org/wiki/Andy_Hunt_(Autor)), [Ron Jeffries](https://de.wikipedia.org/wiki/Ron_Jeffries), Jon Kern, Brian Marick, [Robert C. Martin](https://de.wikipedia.org/wiki/Robert_C._Martin), Steve Mellor, [Ken Schwaber](https://de.wikipedia.org/wiki/Ken_Schwaber), [Jeff Sutherland](https://de.wikipedia.org/wiki/Jeff_Sutherland) und [Dave Thomas](https://de.wikipedia.org/wiki/Dave_Thomas_(Programmierer))

### **Agile Prinzipien**

### *Agile Prinzipien* dienen als Leitsätze für agile Arbeit. Manchmal werden agile Prinzipien auch als Methode bezeichnet. Bei schwergewichtigen Prozessen werden Prinzipien von umfangreichen Methodenbeschreibungen überlagert und lassen Prinzipien in den Hintergrund treten; zudem wurden Prozesse früher hauptsächlich über Methoden, nicht über Prinzipien definiert. Die Benennung der Prinzipien soll ihnen gegenüber formalen Methoden wieder mehr Gewicht verleihen.

Im *Agilen Manifest* sind zwölf Prinzipien aufgelistet.[[13]](https://de.wikipedia.org/wiki/Agile_Softwareentwicklung#cite_note-13)

* Zufriedenstellung des Kunden durch frühe und kontinuierliche Auslieferung von wertvoller Software
* Agile Prozesse nutzen Veränderungen (selbst spät in der Entwicklung) zum Wettbewerbsvorteil des Kunden.
* Lieferung von funktionierender Software in regelmäßigen, bevorzugt kurzen Zeitspannen (wenige Wochen oder Monate)
* Nahezu tägliche Zusammenarbeit von Fachexperten und Entwicklern während des Projektes (Bsp.: Gemeinsamer Code-Besitz (Collective Code Ownership))
* Bereitstellung des Umfeldes und der Unterstützung, welche von motivierten Individuen für die Aufgabenerfüllung benötigt wird
* Informationsübertragung nach Möglichkeit im Gespräch von Angesicht zu Angesicht
* Als wichtigstes Fortschrittsmaß gilt die Funktionsfähigkeit der Software
* Einhalten eines gleichmäßigen Arbeitstempos von Auftraggebern, Entwicklern und Benutzern für eine nachhaltige Entwicklung
* Ständiges Augenmerk auf technische Exzellenz und gutes Design
* Einfachheit ist essenziell ([KISS-Prinzip](https://de.wikipedia.org/wiki/KISS-Prinzip))
* Die besten Architekturen, Anforderungen und Entwürfe entstehen in selbstorganisierten Teams
* Selbstreflexion der Teams über das eigene Verhalten zur Anpassung im Hinblick auf Steigerung der Effektivität

Der Übergang zwischen Prinzipien und Methoden ist fließend.

### **Agile Methoden** Agile Methoden sollen dazu dienen, dass die Aufwandskurve möglichst flach bleibt; d. h., Änderungen oder neue Anforderungen sollen mit wenig Aufwand berücksichtigt werden können.

Beispiele für agile Methoden sind:

* [Kontinuierliche Integration](https://de.wikipedia.org/wiki/Kontinuierliche_Integration)
* ständige [Refaktorierungen](https://de.wikipedia.org/wiki/Refaktorierung)
* [Continuous Delivery](https://de.wikipedia.org/wiki/Continuous_Delivery)
* [Continuous Deployment](https://de.wikipedia.org/wiki/Continuous_Delivery#Continuous_Deployment)
* [Paarprogrammierung](https://de.wikipedia.org/wiki/Paarprogrammierung)
* [Testgetriebene Entwicklung](https://de.wikipedia.org/wiki/Testgetriebene_Entwicklung)
* [Behavior Driven Development](https://de.wikipedia.org/wiki/Behavior_Driven_Development)
* [Story-Cards](https://de.wikipedia.org/wiki/Story-Card)
* schnelle [Codereviews](https://de.wikipedia.org/wiki/Codereview)
* [Agiles Testen](https://de.wikipedia.org/wiki/Agiles_Testen)
* [Rapid Application Development](https://de.wikipedia.org/wiki/Rapid_Application_Development)

Streng genommen bezeichnet *agile Methode* eine an Agilität ausgerichtete [Methode](https://de.wikipedia.org/wiki/Methode_(Softwaretechnik)) zur Softwareentwicklung.

### **Agile Prozesse**

### Zu den bekannten agilen Prozessen zählen:

* [Scrum](https://de.wikipedia.org/wiki/Scrum)
* [Extreme Programming](https://de.wikipedia.org/wiki/Extreme_Programming) (XP)
* [Kanban](https://de.wikipedia.org/wiki/Kanban_(Softwareentwicklung))
* Scrumban
* Dynamic Systems Development Method (DSDM)
* [Feature Driven Development](https://de.wikipedia.org/wiki/Feature_Driven_Development) (FDD)
* [Adaptive Software Development](https://de.wikipedia.org/wiki/Adaptive_Software_Development) (ASD)
* [Crystal](https://de.wikipedia.org/wiki/Crystal_Family)

Quelle:<https://de.wikipedia.org/wiki/Agile_Softwareentwicklung>,<https://www.atlassian.com/de/agile/project-management/program>