

Vorgehensmodelle

Tag 2 – Spiralmodell, weitere Modelle, Modellauswahl

Schriftliche Ausarbeitung – Modul E (FIAE / FISl)

1 Einleitung

Tag 2 schließt direkt an die klassischen, sequenziellen Vorgehensmodelle des ersten Tages – Wasserfall und V-Modell XT – an. Im Mittelpunkt steht jetzt der Übergang vom plangetriebenen zum iterativen und risikoorientierten Denken. Den Auftakt bildet das Spiralmodell nach Barry Boehm aus dem Jahr 1988, das als historisches Bindeglied zwischen klassischen und agilen Ansätzen gilt. Anschließend werden mit dem Rational Unified Process, Extreme Programming und dem Prototypenmodell drei weitere wichtige Modelle eingeführt, bevor die Frage nach der richtigen Modellauswahl im Vordergrund steht.

Diese Ausarbeitung folgt der Stoffaufteilung der Unterrichtsplanung und vertieft die fachlichen Inhalte. Sie ist als zusammenhängender Lehrtext angelegt – ohne Übungsaufgaben, ohne Lernkontrollen und ohne Merksätze.

2 Das Spiralmodell nach Boehm (1988)

2.1 Entstehung und Grundidee

Das Spiralmodell wurde 1988 von Barry W. Boehm vorgestellt und löst ein zentrales Problem klassischer Vorgehensmodelle: Bei Wasserfall und V-Modell wird umfassend vorab geplant, und Anforderungen werden früh festgeschrieben. Bei komplexen, innovativen oder forschungsnahen Vorhaben sind aber genau diese Voraussetzungen nicht gegeben – Anforderungen sind unsicher, Technologien unerprobt, der Lösungsweg ist offen.

Boehms Antwort darauf ist ein iteratives Modell, das den Schwerpunkt explizit auf das Risiko legt. Das Projekt wird nicht in einer geraden Linie von der Anforderungsanalyse bis zur Wartung durchgezogen, sondern in mehreren Schleifen – Spiralwindungen – durchlaufen. Mit jeder Iteration wächst die Reife des Produkts, gleichzeitig sinkt die Unsicherheit.

2.2 Die vier Quadranten je Iteration

Jede Iteration im Spiralmodell durchläuft vier Aktivitätsbereiche, die grafisch als Quadranten dargestellt werden. Der Spiralarm wird von Iteration zu Iteration länger, weil mehr Inhalt erarbeitet wurde.

Quadrant	Aufgaben in dieser Iteration
1. Ziele und Alternativen bestimmen	Festlegen der Ziele dieser Iteration, Identifikation möglicher Alternativen (Architekturen, Komponenten, Lieferanten) und der Randbedingungen (Budget, Termine, Qualitätsanforderungen).
2. Risiken bewerten und behandeln	Identifizieren der zentralen Risiken dieser Iteration, Bewertung nach Eintrittswahrscheinlichkeit und Schaden, Auswahl von Maßnahmen. Häufig kommen Prototypen, Simulationen, Machbarkeitsstudien oder Benchmarks zum Einsatz.
3. Entwicklung und Verifikation	Tatsächliches Erstellen des nächsten Inkrements – etwa eines Prototyps, eines Architekturkonzepts oder eines lauffähigen Teilprodukts. Anschließend wird das Ergebnis verifiziert und validiert.
4. Planung der nächsten Iteration	Bewertung des bisherigen Iterationsergebnisses, Entscheidung über das weitere Vorgehen und Planung der nächsten Spirale. Hier fällt auch die Entscheidung, ob das Projekt fortgesetzt oder abgebrochen wird.

2.3 Risikofokus als Alleinstellungsmerkmal

Das Besondere am Spiralmodell ist der explizite und systematische Umgang mit Risiken. Während im Wasserfallmodell Risiken erst spät – oft erst im Test oder bei der Einführung – sichtbar werden, zwingt das Spiralmodell die Beteiligten in jeder Iteration zur expliziten Auseinandersetzung mit den größten Unsicherheiten.

Die Konsequenz: Wenn sich in einer Iteration zeigt, dass ein zentrales Risiko nicht beherrschbar ist – etwa weil eine Technologie nicht funktioniert oder ein Markt anders reagiert als erwartet – kann das Projekt frühzeitig und mit geringen Verlusten beendet werden. Das Spiralmodell ist damit auch ein Werkzeug der wirtschaftlichen Projektsteuerung.

2.4 Typische Iterationen

In der Praxis wird das Spiralmodell häufig mit Iterationen besetzt, die unterschiedliche Reifegrade des Systems repräsentieren. Eine typische Reihenfolge ist:

- Erste Iteration: Machbarkeitsstudie. Klärung, ob das Vorhaben grundsätzlich realisierbar ist.
- Zweite Iteration: Anforderungsanalyse und erster, einfacher Prototyp.
- Dritte Iteration: Designkonzept, erweiterter Prototyp mit Kernfunktionen.
- Vierte Iteration: Lauffähige Pilotversion mit eingeschränktem Funktionsumfang.

- Weitere Iterationen: Schrittweiser Ausbau bis zum Produktivsystem.

Die genaue Zahl und Ausprägung der Iterationen hängt von der Komplexität des Vorhabens ab. Wichtig ist, dass jede Iteration ein vorzeigbares Ergebnis liefert und in den vier Quadranten dieselben Aktivitäten durchläuft.

2.5 Stärken und Schwächen

Das Spiralmodell entfaltet seine Stärken bei Vorhaben mit hohem technischem oder fachlichem Risiko, etwa Forschungs- und Innovationsprojekten, Pilotsystemen mit unsicheren Anforderungen oder großen Vorhaben, bei denen ein Scheitern in späten Phasen besonders teuer wäre.

Stärken: frühe und kontinuierliche Risikotransparenz, Möglichkeit zum geordneten Projektabbruch, schrittweise Konkretisierung von Anforderungen und Lösung, Integration von Prototypen und Validierungsschritten.

Schwächen: hoher Steuerungs- und Dokumentationsaufwand, anspruchsvolle Risikoanalyse, die qualifizierte Beteiligte voraussetzt, schwierige Aufwandsschätzung am Anfang, weil Iterationszahl und -inhalt offen sind. Bei kleinen, gut planbaren Vorhaben ist das Modell überdimensioniert.

3 Weitere Vorgehensmodelle im Überblick

Neben Wasserfall, V-Modell XT und Spiralmodell haben sich weitere Modelle etabliert, die für das Verständnis der Methodenlandschaft wichtig sind. Sie zeigen, dass es zwischen den klassischen sequenziellen und den rein agilen Ansätzen ein breites Spektrum gibt.

3.1 Prototypenmodell

Das Prototypenmodell baut auf der Idee auf, Anforderungen und Designentscheidungen frühzeitig durch greifbare Zwischenergebnisse zu validieren. Statt umfangreicher schriftlicher Spezifikationen entsteht ein Prototyp – ein lauffähiges Teilsystem oder ein interaktives Mockup –, das den Anwendern zur Bewertung vorgelegt wird.

Es lassen sich verschiedene Prototypen unterscheiden:

- **Wegwerfprototyp (Throwaway Prototype):** Wird nur zur Anforderungsklä rung gebaut und nach Erkenntnisgewinn verworfen. Das eigentliche System wird anschließend von Grund auf neu entwickelt.
- **Evolutionärer Prototyp:** Der Prototyp wird Schritt für Schritt zum Endprodukt ausgebaut. Erfordert von Anfang an saubere Architektur, sonst entsteht technische Schuld.
- **Horizontaler Prototyp:** Zeigt eine breite Funktionsoberfläche, aber ohne tiefe Geschäftslogik. Geeignet zur Klärung der Benutzerführung.
- **Vertikaler Prototyp:** Implementiert einen schmalen Funktionsausschnitt vollständig, einschließlich Datenhaltung und Logik. Geeignet zur Klärung technischer Machbarkeit.

Das Prototypenmodell wird selten als alleiniges Vorgehensmodell eingesetzt, sondern meist innerhalb anderer Modelle als Technik – etwa in der zweiten Iteration eines Spiralmodells oder in der Anforderungsphase eines V-Modells.

3.2 Rational Unified Process (RUP)

Der Rational Unified Process ist ein iterativ-inkrementelles Vorgehensmodell, das in den 1990er-Jahren von der Firma Rational Software (später IBM) entwickelt wurde. Er ist stark dokumentations- und rollengetrieben und verbindet objektorientierte Analyse und Design mit einem strukturierten Projektablauf.

3.2.1 Die vier Phasen

Phase	Schwerpunkte
Inception (Konzeption)	Geschäftsidee, Vision, grobe Anforderungen, Wirtschaftlichkeitsbetrachtung. Am Ende steht die Entscheidung, ob das Projekt fortgesetzt wird.
Elaboration (Entwurf)	Detailanalyse der Anforderungen, Architekturentscheidungen, Identifikation der größten Risiken, lauffähiger Architekturprototyp.

Construction (Konstruktion)	Umsetzung des Großteils der Funktionalität in mehreren Iterationen, jeweils mit lauffähigem Inkrement.
Transition (Übergang)	Einführung beim Anwender, Tests im Produktivumfeld, Schulung, Übergabe in den Betrieb.

3.2.2 Disziplinen und Iterationen

Quer zu den Phasen verlaufen sogenannte Disziplinen wie Anforderungsanalyse, Design, Implementierung, Test, Konfigurationsmanagement und Projektmanagement. In jeder Iteration werden alle Disziplinen berührt, allerdings in unterschiedlicher Intensität – in der Inception dominiert die Anforderungsanalyse, in der Construction die Implementierung.

Der Rational Unified Process gilt heute als schwergewichtig und wird in seiner Reinform selten eingesetzt. Seine Konzepte – iteratives Vorgehen, klare Architekturentscheidungen vor der Massenumsetzung, Risikoorientierung – finden sich aber in vielen modernen Vorgehensweisen wieder.

3.3 Extreme Programming (XP)

Extreme Programming ist eine agile Entwicklungsmethode, die Mitte der 1990er-Jahre von Kent Beck formuliert wurde. Sie konzentriert sich auf die handwerkliche Qualität der Softwareentwicklung und setzt etablierte Praktiken konsequent – „extrem“ – um.

3.3.1 Werte

XP basiert auf fünf Werten: Kommunikation, Einfachheit, Feedback, Mut und Respekt. Diese Werte sind keine Marketing-Floskeln, sondern leiten konkrete Entscheidungen im Projektalltag – etwa die Entscheidung, lieber eine einfache Lösung jetzt zu implementieren und später zu verfeinern, als spekulativ für die Zukunft zu bauen.

3.3.2 Kernpraktiken

- **Pair Programming:** Zwei Entwickelnde arbeiten an einem Rechner. Eine Person schreibt Code, die andere reviewt in Echtzeit. Steigert Codequalität und Wissenstransfer.
- **Test-Driven Development (TDD):** Zuerst wird ein Test geschrieben, der den gewünschten Funktionsausschnitt prüft. Dann der minimale Code, der den Test grün macht. Anschließend Refactoring. Dieser Zyklus wird in Minutenschritten wiederholt.
- **Continuous Integration:** Alle Änderungen werden mehrmals täglich in einen gemeinsamen Stand integriert. Automatisierte Builds und Tests stellen sicher, dass das System stets in einem lauffähigen Zustand ist.
- **Refactoring:** Kontinuierliche Verbesserung der Codestruktur ohne Veränderung des Verhaltens. Ziel ist es, technische Schuld konsequent abzubauen.
- **Kollektives Codeeigentum:** Jede Person darf jeden Codeabschnitt verändern. Vermeidet Engpässe und stärkt das Teamverantwortungsgefühl.

- **Kurze Releases:** Statt langer Releasezyklen werden produktive Software-Stände in kurzen Abständen ausgeliefert – idealerweise mehrmals pro Monat.
- **Planning Game:** Gemeinsame Planung mit dem Kunden anhand von User Stories. Die Geschäftsseite priorisiert nach Wert, die Technik schätzt den Aufwand.

XP ist in seiner ursprünglichen Form weniger verbreitet als Scrum, viele seiner Praktiken sind aber heute Standard – insbesondere TDD, Continuous Integration und Refactoring.

4 Abgrenzung Spiralmodell und Scrum

Auf den ersten Blick wirken Spiralmodell und Scrum ähnlich: Beide arbeiten iterativ, beide setzen auf schrittweise Konkretisierung. Bei genauerem Hinsehen unterscheiden sie sich aber deutlich – nicht nur in den Details, sondern in der Grundphilosophie.

4.1 Gemeinsamkeiten

- Iterativ-inkrementelles Vorgehen: Das System entsteht in mehreren Schritten, nicht in einem großen Wurf.
- Frühe und wiederholte Validierung von Zwischenergebnissen.
- Möglichkeit, auf Veränderungen während des Projekts zu reagieren.
- Fortlaufender Risikoumgang.

4.2 Unterschiede

Aspekt	Spiralmodell	Scrum	Konsequenz
Treiber	Risiko	Wert (Business Value)	Spirale priorisiert das Riskanteste, Scrum das Wertvollste.
Iterationslänge	Variabel, je nach Iterationsziel	Fest (1–4 Wochen Sprint)	Spirale ist flexibler im Takt, Scrum schafft Vorhersagbarkeit.
Rollen	Keine festen Rollen vorgeschrieben	PO, SM, Entwicklungsteam	Scrum normiert Verantwortlichkeiten klarer.
Planung	Pro Iteration neu, mit Quadranten-Logik	Sprint Planning, Backlog Refinement	Scrum hat festere Routinen für Planung.
Endpunkt	Bei akzeptabler Lösung oder Abbruchentscheidung	Wenn das Produkt-Ziel erreicht ist	Spirale macht Abbruch explizit zur Option.
Dokumentation	Eher umfangreich, je Iteration	Schlank, gerade ausreichend	Spirale liefert mehr formale Artefakte.

Das Spiralmodell ist ein risikogetriebenes Phasenmodell mit iterativer Struktur, Scrum ist ein wertorientiertes Rahmenwerk für Produktentwicklung. Beide gehören zur iterativ-agilen Modellfamilie, gewichten aber unterschiedlich.

5 Vergleich klassisch, iterativ, agil

5.1 Die drei Modellfamilien

Vorgehensmodelle lassen sich grob in drei Familien einteilen. Die Grenzen sind fließend, und in der Praxis kommen häufig Mischformen zum Einsatz.

5.1.1 Klassisch (plangetrieben)

Beispiele: Wasserfallmodell, V-Modell XT. Charakteristisch sind ein sequenzieller Phasenablauf, umfangreiche Vorabplanung, ausführliche Dokumentation und feste Phasenübergänge. Anforderungen werden früh festgeschrieben, Änderungen sind teuer.

5.1.2 Iterativ

Beispiele: Spiralmodell, Rational Unified Process, Prototypenmodell. Das System entsteht in mehreren Schleifen, die jeweils eine Anforderungsanalyse, einen Entwurf, eine Umsetzung und eine Bewertung enthalten. Risiken und Anforderungen werden Iteration für Iteration konkretisiert.

5.1.3 Agil

Beispiele: Scrum, Kanban, Extreme Programming. Kurze Iterationen oder kontinuierlicher Fluss, hohe Selbstorganisation des Teams, schlanke Dokumentation, kontinuierliche Auslieferung von Inkrementen. Veränderung wird ausdrücklich willkommen geheißen.

5.2 Vergleichstabelle

Kriterium	Klassisch	Iterativ	Agil
Anforderungen	Stabil, früh vollständig	Werden iterativ konkretisiert	Veränderlich, kontinuierlich verfeinert
Planung	Umfassend vorab	Pro Iteration	Just-in-time
Lieferung	Am Projektende	Inkrement je Iteration	Regelmäßig, kurze Zyklen
Dokumentation	Sehr ausführlich	Pro Iteration angemessen	Gerade ausreichend
Risikoumgang	Spät sichtbar	Iterativ adressiert	Kontinuierlich sichtbar
Kundenkontakt	An Phasenübergängen	An Iterationsenden	Kontinuierlich
Änderungen	Teuer, formell	Möglich, planbar	Erwartet, eingeplant
Rollen	Klar hierarchisch	Definiert, aber flexibel	Selbstorganisiert
Aufwand für Steuerung	Hoch (Vorab-Planung)	Mittel	Mittel (laufend, verteilt)

6 Modellauswahl in der Praxis

Es gibt nicht „das“ richtige Vorgehensmodell. Welches Modell zu einem Projekt passt, hängt von einer Reihe von Faktoren ab, die vor Projektstart bewusst geprüft werden sollten. Eine falsche Modellwahl ist eine der häufigsten Ursachen für Projektprobleme.

6.1 Auswahlkriterien

Kriterium	Was wird betrachtet?
Stabilität der Anforderungen	Sind die Anforderungen früh klar und stabil, oder werden sie sich während des Projekts verändern? Stabile Anforderungen begünstigen klassische Modelle.
Technologisches Risiko	Werden bewährte oder neuartige Technologien eingesetzt? Hohe technische Unsicherheit spricht für das Spiralmodell oder eine prototyporientierte Vorgehensweise.
Projektgröße und -dauer	Kleine, kurze Projekte profitieren von agilen Ansätzen. Sehr große, langlaufende Projekte erfordern strukturierte Vorgehensmodelle, oft in hybrider Form.
Regulatorisches Umfeld	Branchen mit hohen regulatorischen Anforderungen (Medizintechnik, Behörden, Verteidigung, Finanzaufsicht) verlangen umfangreiche Nachweise und tendieren zu klassischen oder iterativen Modellen.
Reife der Organisation	Hat das Unternehmen Erfahrung mit Selbstorganisation und agilen Praktiken? Agile Methoden setzen kulturelle Voraussetzungen voraus.
Verfügbarkeit der Stakeholder	Stehen Anwender und Auftraggeber kontinuierlich für Rückfragen zur Verfügung? Agile Methoden brauchen einen ansprechbaren Product Owner.
Wirtschaftliche Lieferform	Wird das Produkt am Ende übergeben oder in kleinen Schritten ausgerollt? Inkrementelle Auslieferung spricht für agil.
Schnittstellen und Abhängigkeiten	Hängt das Vorhaben stark von anderen Systemen oder Lieferanten ab, die ihrerseits plangetrieben arbeiten, sind Synchronisationspunkte und damit klassische Strukturen oft unumgänglich.

6.2 Typische Zuordnungen aus der Praxis

Die folgenden Beispiele illustrieren, welches Vorgehensmodell für welche Projektsituation typischerweise gewählt wird. Sie ersetzen keine Einzelfallprüfung, helfen aber bei der Orientierung.

Projektsituation	Typisches Vorgehensmodell
Behördliches IT-Vorhaben mit regulatorischen Auflagen und klar definiertem Leistungsumfang	V-Modell XT
Standardisierte Infrastruktur-Migration in einem mittelständischen Unternehmen	Wasserfallmodell oder schlankes V-Modell
Forschungsprojekt zur Untersuchung eines neuen KI-Diagnoseverfahrens	Spiralmodell
Entwicklung eines neuen Mobilprodukts mit unklaren Marktreaktionen	Scrum
Wartung und Weiterentwicklung eines bestehenden Webportals	Kanban
Großes Transformationsvorhaben mit mehreren Teilprojekten unterschiedlicher Reife	Hybrid: klassisches Programm-Management mit agilen Entwicklungsteams
Pilotsystem zur Erprobung einer neuen Architektur, später Ausbau zum Produktivsystem	Spiralmodell oder evolutionärer Prototyp
Eingebettete Software in einem sicherheitskritischen Medizingerät	V-Modell XT (oft mit ergänzendem Risikomanagement nach IEC 62304)

6.3 Hybride Modelle

In der Realität kombinieren viele Organisationen Elemente verschiedener Modelle. Typische Hybridmuster sind:

- **Wasserfall außen, Scrum innen:** Die übergeordnete Projektphasenstruktur, Budgetierung und Berichterstattung folgen einem klassischen Modell. Die einzelnen Entwicklungspakete werden agil mit Scrum umgesetzt.
- **Spiralmodell mit agilen Iterationen:** Die übergeordnete Spirale liefert die strategische Risikosteuerung, einzelne Iterationen werden als Scrum-Sprints durchgeführt.
- **Scaled-Agile-Ansätze:** Rahmenwerke wie SAFe oder LeSS skalieren agile Vorgehensweisen auf große Organisationen und integrieren klassische Planungs- und Steuerungsmechanismen.

Wichtig bei jeder Hybridisierung: Die kombinierten Elemente müssen widerspruchsfrei zusammenpassen. Ein klassisches Reporting in Wochentakten verträgt sich beispielsweise schlecht mit dreiwöchigen Sprintzyklen, wenn die Mechanik nicht aufeinander abgestimmt ist.

7 Zusammenfassung

Tag 2 erweitert den Blick auf Vorgehensmodelle vom rein klassischen, sequenziellen Denken zu iterativen und agilen Ansätzen. Das Spiralmodell von Boehm bildet historisch und konzeptuell die Brücke: Es bringt das Iterieren in die Welt der Vorgehensmodelle und etabliert das Risiko als zentralen Steuerungsfaktor.

Mit Rational Unified Process, Extreme Programming und Prototypenmodell wurden drei weitere Modelle vorgestellt, die jeweils einen anderen Schwerpunkt setzen – RUP auf strukturierte Iterationen mit klaren Phasen, XP auf handwerkliche Qualität und enge Zusammenarbeit, Prototypenmodell auf greifbare Zwischenergebnisse zur Anforderungsklä rung.

Die Modellauswahl ist keine Stilfrage, sondern eine fachliche Entscheidung mit erheblichen Konsequenzen. Stabilität der Anforderungen, technisches Risiko, Projektgröße, regulatorisches Umfeld und organisatorische Reife liefern die wesentlichen Kriterien. Hybride Modelle sind in der Praxis eher die Regel als die Ausnahme – das Verständnis der reinen Modelle ist aber Voraussetzung, um Hybride sinnvoll zusammensetzen zu können.