

programmierung und datenbanken

Joern Ploennigs

Softwareentwurf

MIDJOURNEY: WATERFALL IN CHINESE MOUNTAIN RANGE

WIEDERHOLUNG: HÖRSAALFRAGE

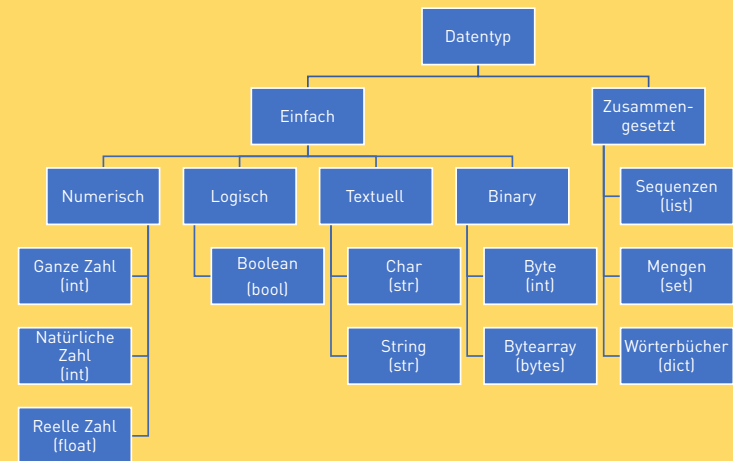
Welche Datentypen gibt es in Python?



Midjourney: A python programming a robot

WIEDERHOLUNG: DATENTYP - PYTHON

Python vereinfacht einige Datentypen.



WIEDERHOLUNG: HÖRSAALFRAGE

Was sind Objekte? ng, Generalisierung,
Kapselung, Polymorphismus?



Midjourney: Object oriented man

WIEDERHOLUNG: OBJEKTORIENTIERTE PROGRAMMIERUNG



Definition: Objektorientierte Programmierung

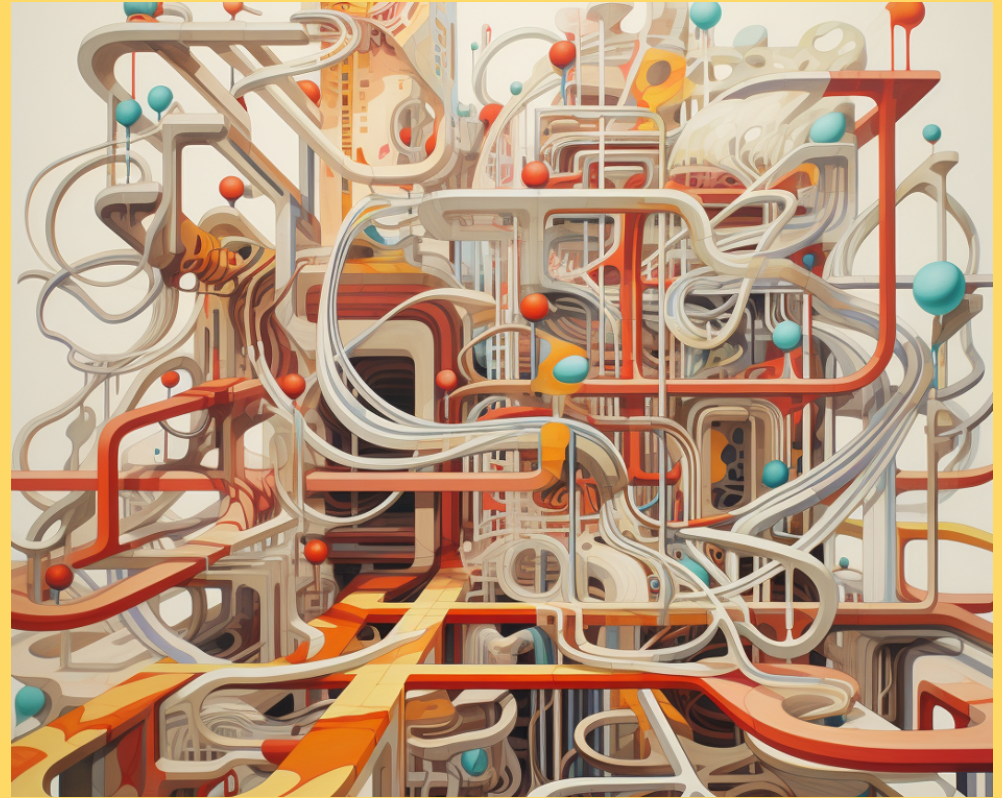
Objektorientierte Programmierung (OOP) ist ein Programmierparadigma das annimmt, dass ein Programm ausschließlich aus Objekten besteht, die miteinander kooperativ interagieren.

Jedes Objekt verfügt über:

- *Attribute* (Eigenschaften): Definiert den Wert über den Zustand eines Objektes.
- *Methoden* definieren die möglichen Zustandsänderungen (Handlungen) eines Objektes.

WIEDERHOLUNG: HÖRSAALFRAGE

Was sind die vier Grundelemente eines Programms?



Midjourney: Fundamentals of a Programming Language

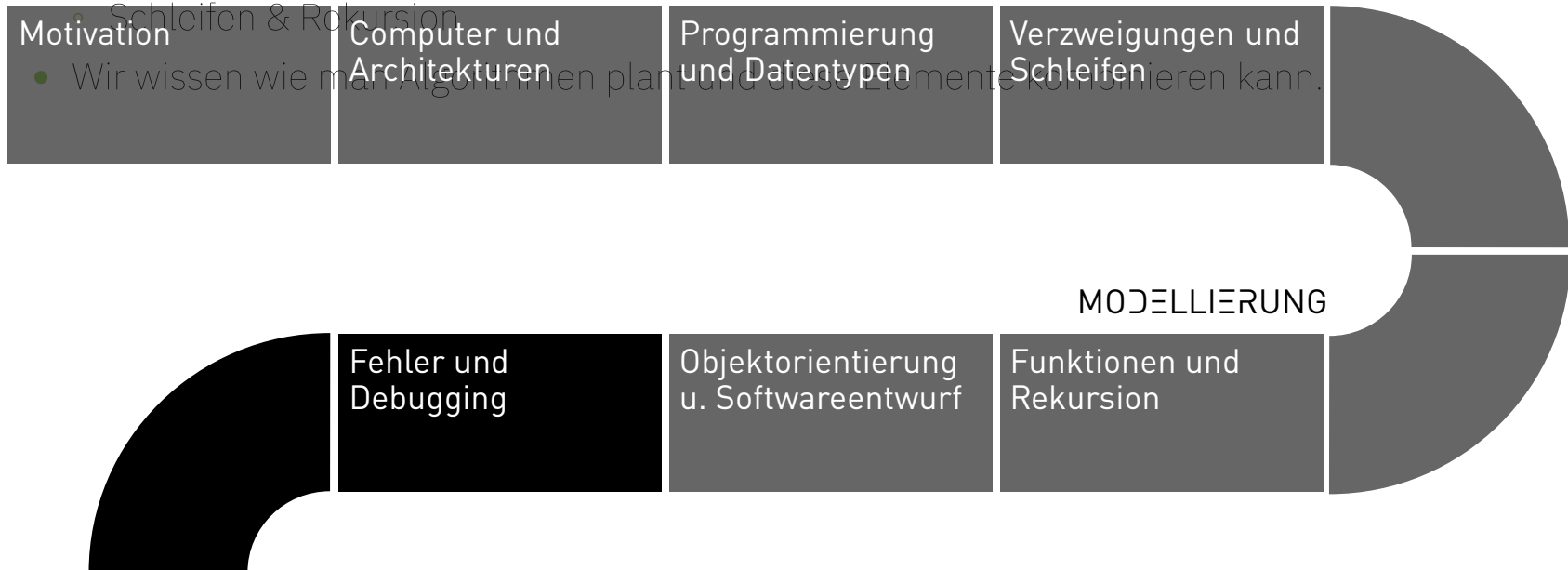
WIEDERHOLUNG: PROGRAMMELEMENTE

- Wir kennen alle grundlegenden Grundelemente eines Programms:

ABLAUF

- Funktionen
- Verzweigungen
- Schleifen & Rekursion

GRUNDLAGEN



MODELLIERUNG MIT OBJEKTEN

- Unsere Wahrnehmung und unser Denken besteht darauf Objekte zu erkennen. Das setzt voraus, dass wir das Objekt von anderen Objekten trennen können (z.B. Stuhl und Tisch)
- Das Nachbilden dieser Wahrnehmung und dieses Weltverständnisses im Computer, nennt man *Modellierung*.
- Da unsere Wahrnehmung auf Objekten basiert, ist es naheliegend für die Modellierung auch Objekte zu nutzen.
- Die Objekte in der Welt interagieren, dieses Verhalten müssen wir auch modellieren.



Definition: Modell

Ein Modell ist eine abstrakte, vereinfachte Abbildung eines Systems (meist aus der Realität).

HÖRSAALFRAGE

Was haben diese Elemente gemein?



HÖRSAALFRAGE

- Das sind alles *geometrische Objekte*
- Sie bestehen alle aus *Punkten*, die mit *Linien* verbunden sind
- Das können wir bei der Modellierung ausnutzen und Klassen so definieren, dass sie die *Gemeinsamkeit* ausnutzen



OBJEKTORIENTIERTER SOFTWAREENTWURF

Definition: Objektorientierten Softwareentwurf

Objektorientierten Softwareentwurf, wird ein Programm so entworfen, dass es nur aus *Objekten* besteht.

- Der Softwareentwurf gleicht dabei stark dem Vorgehen von Ingenieuren beim Entwurf von Plänen
- Die übliche Modellierungssprache sind *UML Diagramme*.
- Man modelliert im Entwurf wie:
 - wie Objekte in Form von *Klassen* definiert sind,
 - welche *Attribute und Methoden* sie besitzen,
 - wie diese Klassen aufeinander aufbauen (*Vererbung*),
 - als auch wie sie miteinander in statischer Beziehung stehen (*Referenzen*)
 - und wie sie dynamisch interagieren (*Verhalten*)

REFERENZEN

- In Programmen stehen Objekte meist im Zusammenhang. z.B. besteht eine Linie aus zwei Punkten.
- Diesen Zusammenhang zweier Objektklassen bezeichnet man als *Referenz*.
- Referenzen in Python erstellt man als Attribut vom Datentyp des anderen Objektes.

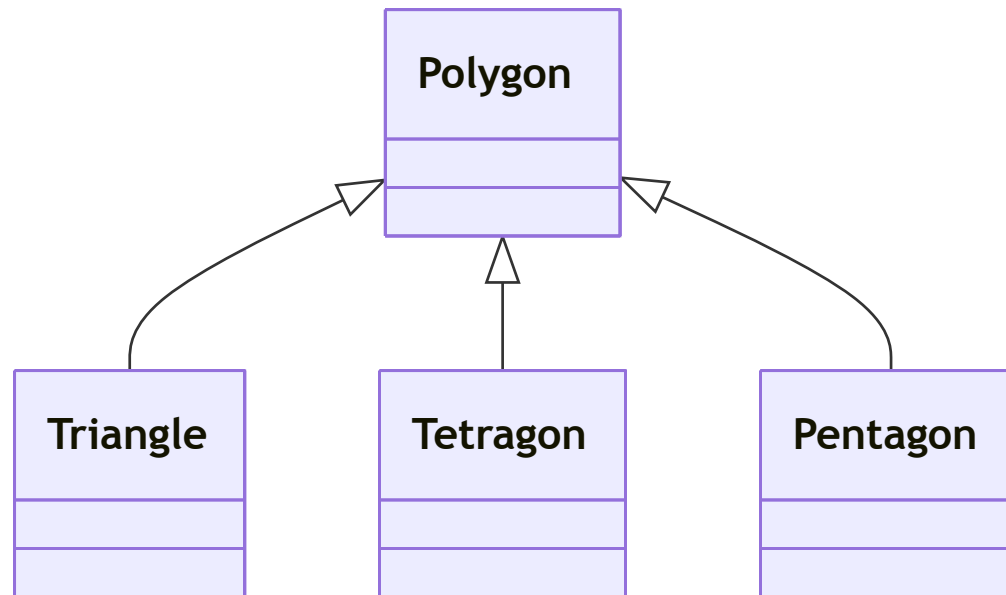
```
class Line:
    def __init__(self, start, end):
        self.start = start
        self.end = end

    def length(self):
        return
self.start.distance(self.end)
```

VERERBUNG

- Gleiche Objekte teilen oft auch ähnliche Eigenschaften und Methoden. z.B. bestehen alle geometrischen Elemente aus Punkten die mit Linien verbunden sind
- Um diese Attribute und Methoden nicht jedes mal neu definieren zu müssen dabei ggf. Fehler zu machen, gibt es die *Vererbung* in der OOP
- OOP erlaubt das Definieren von spezialisierten *Unterklassen*, bzw. von generalisierten *Oberklassen*.
- Die Unterklasse (Spezialklasse) besitzt die Methoden und Attribute der generalisierten Oberklasse (Basisklasse).
- Angelehnt an die Genetik auch „*Vererbung*“ genannt

BEISPIEL VERERBUNG



POLYMORPHIE

- Ein Objekt einer *Spezialklasse* kann stets auch als Mitglied der *Basisklasse* betrachtet werden.
- Die Spezialklasse kann die geerbten Methoden/Attribute der Basisklasse *überschreiben* und somit umdefinieren.
- In statisch typisierten Sprachen gilt außerdem: In einer Variablen die ein Objekt der Basisklasse aufnehmen kann, kann auch ein Objekt einer abgeleiteten Klasse gespeichert werden.

DATENKAPSELUNG

- Ziel der *Datenkapselung* ist es Kontrolle darüber zu haben, welche Attribute lesbar oder beschreibbar sind und wie dies geschehen kann
- Ferner kontrolliert man ob Attribute nur *lesbar* oder auch *schreibbar* sind. Dafür werden Attribute als *privat* deklariert und sind dann nur durch Get-Funktionen lesbar und durch Set-Funktionen veränderbar.
- Hierfür deklariert man Attribute oder Methoden als:
 - *privat* – Nur die Instanz der gleichen Klasse kann zugreifen
 - *protected* – Nur die Instanz der gleichen Klasse oder Unterklasse kann zugreifen
 - *public* – Alle können zugreifen

DYNAMIK IN KLASSEN

- In der OOP wird angenommen, dass Klassen *statisch* sind und es keine anderen als die in der Klasse definierten Attribute und Methoden gibt.
- Python bietet ein höheres Maß an Dynamik als andere Programmiersprachen:
 - Dynamische Typisierung von Variablen
 - Dynamisches Binden von Methoden und Attributen
 - Wir können sogar Attribute und Methoden selbst hinzufügen, wenn diese nicht in der Klasse definiert sind.
 - So können z.B. zur Laufzeit Referenzen zwischen Objekten schaffen die im vordefinierten Datenmodell keinen Bezug haben.

OBJEKTORIENTIERTE PROGRAMMIERPARADIGMEN

Vererbung	Generalisierung	Polymorphismus	Kapselung
Attribute und Methoden von Elternklassen werden an Kinder vererbt. Das hilft Redundanzen und Fehler zu vermeiden.	Gemeinsamkeiten werden in generalisierten Eltern-Klassen implementiert	Kinderklassen können Methoden überschreiben und somit umdefinieren.	Kapselung von Daten und Methoden in Objekten ist ein Schutzmechanismus, um schadhafte Änderungen einzuschränken.

UNIFIED MODELLING LANGUAGE - GRUNDLAGEN

- *UML (Unified Modelling Language)* ist ein ISO-Standard zur Beschreibung von Softwareentwürfen
- Er wird genutzt für den Entwurf, Ausschreibung, Implementation und Dokumentation von:
 - Funktionalitäten
 - Strukturen
 - Businessprozessen
 - Nutzerinteraktionen

Der Standard umfasst 13 Diagrammtypen, die typischsten:

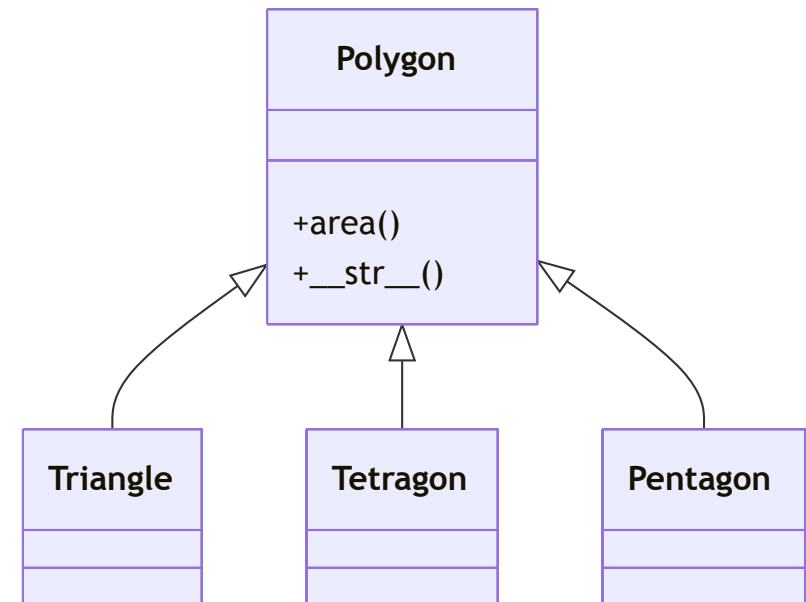
- *Klassendiagramme* (Für die Klassen- und Datenstruktur)
- *Programmablaufplan* (Für allgemeine Abläufe und Algorithmen)
- *Sequenzdiagramme* (Für Interaktionen)
- *Use-Case Diagramme* (Für Nutzer und Nutzfälle)

UML KLASSENDIAGRAMM

- Das am *häufigsten verwendete Diagramm* im objektorientierten Modellieren
- Im Klassendiagramm modelliert werden:
 - Klassen mit: Attributen, Methoden
 - Beziehungen
 - Hierarchien & Vererbungen

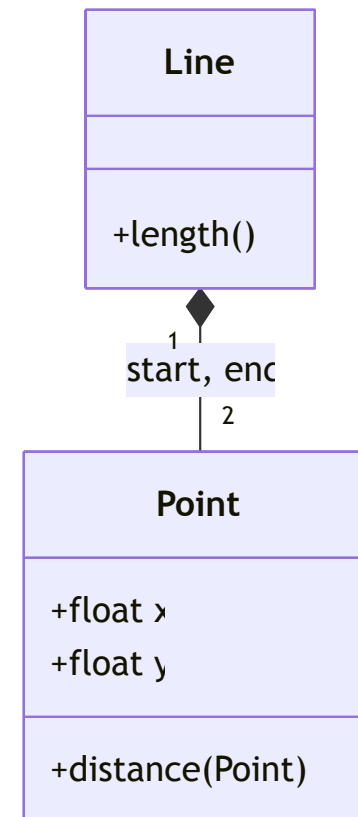
KLASSENDIAGRAMM – VERERBUNG

- *Vererbung* wird in UML durch ein Referenz mit einem *gefüllten Dreieck* "▲" bei der Oberklasse gezeichnet.
- Um zum Beispiel auszudrücken, dass **Triangle**, **Tetragon** und **Pentagon** Unterklassen des Polygons sind, können wir zeichnen.



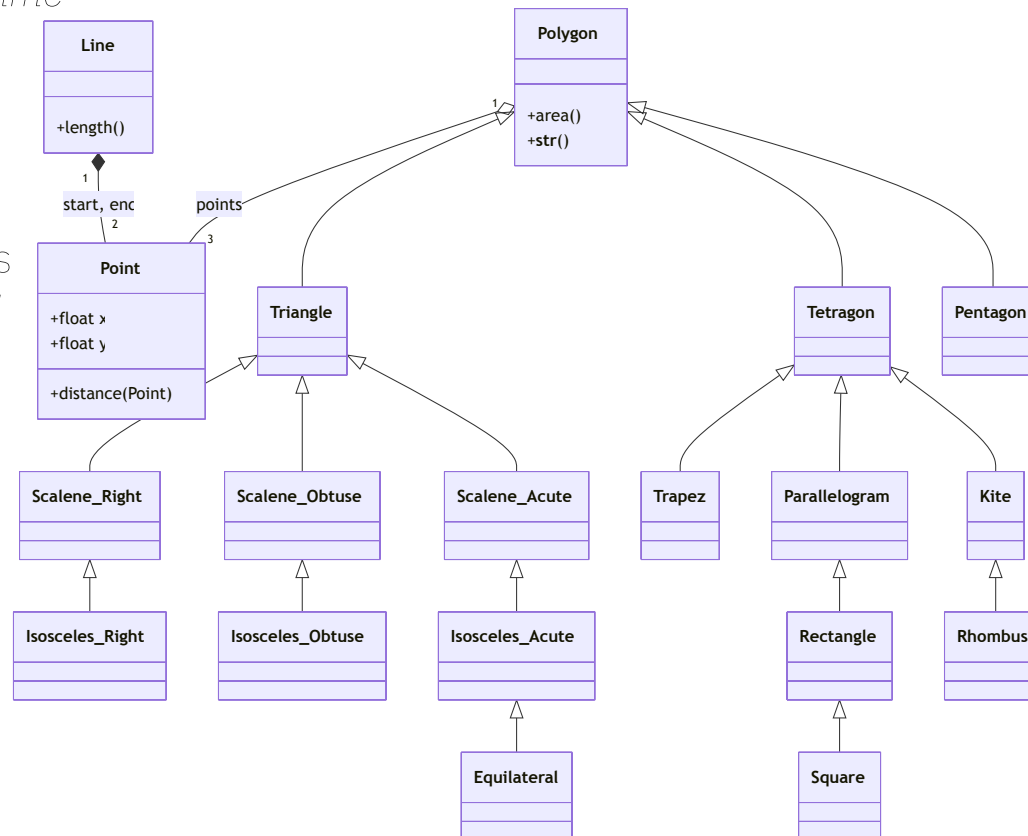
KLASSENDIAGRAMM - REFERENZEN

- *Referenzen* werden in UML mit Linien zwischen Objekten gekennzeichnet. Die Art der Linie und des Pfeils geben den Typ der Referenz an.
- Man unterscheidet in UML die Referenzen in der Art der *Besitzverhältnisse*:
 - *Aggregated* - ein Objekt ist Teil eines anderen und kann ohne ihn existieren
 - *Composition* - kann nicht ohne diesen existieren
 - *Assoziation* - komplett unabhängig
- *Zahlen* an den Referenzen geben die *Multiplizität* an, also wie viele Objekte in dieser Relation im Zusammenhang stehen.



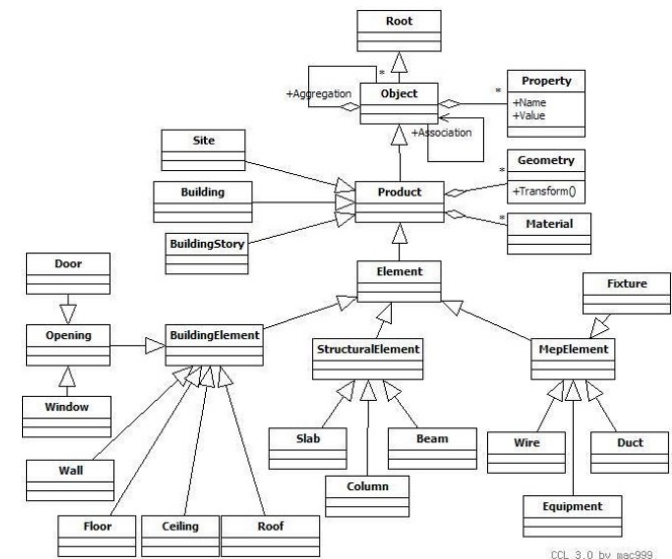
KLASSENDIAGRAMM - GESAMTBEISPIEL

- *Klassendiagramme* können recht schnell groß werden
- Sie eignen sich sehr gut um das (Daten-) Modell darzustellen



ANWENDUNGSBEISPIEL – BIM (BUILDING INFORMATION MODELS)

- Bauzeichnungen werden heutzutage als *IFC* (*Industry Foundation Class*) gespeichert
- *IFC* ist ein objektorientiertes Model, mit Vererbung, Spezialisierung, Generalisierung und Polymorphismus
- Es wird u.a. in *UML* dokumentiert



ENTWURFSVORGEHEN



Midjourney: Software Waterfall

SOFTWAREENTWURF

- Software wird selten allein entwickelt, sondern oft im Team über einen längeren Zeitraum.
- Dafür wird ein Projektplan benötigt, der beschreibt wie die Software arbeitsteilig entwickelt werden soll
- Das Erstellen dieses Projektplans nennt man *Softwareentwurf*
- Der Ablauf ähnelt stark dem *Entwurfsvorgehen im Umwelt- und Bauingenieurwesen*
- Der Projektplan umfasst:
 - den *Klassenentwurf* (Bauplan)
 - die *Programmiervorgehen* (Bauablaufplanung)

SOFTWAREENTWURF - PHASEN

- *Anforderungsdefinition* – Definition welche Funktionen und Randbedingungen realisiert werden
- *Entwurf*
 - Klassenentwurf der Software
 - Ausführungsplanung – Was wird wann, durch wen, und wie implementiert
- *Ausführung*
 - Implementation – Programmieren der einzelnen Klassen & Module
 - Integration – Zusammenführen der Module zur fertigen Lösung
- *Abnahme* – Test des fertigen Softwareprogramms
 - Modultest – Unit-Test von Modulen
 - Integrationstest – Kombination von Module testen
 - Systemtest – Gesamtheit der Module testen

VERGLEICH: SOFTWARE- VS. BAUINGENIEURWESEN

Softwareentwurf

- *Anforderungsdefinition*: Definition welche Funktionen und Randbedingungen realisiert werden
- *Entwurf*
 - Modulentwurf – Definition welche Teile die Software hat
 - Klassenentwurf – Definition in welche Klassen die Software strukturiert ist
 - Ausführungsplanung – Was wird wann, durch wen, und wie implementiert

Bauingenieurwesen

- *Ausschreibung*: Dokumentieren welche Funktionen und Randbedingungen das Bauwerk erfüllen
- *Entwurf*
 - Architektur und Grobentwurf – Definition wie das Bauwerk grob strukturiert ist
 - Fein- und Gewerkplanung - Definition wie Teile und die Gewerke umgesetzt werden
 - Ausführungsplanung – Was wird wann, durch wen, und wie gebaut

VERGLEICH: SOFTWARE- VS. BAUINGENIEURWESEN (FORTSETZUNG)

Softwareentwurf (Fortsetzung)

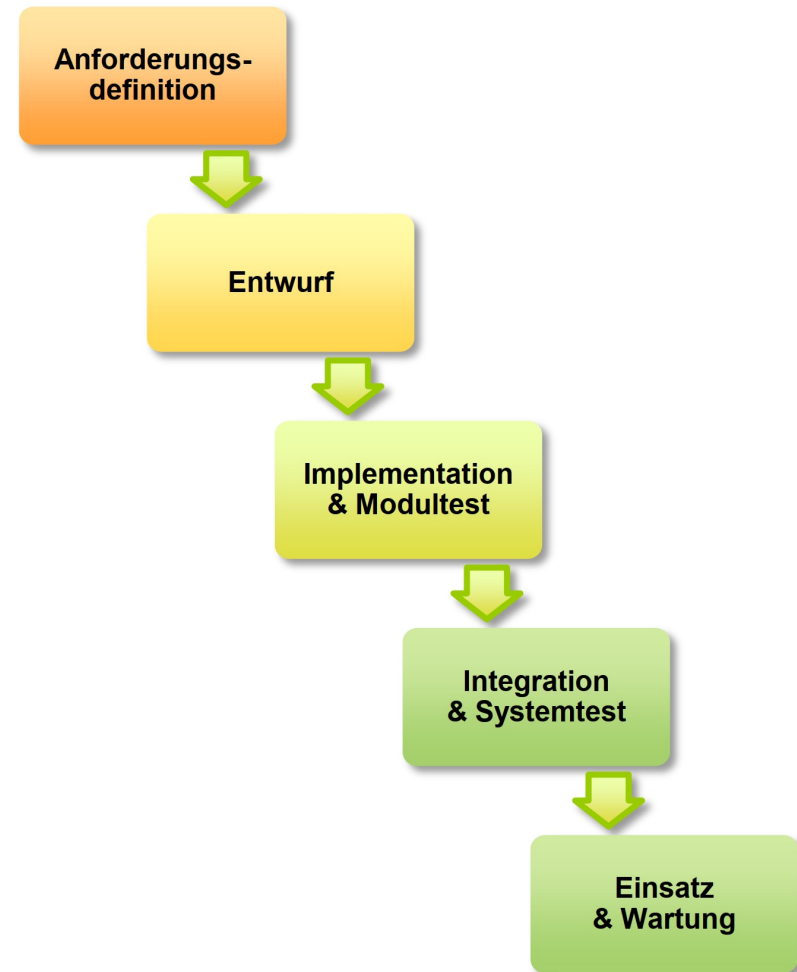
- *Ausführung*
 - Implementation – Programmieren der einzelnen Klassen & Module
 - Integration – Zusammenführen der Module zur fertigen Lösung
- *Abnahme*
 - Modultest – Unit-Test von Modulen
 - Integrationstest – Kombination von Module testen
 - Systemtest – Gesamtheit der Module testen

Bauingenieurwesen (Fortsetzung)

- *Ausführung*
 - Einzelne Gewerke werden umgesetzt
 - Gewerke im Bau integrieren, wie Energie und Wasser mit Klima integrieren
- *Abnahme*
 - Abnahme einzelner Gewerke
 - Test mehrerer Gewerke gemeinsam
 - Gesamtbauwerk abnehmen

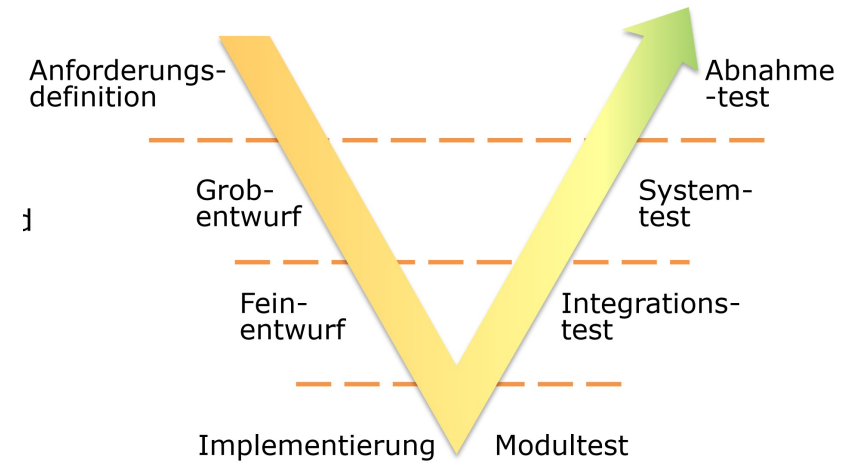
LINEARE METHODE - WASSERFALLMETHODE

- Traditionelles Modell das häufig noch in der Ausschreibung großer Systeme gefordert wird
- Entwicklung ist in mehrere sequenzielle Schritte aufgeteilt und jeder Schritt muss vor dem nächsten beendet werden
- Benutzerbeteiligung nur in der Anforderungsdefinition
- Jeder Aktivität wird dokumentiert → Gut geeignet für Ausschreibungen (nach der Anforderungsdefinition oder dem Entwurf, ISO 9000)



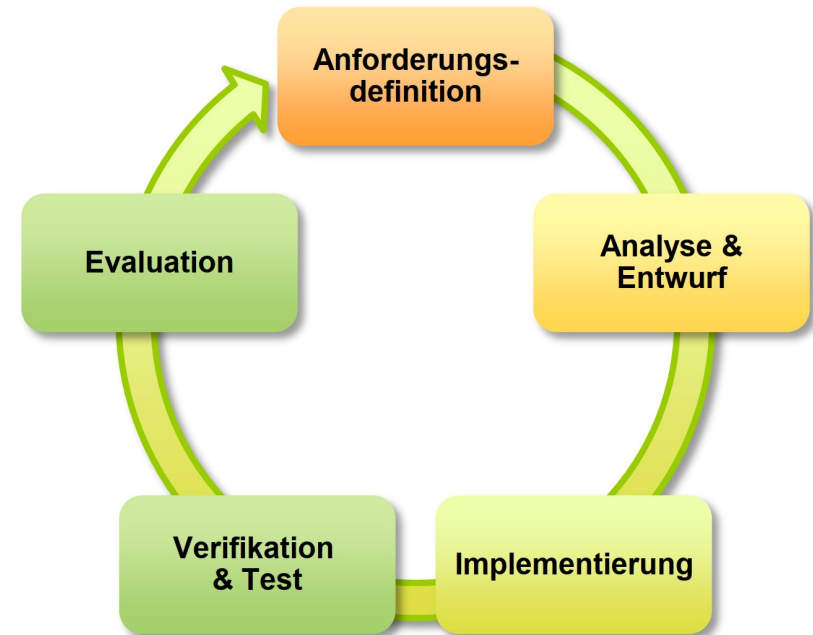
V-METHODE

- Primär verwendet bei der Entwicklung sicherheitskritischer Software (Auto, Flugzeug, etc.)
- Separiert Entwicklungs- (linke Seite) und Testaktivitäten (rechte Seite)
- Tests werden schon im Entwicklungsarm definiert
- Ziel einer hohen Testabdeckung
- Benutzerbeteiligung in der Anforderungsdefinition und im Abnahmetest



AGILE METHODE

- Moderne Methode um mit sich an ständig ändernden Anforderungen anpassen
- Ziel der schrittweisen Entwicklung der Lösung, um Aufwand und Komplexität einzelner Schritte in Grenzen zu halten
- Startet mit einer einfachen und ausbaufähigen Implementierung (MVP – Minimal Viable Produkt)
- Schrittweise Erweiterung und Verbesserung des Produktes mit regelmäßigen Releases (meist alle 3 Monate)
- Entwurfsfehler der ersten Iterationen können zu kompletten Neuentwurf führen



fragen?

