Übungsblatt 4 – Modelle und agile Software Entwicklung

Luca M. Schmidt

1. Spiralmodell nach Böhm

a. Umgang mit Änderungen und Ansatz

Das Spiralmodell geht **iterativ und risikogesteuert** mit Änderungen um.

Begründung:

- **Iterativ:** Das Modell durchläuft Zyklen (Spiralen), wobei jeder Zyklus die Phasen Zieldefinition, Risikoanalyse, Entwicklung/Validierung und Planung des nächsten Zyklus beinhaltet. Änderungen können in der Planungsphase einer neuen Iteration berücksichtigt werden.
- **Risikogesteuert:** Ein Kernaspekt jeder Iteration ist die Risikoanalyse. Änderungen im Projektumfeld oder neue Erkenntnisse (die oft zu Änderungsbedarf führen) werden als Risiken bewertet und fließen in die Planung der nächsten Schritte ein.
- **Prototyping:** Frühe Iterationen fokussieren oft auf Prototypen, um Anforderungen zu klären und Risiken zu minimieren. Das Feedback aus dem Prototyping führt naturgemäß zu Anpassungen und Änderungen.
- **Flexibilität:** Im Gegensatz zu strikt sequentiellen Modellen ist das Spiralmodell darauf ausgelegt, auf Änderungen reagieren zu können, indem diese in den nächsten Zyklus integriert werden.

b. Ausprägungen der Kriterien: V-Modell vs. Spiralmodell

Unterscheidungskriterien	V-Modell	Spiralmodell
Ziele und Pläne	Detailliert, frühzeitig und umfassend für das Gesamtprojekt festgelegt.	Pro Iteration definiert/verfeinert; anfangs grob, werden detaillierter.
Risikoanalyse	Nicht expliziter Kernbestandteil des Grundmodells, oft als begleitender Prozess.	Zentraler, expliziter Bestandteil jeder einzelnen Iteration.
Prototypen	Nicht zwingend, aber für Anforderungsanalyse/UI-Design möglich.	Typischerweise in frühen Iterationen zur Risikominimierung & Anforderungsvalidierung.
Simulationen	Möglich, z.B. für Leistungsanalyse, aber nicht Kernbestandteil.	Können Teil der Risikoanalyse oder des Prototypings sein.

Unterscheidungskriterien	V-Modell	Spiralmodell
Tests	Definiert Teststufen (Komponente, Integration, System, Abnahme) parallel zu den Entwicklungsphasen.	Kontinuierlich in jeder Iteration, oft auf Prototypen oder Inkremente bezogen.
Dokumente	Umfassend, formal, meilensteinbasiert, oft hoher initialer Aufwand.	Iterativ erstellt, anfangs weniger detailliert, wächst mit dem Projekt und den Risiken.
Auslieferung	Typischerweise eine Gesamtauslieferung am Projektende.	Inkrementelle Auslieferungen funktionsfähiger Teile sind möglich und oft Ziel.
Inkrementelle Entwicklung	Grundmodell ist sequentiell, nicht inhärent inkrementell.	Von Natur aus inkrementell, da das System in Zyklen erweitert wird.
Umgang mit Änderungen	Schwieriger und teurer, da Pläne früh fixiert; formale Change-Requests.	Flexibler, Änderungen können in nachfolgenden Iterationen eingeplant werden.
Aufwand/Kosten für die Durchführung aller Schritte	Hoher initialer Planungs- und Spezifikationsaufwand.	Anfangs potenziell geringer, kann aber durch viele Iterationen und detaillierte Risikoanalysen auch hoch werden.

---

2. Softwareprozess-Modelle

a. Armbänder zum Zählen der Schritte (Smartphone-Sync)

- **Modellvorschlag: Agiles Vorgehen (z.B. Scrum oder Kanban)** oder ein **inkrementelles Modell**.
- **Begründung:**
  - **Schnelles Feedback:** Die Synchronisation mit einer Smartphone-App und die User Experience (UX) der App profitieren stark von schnellem Nutzerfeedback.
  - **Iterative Entwicklung:** Funktionen können schrittweise entwickelt und ausgeliefert werden (z.B. erst Schrittzählung, dann Schlaftracking, dann Benachrichtigungen).
  - **Anpassungsfähigkeit:** Markt für Wearables ist dynamisch; neue Anforderungen (z.B. Integration mit anderen Fitness-Apps, neue Smartphone-Betriebssysteme) können schnell aufkommen.
  - **Technologie-Unsicherheit:** Evtl. müssen verschiedene Bluetooth-Protokolle oder Energiesparmechanismen erprobt werden.

b. Blutdruckmessgerät (Speicher für 100 Messungen, ohne Export)

- **Modellvorschlag: V-Modell** oder ggf. **Wasserfallmodell**.
- **Begründung:**
  - **Klare, stabile Anforderungen:** Die Kernfunktionalität (Blutdruck messen, Wert speichern) ist gut definierbar und unterliegt voraussichtlich keinen häufigen Änderungen. Keine externen Schnittstellen

(Export) vereinfachen die Anforderungen.
- **Sicherheitskritikalität (implizit):** Medizinische Messgeräte erfordern hohe Zuverlässigkeit und Genauigkeit. Das V-Modell mit seiner starken Betonung von Verifikation und Validierung auf jeder Stufe unterstützt dies.
- **Nachweisbarkeit:** Die klare Struktur des V-Modells erleichtert die Dokumentation und den Nachweis, dass alle Anforderungen korrekt umgesetzt und getestet wurden (wichtig für etwaige Zertifizierungen, auch wenn hier nicht explizit gefordert).
- **Vorhersagbarkeit:** Bei klaren Anforderungen und geringer erwarteter Änderungshäufigkeit ermöglicht ein plangesteuertes Modell eine bessere Vorhersage von Aufwand und Zeit.

---

3. Plangesteuert oder agil? (nach Sommerville S. 93/94)

Die Entscheidung zwischen einem plangesteuerten und einem agilen Ansatz hängt von technischen, personellen und organisatorischen Faktoren ab. Hier eine Zusammenfassung der Kriterien:

1. **Detaillierungsgrad von Spezifikation und Entwurf vor Implementierung:**
    - **Hoch:** Eher plangesteuert (wenn Spezifikation und Entwurf sehr detailliert vorab nötig sind).
2. **Realistische inkrementelle Auslieferungsstrategie mit schnellem Kundenfeedback:**
    - **Ja:** Eher agil (wenn Software an Kunden geliefert und schnelles Feedback erhalten werden kann).
3. **Systemgröße und Teamstruktur:**
    - **Klein, ko-lokalisiertes Team, informeller Austausch:** Eher agil.
    - **Groß, große Entwicklerteams:** Eher plangesteuert.
4. **Art des Systems (Analysebedarf):**
    - **Hoher Analysebedarf vor Implementierung** (z.B. Echtzeitsysteme mit komplexen Synchronisationsanforderungen): Eher plangesteuert (benötigt detaillierten Entwurf für Analysen).
5. **Erwartete Lebenszeit des Systems und Wartung:**
    - **Langlebig:** Möglicherweise mehr Entwurfsdokumentation für Support (Argument für plangesteuert).
    - **Agile Sicht:** Dokumentation ist oft nicht aktuell und nützt wenig für langdauernde Wartung.
6. **Verfügbare Technologien und Werkzeuge zur Systementwicklung:**
    - **Gute Werkzeuge zur Beobachtung des sich entwickelnden Entwurfs:** Unterstützt agile Methoden.
    - **IDE ohne gute Werkzeuge für Programmvisualisierung/-analyse:** Vermutlich mehr Entwurfsdokumentation nötig (eher plangesteuert).
7. **Organisation des Entwicklerteams:**
    - **Dezentral oder ausgelagert:** Eventuell Entwurfsdokumente zur Kommunikation nötig (eher plangesteuert, mit Planung der Dokumente).
8. **Kulturelle Aspekte im Unternehmen:**
    - **Traditionell planungsbasierte Kultur (Standard im Ingenieurwesen):** Erfordert oft ausführliche Entwurfsdokumentation (eher plangesteuert).
9. **Qualifikation der Entwickler und Programmierer:**
    - **Agile Methoden erfordern oft höheres fachliches Können.**

- **Geringer qualifiziertes Team:** Detaillierter Entwurf durch "beste Leute", Rest setzt um (eher plangesteuert).

10. **Externe Vorschriften und Regulierung:**
    - **System muss von externer Behörde genehmigt werden (z.B. FAA für Flugzeugsoftware):** Wahrscheinlich ausführliche Dokumentation als Teil der Systemsicherheit nötig (eher plangesteuert).

Fazit (nicht explizit gefragt, aber impliziert): Die meisten Projekte beinhalten in der Praxis Elemente beider Ansätze. Die Frage ist oft nicht "entweder/oder", sondern die richtige Balance und Integration agiler Praktiken in einen übergeordneten Rahmen, oder die Ergänzung agiler Prozesse durch notwendige Planungs- und Dokumentationsartefakte.