Curriculum für

Certified Professional for Software Architecture (CPSA)®
Advanced Level

Modul
AGILA

Agile Softwarearchitektur

2025.1-RC1-DE-20250423

Inhaltsverzeichnis

Einführung: Allgemeines zum iSAQB Advanced Level . . . 3
Was vermittelt ein Advanced Level Modul? . . . 3
Was können Absolventen des Advanced Level (CPSA-A)? . . . 3
Voraussetzungen zur CPSA-A-Zertifizierung . . . 3
Grundlegendes . . . 4
Was vermittelt das Modul „AGILA“? . . . 4
Struktur des Lehrplans und empfohlene zeitliche Aufteilung . . . 4
Dauer, Didaktik und weitere Details . . . 4
Voraussetzungen . . . 4
Gliederung des Lehrplans . . . 5
Ergänzende Informationen, Begriffe, Übersetzungen . . . 5
1. Grundlagen agiler Software-Architektur . . . 6
1.1. Begriffe und Konzepte . . . 6
1.2. Lernziele . . . 6
2. Agiles Architekturvorgehen . . . 8
2.1. Begriffe und Konzepte . . . 8
2.2. Lernziele . . . 8
3. Architekturanforderungen in agilen Projekten . . . 10
3.1. Begriffe und Konzepte . . . 10
3.2. Lernziele . . . 10
4. Architekturen im Team entwerfen und weiterentwickeln . . . 12
4.1. Begriffe und Konzepte . . . 12
4.2. Lernziele . . . 12
5. Reflexion und Feedback zu Architekturarbeit im agilen Kontext . . . 14
5.1. Begriffe und Konzepte . . . 14
5.2. Lernziele . . . 14
6. Unterstützende Architektur (Team-Übergreifend) . . . 16
6.1. Begriffe und Konzepte . . . 16
6.2. Lernziele . . . 16
7. Beispiele für agile Architekturarbeit . . . 18
7.1. Lernziele . . . 18
Referenzen . . . 19

Die Abkürzung "e. V." ist Teil des offiziellen Namens des iSAQB und steht für "eingetragener Verein", der seinen Status als juristische Person nach deutschem Recht beschreibt. Der Einfachheit halber wird iSAQB e. V. im Folgenden ohne die Verwendung dieser Abkürzung als iSAQB bezeichnet.

Unresolved directive in curriculum-agila.adoc - include::learning-objectives.adoc[tags=**;DE;!*]

Einführung: Allgemeines zum iSAQB Advanced Level

Was vermittelt ein Advanced Level Modul?

Das Modul kann unabhängig von einer CPSA-F-Zertifizierung besucht werden.

- Der iSAQB Advanced Level bietet eine modulare Ausbildung in drei Kompetenzbereichen mit flexibel gestaltbaren Ausbildungswegen. Er berücksichtigt individuelle Neigungen und Schwerpunkte.
- Die Zertifizierung erfolgt als Hausarbeit. Die Bewertung und mündliche Prüfung wird durch vom iSAQB benannte Experten vorgenommen.

Was können Absolventen des Advanced Level (CPSA-A)?

CPSA-A-Absolventen können:

- eigenständig und methodisch fundiert mittlere bis große IT-Systeme entwerfen
- in IT-Systemen mittlerer bis hoher Kritikalität technische und inhaltliche Verantwortung übernehmen
- Maßnahmen zur Erreichung von Qualitätsanforderungen konzeptionieren, entwerfen und dokumentieren sowie Entwicklungsteams bei der Umsetzung dieser Maßnahmen begleiten
- architekturrelevante Kommunikation in mittleren bis großen Entwicklungsteams steuern und durchführen

Voraussetzungen zur CPSA-A-Zertifizierung

- erfolgreiche Ausbildung und Zertifizierung zum Certified Professional for Software Architecture, Foundation Level® (CPSA-F)
- mindestens drei Jahre Vollzeit-Berufserfahrung in der IT-Branche; dabei Mitarbeit an Entwurf und Entwicklung von mindestens zwei unterschiedlichen IT-Systemen
    - Ausnahmen sind auf Antrag zulässig (etwa: Mitarbeit in Open-Source-Projekten)
- Aus- und Weiterbildung im Rahmen von iSAQB-Advanced-Level-Schulungen im Umfang von mindestens 70 Credit Points aus mindestens drei unterschiedlichen Kompetenzbereichen
- erfolgreiche Bearbeitung der CPSA-A-Zertifizierungsprüfung

Grundlegendes

Was vermittelt das Modul „AGILA“?

Teilnehmer:innen lernen in diesem Modul, Softwaresysteme und -architekturen nach agilen Prinzipien zu entwerfen, weiterzuentwickeln und gemeinsam im Team zu verantworten. Einerseits geht es darum, agile Denk- und Arbeitsweisen auf Architekturarbeit zu übertragen, andererseits darum, Architekturpraktiken wirksam und leichtgewichtig in agile Vorgehensmodelle einzubetten.

Das Modul vermittelt, wie Architekturarbeit in selbstorganisierten Teams gelingen kann – durch geeignete technische, methodische und kommunikative Kompetenzen. Behandelt werden unter anderem kollaborative Entscheidungsprozesse, kontinuierliche Architekturevaluation, Stakeholder-Einbindung sowie die Gestaltung unterstützender Strukturen und Rollen in agilen Organisationen. Diese Inhalte werden sowohl theoretisch vermittelt als auch praktisch geübt.

Struktur des Lehrplans und empfohlene zeitliche Aufteilung

Inhalt	Empfohlene Mindestdauer (min)
Grundlagen	90
Agiles Architekturvorgehen	120
Architekturanforderungen in agilen Projekten	210
Architekturen im Team entwerfen	300
Reflexion und Feedback	120
Unterstützende Architektur (Team-Übergreifend)	120
Beispiele für agile Architekturarbeit	120
Summe	1080min = 18h

Dauer, Didaktik und weitere Details

Die unten genannten Zeiten sind Empfehlungen. Die Dauer einer Schulung zum Modul AGILA sollte mindestens 3 Tage betragen, kann aber länger sein. Anbieter können sich durch Dauer, Didaktik, Art und Aufbau der Übungen sowie der detaillierten Kursgliederung voneinander unterscheiden. Insbesondere die Art der Beispiele und Übungen lässt der Lehrplan komplett offen.

Lizenzierte Schulungen zu AGILA tragen zur Zulassung zur abschließenden Advanced-Level-Zertifizierungsprüfung folgende Credit Points) bei:

Methodische Kompetenz:	20 Punkte
Technische Kompetenz:	00 Punkte
Kommunikative Kompetenz:	10 Punkte

Voraussetzungen

Teilnehmerinnen und Teilnehmer **sollten** folgende Kenntnisse und/oder Erfahrung mitbringen:

- Praktische Erfahrung in Entwurf und Entwicklung kleiner bis mittelgroßer Softwaresysteme.
- Praktische Erfahrung im Umgang mit Architekturentscheidungen – deren Erarbeitung, deren

Dokumentation und Kommunikation.

- Erste praktische Erfahrung in agilen Software-Projekten.

Hilfreich für das Verständnis einiger Konzepte sind darüber hinaus:

- Kenntnisse rund um agile Vorgehensmodelle:
  - Das agile Manifest [AgileM2001]
  - Scrum [Schwaber2013]
  - Lean / Kanban [Anderson2010], [Kniberg2011].
- Kenntnisse und erste praktische Erfahrung in der Erarbeitung und Definition von Architekturanforderungen sowie dem Umgang mit unterschiedlichen Stakeholdern von Entwicklungsprojekten.
- Erste praktische Erfahrungen oder Kenntnisse im Bereich der automatisierten Integration und Auslieferung von Software:
  - Kenntnisse und erste Erfahrungen in automatisiertem Testen auf Unit und Integrationsebene
  - Grundkenntnisse der Laufzeitanalyse von Software, etwa Profiling, Tracing, Log-Analyse, Datenanalyse
  - Grundkenntnisse in automatisiertem Build, Integration und Auslieferung von Software.

Gliederung des Lehrplans

Die einzelnen Abschnitte des Lehrplans sind gemäß folgender Gliederung beschrieben:

- **Begriffe/Konzepte**: Wesentliche Kernbegriffe dieses Themas.
- **Unterrichts-/Übungszeit**: Legt die Unterrichts- und Übungszeit fest, die für dieses Thema bzw. dessen Übung in einer akkreditierten Schulung mindestens aufgewendet werden muss.
- **Lernziele**: Beschreibt die zu vermittelnden Inhalte inklusive ihrer Kernbegriffe und -konzepte.

Dieser Abschnitt skizziert damit auch die zu erwerbenden Kenntnisse in entsprechenden Schulungen.

Ergänzende Informationen, Begriffe, Übersetzungen

Soweit für das Verständnis des Lehrplans erforderlich, haben wir Fachbegriffe ins iSAQB-Glossar aufgenommen, definiert und bei Bedarf durch die Übersetzungen der Originalliteratur ergänzt.

1. Grundlagen agiler Software-Architektur

Dauer: 90 Min.

1.1. Begriffe und Konzepte

- Software-Architektur allgemein
- Agiles Manifest: Werte und Prinzipien
- Cynefin-Framework
- 5 Revolutions of Software Engineering
- Cargo Cult Agilität
- Agile Vorgehensmodelle inkl. Scrum und Lean
- Agiler Umgang mit Software-Architektur
- Software-Architektur in agilen Praktiken/Werkzeugen verankern

1.2. Lernziele

LZ 1-1: Die Bedeutung agiler Ideen für Architekturarbeit kennen und erläutern können

- Definitionen von Softwarearchitektur kennen:
  - Welche Definitionen passen zu agilen Umfeldern?
  - Wie grenzt sich Architektur zu Design ab?
- Agile Weltsicht und Prinzipien kennen:
  - Cynefin-Framework und Komplexität als Einfluss auf Vergehensmodelle
  - Relevante Wertpaare und Prinzipien des agilen Manifests (z. B. "Responding to Change over Following a Plan")
  - Typische Fehlanwendungen (z. B. Cargo Cult Agilität) erkennen und kritisch reflektieren.
- Architektur und Agilität im Zusammenspiel verstehen
  - Synergien von Architektur und Agilität
  - Einflüsse von Scrum, Lean und agilen Denkmodellen auf Architekturentscheidungen
  - Grundhaltung agiler Architekturarbeit: frühes Feedback, kollaborative Entscheidungsfindung, evolutionäre Weiterentwicklung über Hypothesen

LZ 1-2: Aufgaben der Architekturentwicklung und deren Abwandlung im agilen Umfeld kennen

- Klärung wie die folgenden, im CPSA-Foundation-Lehrplan definierten, Aufgaben der Architekturentwicklung im agilen Umfeld wahrgenommen werden:
  - Anforderungen und Randbedingungen klären, hinterfragen und bei Bedarf verfeinern.
  - Strukturentscheidungen hinsichtlich Systemzerlegung und Bausteinstruktur treffen, dabei Abhängigkeiten und Schnittstellen zwischen den Bausteinen festlegen.
  - Übergreifende technische Konzepte entscheiden (beispielsweise Persistenz, Kommunikation, GUI) und bei Bedarf umsetzen.

- Software-Architektur auf Basis von Sichten, Architekturmustern und technischen Konzepten kommunizieren und dokumentieren.
- Umsetzung und Implementierung der Architektur begleiten, Rückmeldungen der beteiligten Stakeholder bei Bedarf in die Architektur einarbeiten, Konsistenz von Quellcode und Software-Architektur prüfen und sicherstellen.
- Software-Architektur bewerten, insbesondere hinsichtlich Risiken bezüglich der geforderten Qualitätsmerkmale.
- Wenig Änderung an den Themen der Architekturarbeit, jedoch häufig Anpassung der Arbeitsweise bei der Bearbeitung von Architekturfragen (Aufgabenverteilung, Entscheidungsfindung, Ergebniskommunikation etc.).
- Mapping der wichtigsten Veränderungen auf die Inhalte dieses Lehrplans.

LZ 1-3: Architekturaufwand angemessen an Problemstellung und Projekt ausrichten können

- Spannungsfeld zwischen schneller Reaktionsfähigkeit und guter Planbarkeit bzw. Steuerbarkeit.
  - Im negativen Extrem: Spannungsfeld zwischen geringer Reaktionsfähigkeit (Starre) und schwerer Beherrschbarkeit (Chaos)
- Gerade genug Architekturarbeit und unterstützende Konzepte - Arbeitseinsatz auf relevante Probleme beschränken:
  - Kriterien zur Bewertung von Architekturrelevanz und zum gezielten Einsatz architektonischer Aktivitäten
  - Prinzipien aus Lean (z. B. Waste-Reduktion) auf Architekturarbeit
  - Fokussierung von Kommunikation

2. Agiles Architekturvorgehen

Dauer: 120 Min.	

2.1. Begriffe und Konzepte

- Iterativ, inkrementelle Architekturentwicklung: Risikogetriebene Architekturarbeit und Release-Planung
- Architekturarbeit mit Scrum und Kanban
- Architekturvision und Architecture Communication Canvas
- Walking Skeleton
- Agile Architekturrollen
- Stakeholder-Einbindung

2.2. Lernziele

LZ 2-1: Architekturarbeit um Orientierung zu schaffen agil gestalten können

- Zweck und die Grenzen einer Architekturvision im agilen Kontext erklären
- Architekturvisionen leichtgewichtig entwickeln (Architekturvision oder Architecture Communication Canvas)
- Walking Skeleton mit breiter Architekturüberdeckung planen und gegen ein Minimal Viable Product abgrenzen können
- Mit Architekturvision und Skeleton erste Risiken sichtbar machen und absichern

LZ 2-2: Architekturarbeit kontinuierlich und iterativ weiterführen können

- Risikogetriebene Architekturarbeit: Architektonisches Risiko als bestimmender Faktor für den Einsatz von Architekturpraktiken
- Dringlichkeit von Architekturfragestellungen einschätzen und Entscheidungen verschieben
- Architekturthemen in Backlogs integrieren und dynamisch priorisieren
- Hypothesengetriebenes Vorgehen zur (In)Validierung von Architekturansätzen verwenden
- Optional: Architekturarbeit in Kanban-Systeme einbetten und über Flussmetriken (z. B. WIP-Limits, Lead Time) steuern

LZ 2-3: Rollenmodelle für Architekten in agilen Projekten kennen

- Rollenmodelle für Architekten kontextabhängig auswählen. Beispielsweise:
  - Architecture Owner/unterstützende Architekt:in als Enabler, Coach und Moderator:in
  - Architektur-Champions
  - Selbstorganisierte Teams ohne explizite Architektenrolle
- Die Verantwortungsteilung im Scrum-Team (PO, SM, Dev-Team) mit Blick auf Architekturaufgaben verstehen
  - Das Zusammenspiel von Dev-Team und ggf. Architekt:innen mit Product Ownern bei der Klärung und Priorisierung von Architekturthemen darstellen

LZ 2-4: Möglichkeiten Stakeholder in Architekturarbeit zu involvieren kennen

- Methodische Andockpunkte für die Zusammenarbeit schaffen
- Stakeholder motivieren zur Architektur beizutragen
- Architekturaspekte und -auswirkungen für fachliche Ansprechpartner verständlich machen, z. B. mit Hilfe von Visualisierungen, Storytelling, Canvas-Modellen

3. Architekturanforderungen in agilen Projekten

Dauer: 210 Min.	

3.1. Begriffe und Konzepte

- Zusammenarbeit mit Kunden und anderen Stakeholdern: Kommunikationsebenen, Architekturprobleme zielgruppenorientiert ausdrücken, Anforderungen und Wünsche in Architekturkonzepte übersetzen
- Qualitätsmerkmale und ISO/IEC 25010
- Qualitätsanforderungen in agilen Prozessen
- Technische Schulden
- Architektonische Risiken inkl. Erhebung
- Anforderungspflege, Refinements, Schätzung und Priorisierung von Architekturanforderungen
- Dringlichkeit von Architekturthemen und der letzte vernünftige Moment

3.2. Lernziele

LZ 3-1: Qualitätsanforderungen zielgruppengerecht formulieren können

- Qualitätsmerkmale (z. B. Wartbarkeit, Performance, Sicherheit) anhand von ISO/IEC 25010 benennen und differenzieren Qualitätsanforderungen in unterschiedlichen Abstraktionsebenen formulieren (z. B. als Qualitätsziel, Qualitätsszenario, Akzeptanzkriterium, …)
- Architektonische Probleme und technische Kompromisse in qualitativen Aussagen ausdrückbar machen

LZ 3-2: Agile Konzepte für Architekturanforderungen nutzen können

- Stories mit qualitativen Aspekten erweitern (Akzeptanzkriterien)
- Architekturanforderungen in Backlogs verankern (z. B. als Qualitätsgeschichte, Enabler-Story oder Tech Task). Vor- und Nachteile der Konzepte verstehen.
- Architekturthemen so strukturieren und schneiden, dass sie in agilen Iterationen bearbeitbar sind
- Definition of Done (DoD) für qualitative Zielsetzungen verwenden

LZ 3-3: Iterative Ansätze zur stetigen Erhebung von Architekturanforderungen anwenden können

- Laufende Zusammenarbeit mit Kunden(-vertretern):
  - Nutzen für fachliche Ansprechpartner herausstellen
  - Techniken zur effektiven, regelmäßigen Einbindung fachlicher Ansprechpartner
- Refinements, Reviews, Risikoworkshops richtig einsetzen
- Technische Schulden auf Architekturebene erkennen, erfassen und verarbeiten
- Statische Analysen, Tests und Monitoringdaten als Quelle für neue Architekturaufgaben nutzen

LZ 3-4: Architekturarbeit für Refinements aufbereiten können

- Architekturanforderungen regelmäßig reflektieren, präzisieren und neu priorisieren

- Geeignete Techniken zur gemeinsamen Schätzung nutzen (z. B. T-Shirt Sizing, Magic Estimation, Risk-Based Sizing)
- Methoden zur Priorisierung architektonischer Themen einsetzen
- Arbeit in Refinements und vorbereitende Arbeiten planen können

LZ 3-5: Dringlichkeit als Treiber für Architekturarbeit kennen und erläutern können

- Späte Architekturentscheidungen und offene Optionen als Wert (Real Options Theorie)
- Ansätze, um dringend notwendige Architekturentscheidungen erkennen zu können (z. B. wachsender Implementierungsaufwand, Integrationsthemen)
- Die Abwägung zwischen „schnell entscheiden“ und „gezielt verzögern“ reflektieren

4. Architekturen im Team entwerfen und weiterentwickeln

Dauer: 300 Min.	

4.1. Begriffe und Konzepte

- Gruppenentscheidungsverfahren: Konsensieren, Veto-Verfahren
- Moderation von Entscheidungsverfahren
- Konzeptionelle Integrität bei mehreren Entscheidern
- Vertikale Architekturstile (im Gegensatz zu systemweiter Schichtung)
- Architekturentscheidungen als Prozess
- Just-in-Time-Entscheidungen & Set-Based Design
- Architekturprinzipien
- Architekturwand
- Architecture Decision Records (ADRs)

4.2. Lernziele

LZ 4-1: Methoden zum Treffen von Entscheidungen in Gruppen anwenden können

- Methodische Grundlagen um (gleichberechtigte) Gruppenmitglieder effektiv zu Entscheidungen zu führen:
  - Wahl- und Veto-Verfahren
  - Konsensieren/Konsens
  - Aufteilung von Entscheidungsvorbereitung und Entscheidung
  - Dynamik der Delegation von Entscheidungskompetenz
  - Anwendung im Zusammenspiel mit agilen Vorgehen (Scrum)
  - Gruppendynamische Aspekte und Rollenklarheit im Architekturkontext

LZ 4-2: Gruppen und Teams bei Entscheidungen begleiten können

Um in Gruppen effektiv entscheiden zu können, sind methodische und kommunikative Aspekte wichtig: - Entscheidungsprozesse in Gruppen methodisch begleiten und moderieren - Mit Einwänden und Blockaden konstruktiv umgehen - Diskussionen zielgerichtet strukturieren (z. B. durch strukturierte Entscheidungsformate, Facilitation-Techniken)

LZ 4-3: Rahmenbedingungen für Teamentscheidungen schaffen können

- Rollen, Verantwortlichkeiten und Entscheidungsräume klar definieren
- Architekturprinzipien als Leitplanken für kohärente Entscheidungen etablieren
- technische Voraussetzungen für lokale Architekturentscheidungen erkennen und gestalten
- Risiken verteilter Entscheidungshoheit (z. B. Redundanz, Inkonsistenzen) benennen

LZ 4-4: Architekturentscheidungen als Prozess verstehen und just-in-time treffen können

- Architekturentscheidungen als iterativen Entscheidungsprozess verstehen (offen → in Bearbeitung → entschieden → veraltet)
- Konzept des letzten vernünftigen Moments richtig anwenden
- Technische Möglichkeiten, um die Festlegung von architektonischen Fragestellungen zu verschieben
- Architektonische Fragestellungen in Teilentscheidungen auftrennen
- Techniken um dringende und risikoreiche Entscheidungen zu bearbeiten (Set based Design)

LZ 4-5: Möglichkeiten Architekturentscheidungen in agilen Projekten zu kommunizieren kennen

- Architekturentscheidungen in Meetings und informellen Settings sichtbar machen (z. B. Architekturwand, Team-Stand-Ups, Reviews)
- Den Nutzen von leichtgewichtigen Bewertungsformaten erklären (siehe auch Kapitel 5)
- Erkennen, wann eine formalisierte, dokumentierte Kommunikation erforderlich ist

LZ 4-6: Architekturentscheidungen nachvollziehbar dokumentieren können

- Zweck und Aufbau von Architecture Decision Records (ADRs) erklären
- ADRs einsetzen, um Transparenz, Nachvollziehbarkeit und Teambeteiligung zu fördern
- Unterschiede zu klassischen Architekturdokumentationen verstehen (lightweight, dezidiert, inkrementell erstellt)
- Möglichkeiten kennen, wie ADRs im Team eingeführt werden können (z. B. Wiki, Markdown, arc42).

5. Reflexion und Feedback zu Architekturarbeit im agilen Kontext

Dauer: 120 Min.	

5.1. Begriffe und Konzepte

- Leichtgewichtige Architekturbewertung (DCAR, LASR)
- Root-Cause-Analyse
- Testen qualitativer Systemeigenschaften
- Qualitätsindikatoren und Metriken
- Fitness Functions

5.2. Lernziele

LZ 5-1: Techniken zur gemeinsamen Reflexion von Architekturentscheidungen kennen

- Iterativ und leichtgewichtig durchführbare Bewertungsformate auswählen und anwenden:
  - DCAR - Decision Centric Architecture Reviews - für ADR-basierte, iterative Reviews
  - LASR - Lightweight Approach for Software Reviews - als leichtgewichtige Risiko-Sessions
  - Realitätscheck für Architekturziele um Architekturlücken zu identifizieren
- Kleine, iterative Reviews moderieren und strukturieren
- Feedback zu Architekturideen konstruktiv und zielgerichtet formulieren (Feedbackregeln, Moderationstechniken)

LZ 5-2: Ursachen für Architekturprobleme gezielt finden können

- Ursachenanalyse in Gruppen durchführen (z. B. mit Root Cause Analysis, 5-Why)
- Symptome, Ursachen und Auswirkungen von Architekturproblemen trennen
- Zusammenhang zwischen Test- bzw. Messergebnissen und Architekturentscheidungen herstellen

LZ 5-3: Feedback-Möglichkeiten aus der Umsetzung kennen und Ergebnisse auf Architekturziele zurückführen können

- Testen qualitativer Eigenschaften: Techniken, Werkzeuge und Einbindung in die Entwicklungsaktivitäten
- Statische Qualitätsindikatoren auswählen und nutzen
- Trendanalysen auf Qualitätsindikatoren durchführen und Erkenntnisse ableiten
- Vorteile von Continuous Integration und Continuous Deployment in diesem Zusammenhang nutzen

LZ 5-4: Fitness Functions zur kontinuierlichen Architekturvalidierung verstehen

- Das Konzept von Fitness Functions erklären
- Den Fitness Function Quadrant nutzen, um geeignete Bewertungsstrategien auszuwählen (holistisch vs. atomar, triggered vs. kontinuierlich),
- Tools und Techniken zur Umsetzung identifizieren (z. B. ArchUnit, jQAssistant, Chaos Mesh, Linters, Metrik-Dashboards)

- Fitness Functions in CI/CD-Prozesse, Deployments oder Laufzeitumgebungen integrieren, um automatische Rückmeldungen zur Architekturqualität zu erhalten
- Vorteile von kontinuierlichem Qualitäts-Feedback für die Architekturarbeit (im Team) benennen

6. Unterstützende Architektur (Team-Übergreifend)

Dauer: 120 Min.	

6.1. Begriffe und Konzepte

- Enabling Architecture: Architekt:innen als Unterstützer:innen, nicht Entscheider:innen
- Zentralisierung vs. Dezentralisierung, Governance vs. Autonomie
- Architecture Advice Process
- Architecture Boards, Communities of Practice, Advice Forums
- Architecture Radar, Architekturprinzipien
- Pfad des geringsten Widerstands
- Sozio-technische Systeme, Conway’s Law

6.2. Lernziele

LZ 6-1: Möglichkeiten Architektur als Querschnittsaspekt in agilen Organisationen zu verankern kennen

Architektur als Basis-Fähigkeit von Entwicklern etablieren. Beispielsweise durch: - Offene, projektübergreifende Kommunikation von Best-Practices - Communities-of-Practice zu Architektur aufbauen und lebendig halten - Architektur-Katas als Mittel Architekturwissen zu verbreitern - Architekturverantwortung kommunizieren und übertragen

LZ 6-2: Kräfte der Architekturarbeit zwischen Zentralisierung und Dezentralisierung verstehen

- Spannungsfeld zwischen zentraler Steuerung und dezentraler Entscheidungsautonomie
- Governance-Ansätze mit agiler Denkweise verbinden (Prinzipien statt Regeln)
- Die Grenze zwischen Makro- und Mikroarchitektur aktiv gestalten
- Wichtige Faktoren für erfolgreiche Dezentralisierung benennen (breites Ziel-Verständnis, gemeinsames Architekturverständnis, zeitnahes Feedback, Transparenz & Cross-Pollination, …)

LZ 6-3: Zentrale Architekturstrukturen in Organisationen einordnen und reflektieren

- Verschiedene zentrale Architekturstrukturen beschreiben (z. B. Architecture Board, Advice Forum, virtuelle Architekturgruppen)
- Deren Nutzen, Risiken und sinnvolle Einsatzkontexte bewerten
- Formate für Wissensteilung, Beratung und Standardsetzung an den Bedarf agiler Organisationen anpassen
- Zwischen kontrollierenden und unterstützenden Strukturen unterscheiden.

LZ 6-4: Zusammenhang zwischen Architekturstilen und agiler Architekturarbeit verstehen

- Den Einfluss architektonischer Stile auf Teamautonomie und Veränderbarkeit erläutern (z. B. Microservices vs. Layered Architectures, Self-contained Systems vs. Modulare Monolithen)
- Verstehen, wie Architekturstil, Organisationsstruktur und lebbare Vorgehensmodelle sich gegenseitig beeinflussen.

LZ 6-5: Architekturkonzepte zur Förderung lokaler Entscheidungsmöglichkeiten etablieren können

- Maßnahmen zur Förderung lokaler technischer Entscheidungen identifizieren und einführen
- Einen "Pfad des geringsten Widerstands" aktiv gestalten
- Die Risiken übermäßiger Autonomie einschätzen (z. B. Wildwuchs, redundante Logik, technologische Fragmentierung)
- Passende Architekturprinzipien und Rahmenbedingungen entwickeln, um Balance zu schaffen

LZ 6-6: Architekturentscheidungen beratend begleiten können

- Die Grundidee und Phasen des Architecture Advice Process anwenden
- Zwischen Entscheidungsvorbereitung, -begleitung und -verantwortung differenzieren,
- Geeignete Formate für beratende Architekturarbeit wählen (z. B. Pairing, Shadowing, Reviews, Konsens-Formate, ADR-Workshops)
- Als Coach statt als Entscheider agieren – mit Fokus auf Befähigung und Selbstverantwortung der Teams

LZ 6-7: Sozio-technische Systeme verstehen und architektonisch berücksichtigen können

- Die Wirkung von Teamstrukturen auf Architektur (und umgekehrt) beschreiben
- Team-Definitionen nach Team-Topologies kennen und verstehen wie diese architektonisch unterschiedlich begleitet werden können
- System-Strukturierung und Team-Strukturierung hinterfragen können und Annäherungen herausarbeiten (z.B. mit Fracture Planes)

7. Beispiele für agile Architekturarbeit

Dauer: 120 Min.	

Anmerkung: Beispiele für reale Ausprägungen agiler Architekturarbeit können durch Trainer und Schulungsanbieter individuell ausgesucht werden.Aufgrund der bei realen IT-Systemen oftmals geltenden Geheimhaltungs- oder Vertraulichkeitsregelungen stellt der iSAQB e. V. Schulungsanbietern und Trainern frei, Beispiele abstrahiert oder auszugsweise in Schulungen zu zeigen.

7.1. Lernziele

LZ 7-1: Beispiele für Entscheidungsverfahren in agilen Projekten kennen und nachvollziehen

Teilnehmer sollen an mindestens zwei Beispielen nachvollziehen können, wie agile Projekte unterschiedlicher Größe Architekturentscheidungen treffen und kommunizieren. Dazu sollten in Schulungen die relevanten Meetings, Rollen und Entschiedungsverfahren beschrieben werden, sowie die konkrete Einbettung in den agilen Prozess des jeweiligen Projekts deutlich werden.

LZ 7-2: Beispiele für agil ausgeprägte Architekturanforderungen kennen und nachvollziehen

- Teilnehmer sollen mindestens ein Beispiel für die Formulierung von Architekturanforderungen und deren Einbettung in agile Anforderungsartefakte (wie Stories oder Backlogs) nachvollziehen können.
- Teilnehmer sollen Beispiele nachvollziehen, die gemeinsame Arbeit an Architekturanforderungen in realen Projekten zeigen (Refinement, Planning etc.).

LZ 7-3: Physische Ausprägungen agiler Kommunikationsansätze kennen

Teilnehmer sollen in Beispielen sehen und nachvollziehen können wie agile Projekte Architekturinformationen und -entscheidungen kommunizieren (Fotos von Architekturwänden, Layouts von Informationsradiatoren, Beispiele von Architektur-Radars etc.).

LZ 7-4: Verschiebung von Architekturentscheidungen nachvollziehen können

Teilnehmer sollen in einem realen Beispiel die Verschiebung von Architekturentscheidungen kennenlernen und nachvollziehen können, sowie sehen, wie der Entscheidungsprozess sichtbar gemacht werden kann (z. B. durch Status "Offen – In Diskussion – Getroffen – Veraltet").

LZ 7-5: Beispiele für agil organisierte Architekturgruppen kennen und nachvollziehen

Teilnehmer sollen in mindestens zwei realen Beispielen die Organisation von Architekturcommunities oder -gilden kennenlernen und nachvollziehen können, sowie die Aktivitäten und Techniken kennenlernen die Unternehmen einsetzen, um diese Gruppen zu etablieren und aktiv zu halten. Formate wie Architecture Advice Forums, Peer Reviews, Radar-Workshops oder Reviews sollten plastisch illustriert werden.

Referenzen

Dieser Abschnitt enthält Quellenangaben, die ganz oder teilweise im Curriculum referenziert werden.

A

- [AgileM2001] agilemanifesto.org – The Agile Manifesto. http://agilemanifesto.org/principles.html
- [Anderson2010] David Anderson: "Kanban", Blue Hole Press, 2010, Sequim, WA
- [Appelo2010] Jurgen Appelo: "Management 3.0: Leading Agile Developers, Developing Agile Leaders", Addison Wesley 2010

E - [] Murat Erder, Pierre Pureur, Eoin Woods: “Continuous Architecture in Practice”, Addison-Wesley 2021

F

- [Fairbanks2010] George Fairbanks: "Just Enough Software Architecture: A Risk-Driven Approach", Marshall & Brainerd 2010
- [Ford2017] Neal Ford, Rebecca Parsons, Patrick Kua: “Building Evolutionary Architectures”, O’Reilly Media 2017

H

- [Harmel-Law2024] Andrew Harmel-Law: "Facilitating Software Architecture", O’Reilly Media 2024

K

- [Kniberg2011] Henrik Kniberg: "Lean from the Trenches: Managing Large-Scale Projects with Kanban", Pragmatic Bookshelf 2011

L

- [Larman2008] Craig Larman, Bas Vodde: "Scaling Lean & Agile Development: Thinking and Organizational Tools for Large-Scale Scrum", Addison Wesley 2008

R

- [Richardson2005] Jared Richardson, William A. Gwaltney: “Ship It!: A Practical Guide to Successful Software Projects”, Pragmatic Bookshelf 2005
- [Richards2020] Mark Richards, Neal Ford: “Fundamentals of Software Architecture”, O’Reilly Media 2020

S

- [Skelton2019] Matthew Skelton, Manuel Pais: “Team Topologies: Organizing Business and Technology Teams for Fast Flow”, IT Revolution Press 2019
- [Schwaber2013] Ken Schwaber, Jeff Sutherland: "The Definitive Guide to Scrum: The Rules of the Game", http://www.scrumguides.org

T

- [Toth2025] Stefan Toth: "Vorgehensmuster für Softwarearchitektur: Kombinierbare Praktiken in Zeiten von Agile und Lean", 4. Auflage, Hanser Verlag 2025

V

- [Vigenschow2012] Uwe Vigenschow, Björn Schneider, Ines Meyrose: "Soft Skills für IT-Berater: Workshops durchführen, Kunden methodisch beraten und Veränderungen aktiv gestalten", dpunkt.verlag 2012