PROGRAMMIERPROJEKT 2018

EINFÜHRUNG

Annette Bieniusa

ZIELE DES PROJEKTS

- Planung und Durchführung eines Projekts zur Software-Entwicklung

PROGRAMMIERUNG!

INHALTE

- **Softwareentwicklungsprozesse**
  - Phasen und ihre Interaktion
- **Softwarearchitekturen**
  - Client-Server
  - Model-View-Controller
- **Modellierung** von OOP
  - UML (Klassendiagramme und Use Cases)
- **Bibliotheken** und **Frameworks**
  - Web-Frameworks
  - Datenbanken (SQL)
- **Werkzeuge** und Methoden
  - Entwicklungsumgebungen
  - Testing
  - Versionsverwaltung (GitLab)

ORGANISATORISCHES

- Regelmäßige Treffen (ca. wöchentlich)
  - immer Dienstags 17:15
- Programmieren in Teams à 3 Personen
- Benotung auf Grund von Einzelleistungen
  - Software-Komponenten
  - Dokumentation / Berichte
- Einzureichende Artefakte
  - Lastenheft => Pflichtenheft
  - Erster Prototyp
  - Grundversion (Implementierung der Kernanforderungen)
  - Finale Version (Ergänzung um optionale Anforderungen)
  - Präsentationsfolien

ZEITERFASSUNG

- Legen Sie eine Liste mit Ihren Stunden an!
- Teil der jeweiligen Abgabe(n)
- Was wird erfasst?
  - Teilnahme an Treffen
  - Schreiben von Dokumentation
  - Coden
  - Testen
  - Debuggen
  - Lektüre von Fremdquellen
- **Mindestens 120 Stunden gesamt!**
- **Maximal 20 Studen reserviert für Abschlusspräsentation**

ABSCHLUSSPRÄSENTATION

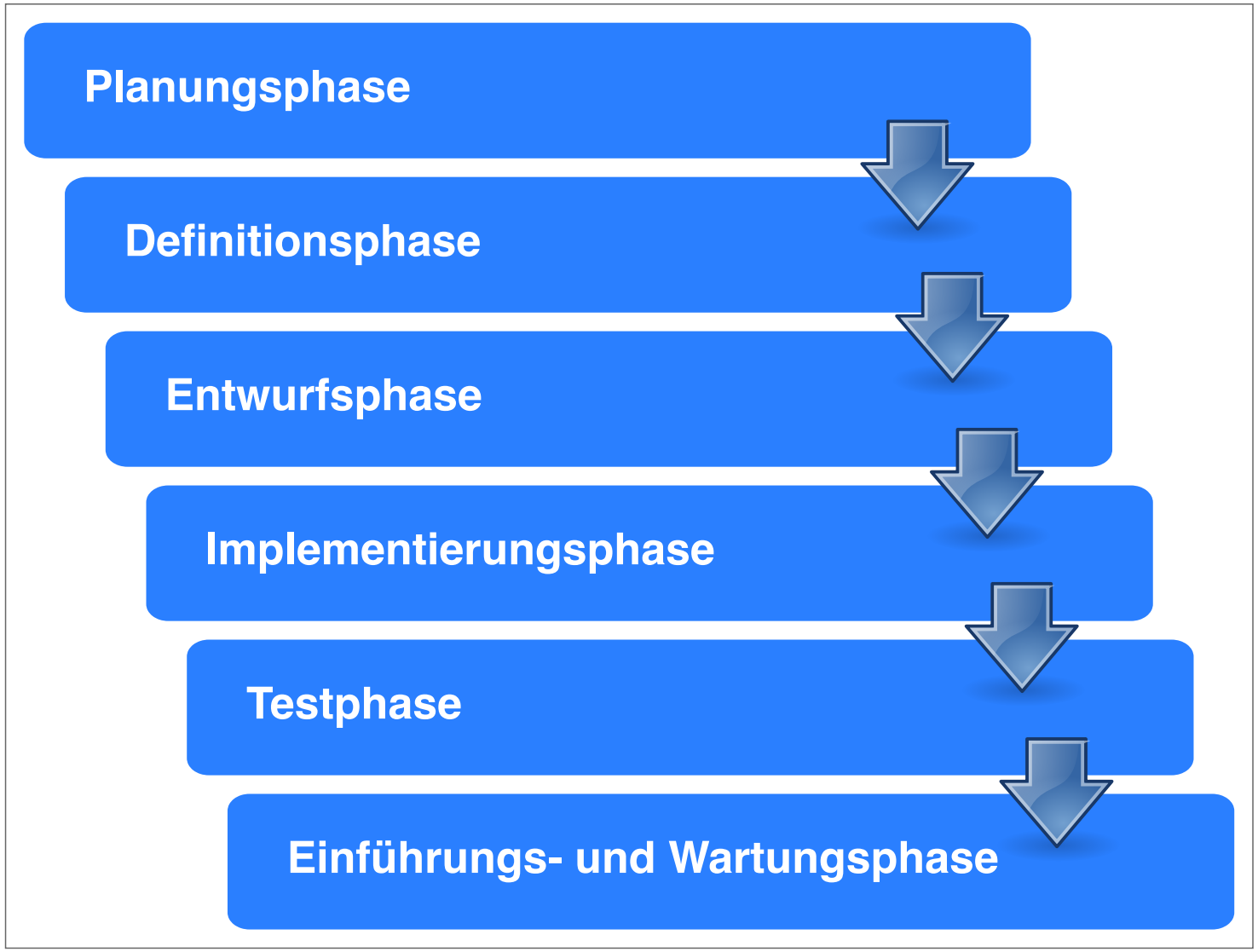
- In der Woche vom 06. - 10.08.

Bitte Notebooks zu den Treffen mitbringen, falls vorhanden.

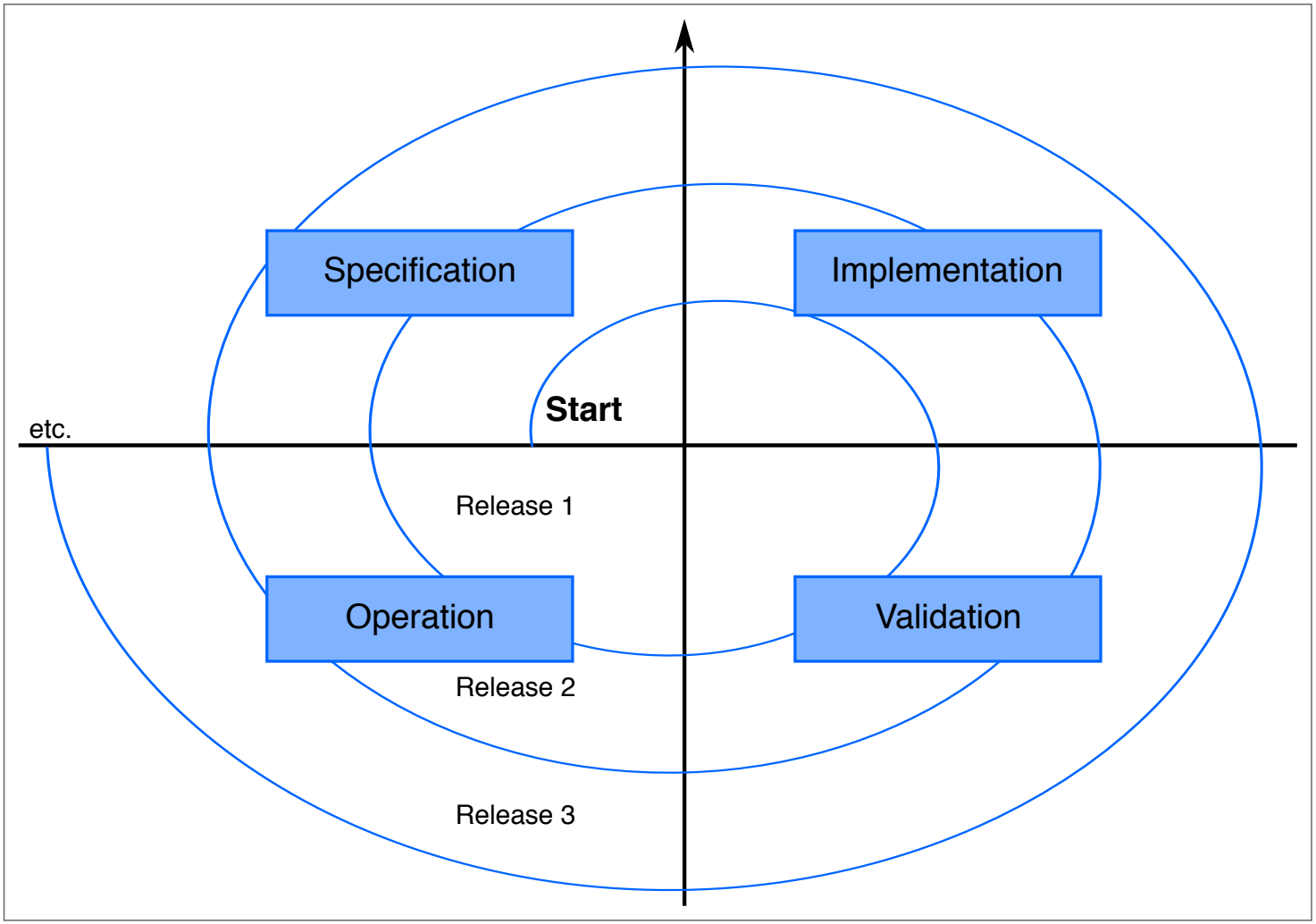
- Betriebssystem: Windows, Linux, Mac
- Richtige Tastatur
- **Kein Android Tablet oder iPad**

PHASEN DER SOFTWARE-ENTWICKLUNG

WASSERFALL-MODELL



EVOLUTIONÄRE MODELLE



Sommerville, Software Engineering, 9 ed., Chapter 9

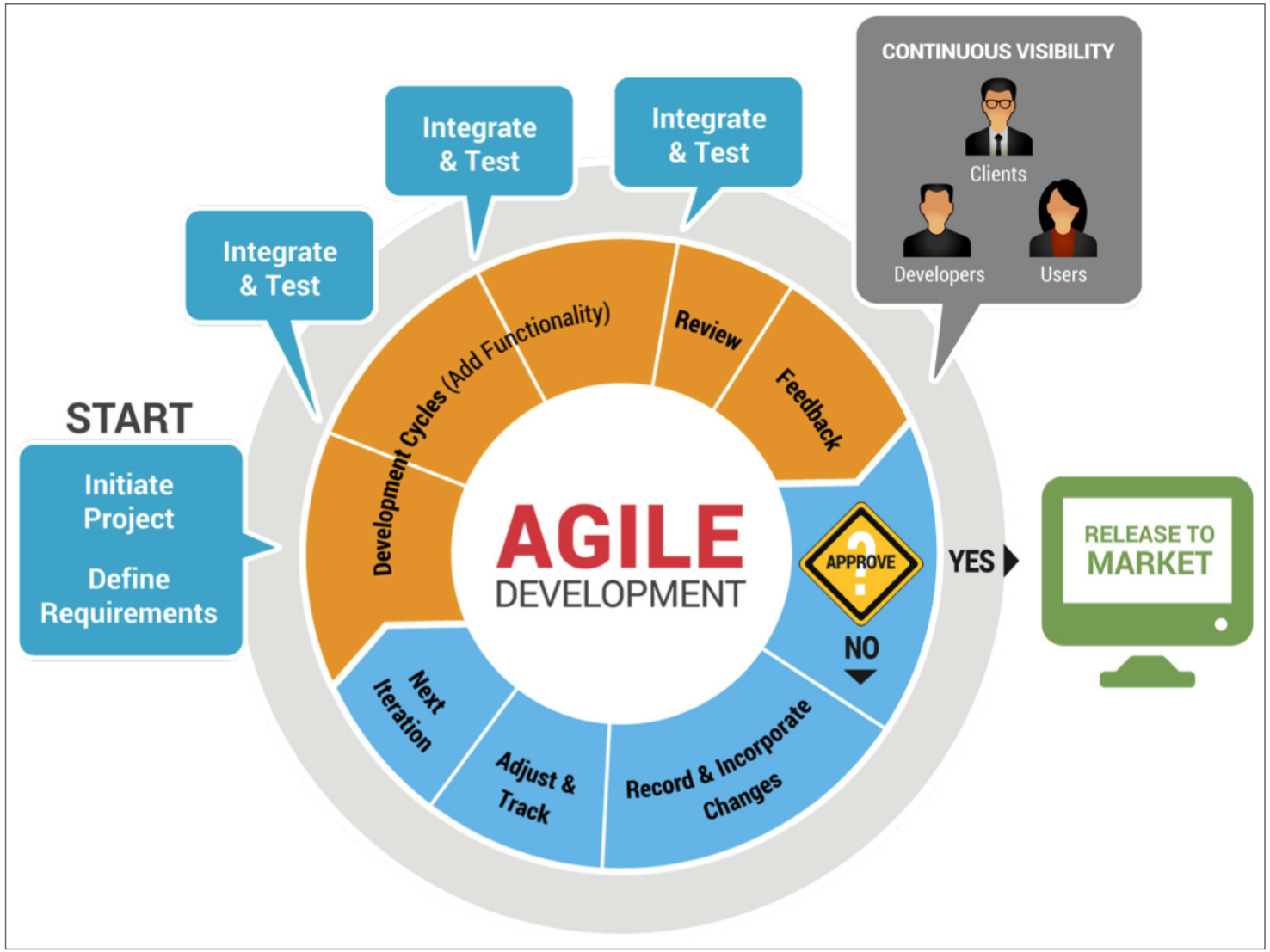
EVOLUTIONÄRE MODELLE

- Stufenweise, allmähliche Entwicklung
- Produktkern basiert auf den Muss-Anforderungen
- Weitere Entwicklung/Anforderungen sind gesteuert durch Erfahrungen mit dem Produkt
- Gefahr: Systemarchitektur muss für spätere Anpassungen vollständig überarbeitet werden

INKREMENTELLE MODELLE

- Zu Beginn werden alle Anforderungen vollständig erhoben
- Anforderungen werden dann schrittweise implementiert
- Mehr Zeitaufwand am Anfang
- Gefahr: Anforderungen werden übersehen

AGILE SOFTWARE-ENTWICKLUNG



AGILE SOFTWARE-ENTWICKLUNG

- Flexibler und schlanker Prozess
- Agiles Manifest:
  1. **Menschen und Interaktionen** sind wichtiger als Prozesse und Werkzeuge.
  2. **Funktionierende Software** ist wichtiger als umfassende Dokumentation.
  3. **Zusammenarbeit mit dem Kunden** ist wichtiger als die ursprünglich formulierten Leistungsanforderungen.
  4. **Eingehen auf Veränderungen** ist wichtiger als Festhalten an einem Plan.

AGILE METHODIK UND PRINZIPIEN

- Story cards / User stories
- Pair programming
- Testgetriebene Entwicklung

- Selbstorganisation und -reflexion des Teams
- Fokus auf funktionierenden Code
- Regelmäßige und häufige Releases

BEISPIEL: ANFORDERUNGSANALYSE FÜR EINE TODO-APP

INTERAKTIONEN MIT BENUTZER (USER STORIES)

ÜBERSICHT

LISTE ANLEGEN

- Voraussetzungen: Keine
- Namen wird eingegeben und Listen im System hinterlegt
- Namen muss eindeutig sein
- Liste ist im System zugreifbar

TODO-ITEM ANLEGEN

- Voraussetzung: Liste muss im System vorhanden sein
- Benutzer gibt Titel des Items an
- TODO-Item wird auf Liste aufgenommen
- Item ist nicht erledigt

ITEM ALS ERLEDIGT MARKIEREN

- Voraussetzung: Item auf existierender Liste und Item nicht erledigt
- Benutzer markiert Item als erledigt
- Item taucht nicht mehr in der Liste der zu erledigenden Items auf

ERLEDIGTE ITEMS ANZEIGEN

- Voraussetzung: Liste muss im System vorhanden sein
- Benutzer zeigt Liste an
- Benutzer aktiviert die Anzeige erledigter Items
- Alle Items, auch die erledigten, werden angezeigt
- Erledigte Items sind farblich markiert

LISTE EXPORTIEREN

- Voraussetzung: Liste muss im System vorhanden sein
- Inhalt der Liste wird in geeignetem Format in eine Datei geschrieben
- Benutzer kann frei wählen, wo die Datei auf dem Rechner abgelegt werden soll.

LISTE IMPORTIEREN

- Voraussetzung: Datei mit Inhalt der Liste durch *Liste exportieren* liegt vor
- Benutzer wählt Datei auf seinem Rechner aus
- Benutzer wählt Namen für die Liste
- Neuer Name der Liste darf nicht schon vorhanden sein
- Importierte Liste ist danach unter dem gewählten Namen im System vorhanden

DATENMODELL

TODO-LISTEN

- Name der Liste
- Einträge/Items der Liste

ITEMS/EINTRÄGE

- Titel
- Erledigt-Flag

SICHTEN AUF DIE DATEN

LISTENÜBERSICHT

- Auflistung aller Listen im System
- Nur Anzeige der Namen

ANZEIGE EINER LISTE

- Name der Liste
- Anzeigen nicht erledigter Items mit Titel

ERWEITERTE LISTENANSICHT

- Name der Liste
- Anzeigen aller Items, auch der erledigten
- Erledigte Items sind als solche markiert

EINTEILUNG DER ANFORDERUNGEN

KERNANFORDERUNGEN

- Liste anlegen
- TODO-Item anlegen
- Item als erledigt markieren
- Erledigte Items anzeigen

ZUSÄTZLICHE FEATURES

- Liste exportieren
- Liste importieren

UNSER PROJEKT

- Inkrementelles Vorgehen
  - Zunächst: Produktspezifikation
  - Prototyp der Kernanforderungen
  - Dann iteratives Erweitern der Software
- Elemente aus der Agilen Software-Entwicklung
  - Test-driven development
  - Iteratives Vorgehen

PROJEKT 1: VERWALTUNG VON ÜBUNGSGRUPPEN

Assistentin Lisa Lustig verliert allmählich den Überblick bei der Punkteverwaltung der Übungsaufgaben.

- Für jeden Studierenden der Veranstaltung müssen die Punkte pro Übungsblatt von den jeweiligen Tutoren eingetragen werden.
- Tutoren sollen die Punkte ihrer eigenen Studierendengruppe sehen können.
- Studierende sollen ihren eigenen akutellen Punktestand einsehen können.
- Für jedes Blatt soll außerdem die Punkteverteilung aller Abgaben angezeigt werden, damit die Studierenden ihre Leistung einschätzen können.
- Am Semesterende soll berechnet werden, welche Studierenden die Zulassungsvoraussetzung erfüllen (50% der erreichbaren Punkte).
- etc.

PROJEKT 2: UNSER MODULHANDBUCH SOLL SCHÖNER WERDEN!

Der Vizepräsident für Studium und Lehre der TU Kunterbunt will Ordnung in das Chaos der Modulhandbücher bringen.

- Die Daten für das neue universitätsweite Modulhandbuch sollen von den Studienmanagern für ihren Fachbereich eingegeben und bei Bedarf anpasst werden.
- Interessenten können sich die Informationen online in einer Übersicht ansehen und durch Klick auf eine Veranstaltung die Details erfahren.
- Manche Module werden von anderen Fachbereichen importiert (z.B. wird das "Programmierprojekt" des Fachbereichs Informatik vom Fachbereich WiWi). Diese Module dürfen nur dann gelöscht bzw. geändert werden, wenn alle betroffenen Fachbereiche zustimmen.
- Änderungen sollen erst dann sichtbar werden, wenn der jeweilige Fachbereichsrat (und evtl. abhängige Fachbereiche) zugestimmt haben.
- Man muss neben dem aktuellen Stand auch die früheren Einträge einsehen können. Etc.

IHRE AUFGABE HEUTE

- Bilden Sie Gruppen von 3-4 Studierenden!
- Wählen Sie eine Projektidee (Übungspunkte vs. Modulhandbuch)!
- **Bis 18.04.18**: Ausarbeitung der Projektidee (per Email abgeben)
  - User stories: Welche Benutzergruppen gibt es? Wie interagieren diese mit dem System? (Voraussetzungen für die jeweilige Story analysieren!)
  - Datenmodell: Welche Daten müssen gespeichert werden? Welche Zusammenhänge gibt es zwischen den Daten?
  - Sichten auf die Daten: Welche Ansichten muss die Anwendung haben? (Anmelde/Abmeldefenster, Eingabemasken, Übersichten, ...)
  - Unterscheidung: Kernanforderungen vs. optionale Features