Curriculum für

Certified Professional for Software Architecture (CPSA)®
Advanced Level

Modul
ARCEVAL

Architekturbewertung

2022.1-DE-20230413

Inhaltsverzeichnis

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.

Verzeichnis der Lernziele

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

Struktur des Lehrplans und empfohlene zeitliche Aufteilung

Inhalt	Empfohlene Mindestdauer (min)
1. Grundbegriffe von Architekturbewertung	60
2. Anforderungen in der Architekturbewertung	120
3. Szenario-basierte Bewertung	150
4. Bewertung bestehender System(-teil)e	60
5. Alternative Bewertungsmethoden	90
6. Beispiele	90
Summe	570 (9,5h)

Dauer, Didaktik und weitere Details

Die unten genannten Zeiten sind Empfehlungen. Die Dauer einer Schulung zum Modul ARCEVAL sollte mindestens 2 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 ARCEVAL tragen zur Zulassung zur abschließenden Advanced-Level-Zertifizierungsprüfung folgende Credit Points) bei:

Methodische Kompetenz:	20 Punkte
Technische Kompetenz:	0 Punkte
Kommunikative Kompetenz:	0 Punkte

Voraussetzungen

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

- Grundlagen der Beschreibung von Architekturen mit Hilfe verschiedener Sichten, übergreifender Konzepte, Entwurfsentscheidungen, Randbedingungen etc., wie es im CPSA-Foundation Level vermittelt wird.
- Wünschenswert sind eigene Erfahrungen in der Erstellung und Pflege von Software, insbesondere der Architektur von Software- oder Software-nahen Systemen.

Hilfreich für das Verständnis einiger Konzepte sind darüber hinaus:

- Kenntnis typischer Herausforderungen beim Entwurf von Softwarearchitekturen:
  - Auswahl geeigneter Dokumentationsstrukturen, Notationen, Ergebnistypen (Stakeholderorientierung)
  - Gemeinsame Arbeit an grundlegenden Konzepten der zu erstellenden Software
  - Orientierung von Softwarearchitektur an Anforderungen und Rahmenbedingungen
  - Messung und Evaluation der Qualität von Entwürfen und Entscheidungen

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. Grundbegriffe von Architekturbewertung

Dauer: 30 Min.	Übungszeit: 30 Min.

1.1. Begriffe und Konzepte

Softwareachitektur, Architekturbewertung, übergreifende Konzepte/Aspekte, Architekturziele, nichtfunktionale und funktionale Anforderungen an Systeme, Einflussfaktoren, Bewertungsmethoden.

1.2. Lernziele

LZ 1-1: Grundlagen der Softwarearchitektur verstehen

- Begründen, warum Architekturarbeit grundlegend ist, im Projekt breite Auswirkungen hat und entsprechende Entscheidungen schwer zurückzunehmen sind
- Begründen, warum Architekturarbeit iterativ erfolgen sollte und Feedbackschleifen benötigt
- Begründen, dass der Fokus von Softwarearchitektur auf Qualitätsmerkmalen liegt als Differenzierung gegenüber reiner Funktionalität
- Verstehen die Notwendigkeit von Gruppenprozessen für die Weiterentwicklung von Softwarearchitekturen

LZ 1-2: Nutzen von Architekturbewertung erklären

- (Frühe) Erkennung von zentralen Risiken und Problemen bezüglich der geforderten Qualitätsmerkmale
- Methodische Absicherung von Architekturentscheidungen – Sicherung der langfristigen Nachvollziehbarkeit der Architektur
- Qualifiziertes Feedback zu Architekturentscheidungen
- Generelle Förderung von Kommunikation und Transparenz
  - Verlorenes Vertrauen von Stakeholdern kann rückgewonnen werden
  - Widersprüche in Anforderungen werden aufgedeckt
  - Einordnung und Gewichtung von technischen und konzeptionellen Stärken und Schwächen
- Förderung methodischer Architekturentwicklung und sauberer Architekturdokumentation

LZ 1-3: Anlässe zur Architekturbewertung benennen

- methodische Gründe (als üblicher Teil des Architekturprozesses, …)
- organisatorische Gründe (Strategieentscheidungen, Investitionen, …)
- technische Gründe (Technologiewechsel, Ablösungen, …)
- Unsicherheit (mangelnder Überblick, Betriebsblindheit, …)

LZ 1-4: Voraussetzungen für Architekturbewertung erläutern

- Notwendige Artefakte der Softwarearchitektur und deren angemessenen Detaillierungsgrad in konkreten Projektsituationen
- Rahmenbedingungen

- Konkrete Qualitätsanforderungen
- Dokumentierte Architekturentscheidungen
- Architekturüberblick
- Kooperationsbereitschaft der Stakeholder
- Bereitschaft der Organisation, Budget und Zeit für den Bewertungsprozess bereitzustellen

LZ 1-5: Verschiedene Methoden und Ansätze zur Architekturbewertung voneinander unterscheiden

- Szenarienbasierte Bewertungsmethoden (ATAM, Lightweight ATAM, CBAM, …)
- Entscheidungsbasierte Verfahren (DCAR, PBAR, …)
- Expertenreviews (TARA, informell, …)
- quantitativ, messende Techniken
- Konformitätsanalysen (Architektur, Code)

LZ 1-6: Verstehen, dass verschiedene Methoden einander ergänzen und kombiniert werden können

1.3. Referenzen

[Clements+2002], [Kazman+2000], [Bachmann+2000], [Bass+2003], [Clements+2003], [Kruchten 1995], [Starke 2011]

2. Anforderungen in der Architekturbewertung

Dauer: 60 Min.	Übungszeit: 60 Min.

2.1. Begriffe und Konzepte

Qualität, Qualitätsmerkmale, DIN/ISO 9126, DIN/ISO 25010, Szenarien, Qualitätsbaum, Wechselwirkungen, Taktiken.

2.2. Lernziele

LZ 2-1: Den Zusammenhang von Softwarequalität und Softwarearchitektur verstehen

- Die Bedeutung von qualitativen Anforderungen für die Architektur erklären.
- Die Begriffe "Qualität" und "Qualitätsmerkmal" präzisieren.
- Verschiedene Qualitätsmodelle (z.B. DIN/ISO 25010) interpretieren.
- Eigenschaften wichtiger Qualitätsmerkmale erklären:
  - Grundlegende Herangehensweise zur Behandlung
  - Zusammenhang und Wechselwirkungen von Qualitätsmerkmalen.

LZ 2-2: Qualitätsszenarien erheben

- Arten von Szenarien voneinander unterscheiden:
  - Use-Case-Szenarien
  - Grenzszenarien
  - Stressszenarien
- Erhebungstechniken für Szenarien anwenden:
  - Brainstorming / Quality Storming
  - Interview
  - Ableitung aus NFR-Spezifikation
- Mögliche Bestandteile von Szenarien zur Verfeinerung und Konkretisierung erklären.

LZ 2-3: Qualitätsszenarien dokumentieren

- Unterbringung von Szenarien in Architekturdokumentation beschreiben.
- Arbeiten mit Qualitätsbäumen:
  - Deren Nutzen erklären
  - Eigene Qualitätsbäume erstellen
  - Zur Priorisierung von Qualitätsanforderungen verwenden

LZ 2-4: Architektursichten bedarfsgerecht für die Bewertung auswählen können

- bekannte Sichten auf Softwarearchitekturen verstehen:

    - Kontextsicht
    - Bausteinsicht
    - Laufzeitsicht
    - Verteilungssicht
- Bedeutung der Sichten im Verlauf der verschiedenen Bewertungsmethoden erklären.
- Verwendbarkeit von bestehenden Sichten für die Bewertung einschätzen können.

LZ 2-5: Unterschiedliche Wirkungen der Begriffe „Architekturansatz“ und „Architekturentscheidung“ auf die Haltung von Teilnehmenden und Evaluierenden in Bewertungsworkshops erläutern

LZ 2-6: Geeignete Formen der Entscheidungsdokumentation für die Bewertung auswählen

- ADRs
- Stichpunkte
- Konzepte
- Source-Code-Teile
- Messergebnisse und erhobene Metriken

LZ 2-7: Randbedingungen verstehen

- Typische Kategorien von Rahmenbedingungen kennen:
    - Technische Randbedingungen
    - Organisatorische Randbedingungen
    - Konventionen
- Unterschiede zwischen Randbedingungen und Qualitätsanforderungen erklären:
    - Randbedingungen werden erfüllt oder nicht (binär)
    - Qualitätsanforderungen können teilweise oder auch übererfüllt werden

2.3. Referenzen

[Bass+2003], [Clements+2002], [Kazman+2005], [Barbacci+2003], [Starke 2011]

3. Szenario-basierte Workshops

Dauer: 60 Min.	Übungszeit: 90 Min.

3.1. Begriffe und Konzepte

ATAM, Kompromisse, Risiken, Nicht-Risiken, Sensitive Punkte, Entscheidungen, Rahmenbedingungen, Phasen von ATAM, Stakeholder-Beteiligung, szenariobasierte Durchsprache, Bewertungsworkshop, Discovery Review, Zusammenspiel mit der Entwicklung,Risikocluster, Minderungsmaßnahmen, Kommunikation der Ergebnisse, Reports, Präsentationen, Stärken und Schwächen, Ampel-Scores.

3.2. Lernziele

In diesem Teil lernen die Teilnehmer den Kern des methodischen Vorgehens zur qualitativen Bewertung von Softwarearchitekturen kennen.

LZ 3-1: ATAM (Architecture Tradeoff Analysis Method) verstehen

- Bewertungsworkshops nach ATAM geeignet vorbereiten können
- Die Phasen von ATAM erklären
- Die Schritte der Kernphasen von ATAM (Phase 1 und 2) erklären und auf die Bedürfnisse des eigenen Projekts anpassen
- Beteiligte der einzelnen Phasen benennen
- Inputs für ATAM Workshops kennen und den Schritten zuordnen

LZ 3-2: Qualitative Bewertung von Softwarearchitekturen nach ATAM durchführen

- Vorgehen bei qualitativer Bewertung erklären und durchführen
- Erstellung von Szenarien und Qualitätsbäumen erklären und durchführen
- Ansätze hinsichtlich Szenarien analysieren
- Fragetechniken anwenden können, um die Analyse in Breite oder Tiefe zu lenken und den Fokus zu behalten
- Risiken, Nicht-Risiken, Sensible Punkte und Kompromisse identifizieren und verständlich festhalten können
- Bedeutung von Moderation begründen

LZ 3-3: Direkte und verdichtete Ergebnistypen ableiten

- Risiko-Cluster und Risiko-Gruppen
- Stärken und Schwächen einer Architektur
- Lücken zu den zentralen Qualitätszielen

LZ 3-4: Kommunikationsmittel für die Vermittlung von Ergebnissen beschreiben

- Gegliederte Reports und deren wichtige Teile
- Strukturierte Ergebnispräsentationen

- Empfehlungen und deren Einsortierung in Lücken und Potentiale

LZ 3-5: Einflüsse der Bewertungsergebnisse auf iterative Prozesse verstehen

- Risiken aktiv angehen
- offene Entscheidungen planen
- Unsicherheiten ausräumen

LZ 3-6: Bewertungsprozess an den eigenen Kontext anpassen

- Den unterstützenden Einsatz quantitativer Techniken verstehen
- Methoden zur Verschlankung des Verfahrens kennen (z.B. light weight ATAM)

3.3. Referenzen

[Clements+2002], [Bass+2003], [Bass+2006], [Kazman+2000], [Nord+2003], [Kazman+2005], [Starke 2011]

4. Bewertung bestehender System(-teil)e

Dauer: 40 Min.	Übungszeit: 20 Min.

4.1. Begriffe und Konzepte

Metriken, Messungen, Mathematische Modelle, Umsetzungsprüfung, Ist-Architektur, Soll-Architektur, Pain Points, Werkzeuge.

4.2. Lernziele

LZ 4-1: Bestehende System(teil)e bewerten können

- Unterschiede zur Bewertung von Konzepten und Ideen benennen
- Aktuelle Werkzeuge zur Unterstützung kennen und nach Einsatzzweck kategorisieren.
- Messungen und dynamische Prüfungen im Entwicklungsprozess verankern (beispielsweise als Teil des Check-In-Vorgangs, im Build etc.).

LZ 4-2: Umsetzungsprüfung der Architekturansätze im Source Code verstehen

- Bewertung von Softwarearchitekturen im Hinblick auf ihre Umsetzung durchführen, d. h. die Frage beantworten, in wie weit eine Architektur auch im Code umgesetzt wurde.
- Tools für den Abgleich von Umsetzung und Architekturidee benennen (z.B. Sonargraph, Structure101, ArchUnit)
- Eignung der Tools, um Ist-Architekuren Bottom-Up zu erheben und Abgleiche mit Soll-Architekturen durchzuführen, verstehen

LZ 4-3: Metriken zur Bewertung einsetzen

- Konkrete Metriken für die Überprüfung von Design Best-Practices erklären. (z.B. Distanz, Fan-In/Fan-Out, zyklomatische Komplexität)
- Einsatz von Metriken planen (aus Zielsicht, nicht aus Toolsicht)
- Messungen interpretieren (Default-Einstellungen hinterfragen, auf Trends achten)
- Verstehen, dass Messungen geeignete Ausgangspunkte für tiefere Analysen sind

LZ 4-4: Den Einsatz dynamischer Prüfverfahren planen

- Möglichkeiten der dynamischen Software-Prüfung benennen
- Für dynamische Prüfung geeignete Qualitätsmerkmale kennen.
- Ansätze der statischen Analyse in stark verteilten Umfeldern (Microservices) mit dynamischen Mitteln nachstellen.
- Einflechtung dieser Techniken in größere Bewertungsvorhaben erklären.

4.3. Referenzen

[DeMarco+2006], [Martin 2000], [Sonarqube], [Hello2morrow]

5. Weitere Bewertungsmethoden

Dauer: 45 Min.	Übungszeit: 45 Min.

5.1. Begriffe und Konzepte

Kosten-Nutzen-Bewertungen (CBAM), Entscheidungsbasierte Verfahren (DCAR), Experten-basierte Verfahren (TARA) und andere Bewertungsmöglichkeiten (PBAR, ARID, Pre Mortems, …)

5.2. Lernziele

LZ 5-1: Kosten-Nutzen-Bewertungen als gezielte Maßnahme einordnen können

- Die Cost-Benefit Analysis Method (CBAM) kennen
- Die zu ATAM unterschiedliche Verwendung von Qualitätsszenarien verstehen
- Die Anwendbarkeit und Genauigkeit von CBAM einschätzen und erklären können

LZ 5-2: DCAR (Decision Centric Architecture Reviews) in Grundzügen verstehen

- Die Verwendung von ADRs (Architecture Decision Records) in DCAR kennen
- Grundbegriffe von DCAR (Decision Force, Dependencies Relation Diagram, …) verstehen
- Die Anwendbarkeit von DCAR in agilen und iterativen Prozessen verstehen
- Den Unterschied zur Anwendbarkeit anderer Methoden (wie ATAM) verstehen

LZ 5-3: TARA (Tiny Architecture Review Approach) in Grundzügen verstehen

- Vor- und Nachteile einer Bewertung durch Experten benennen können
- ATAM-Elemente in TARA einordnen und den Zusammenhang verstehen

LZ 5-4: PBAR (Pattern-based Architecture Reviews) in Grundzügen verstehen

- Anwendbarkeit von PBAR verstehen
- Stärken/Schwächen von Musterbasierter Analyse benennen können

LZ 5-5: ARID (Active Review for Intermediate Designs) in Grundzügen verstehen

- Aktiven Review-Ansatz erklären können
- Anwendbarkeit, Aufwand und Nutzen von ARID einordnen können

LZ 5-6: Größere Bewertungsvorhaben strukturieren können

- Faktoren benennen können, die zu aufwendigeren Bewertungvorhaben führen
- Grundelemente qualitativer und quantitativer Bewertungsmethoden kombinieren können, um größere, strukturierte und zielgerichtete Bewertungen zu unterstützen.
- Personalaufwand auf Reviewer- und Teilnehmerseite einschätzen können

LZ 5-7: Ideen für schnelle Reviews von Architekturen erklären können

- Möglichkeiten, qualitätsorientierte Reviews zu verschlanken, verstehen
- Risiko-Sammlungen und Brainstorming-Ansätze benennen können
- Die Skalierung kleiner Bewertungen in fundiertere Betrachtungen verstehen und den Übergang erklären können

5.3. Referenzen

[DeMarco+2006], [Martin 2000], [Sonarqube], [Hello2morrow]

6. Beispiele

Dauer: 90 Min.	Übungszeit: keine

Dieser Abschnitt ist nicht prüfungsrelevant.

6.1. Begriffe und Konzepte

Innerhalb jeder lizenzierten Schulung muss mindestens ein Beispiel für ARCEVAL vorgestellt werden.

Art und Ausprägung der vorgestellten Beispiele können von der Schulung bzw. den Interessen der Teilnehmer abhängen und werden seitens iSAQB nicht vorgegeben.

6.2. Lernziele

Vorstellung, eventuell Erarbeitung und Bewertung mindestens eines Beispiels einer Softwarearchitektur.

LZ 98-1: Was sollen die Teilnehmer verstehen?

Den Ablauf von Bewertungsworkshops und die Bewertungsmöglichkeiten die sich bei realitätsnahen Architekturen ergeben.

LZ 98-2: Was sollen die Teilnehmer kennen?

Einige Beispiele (möglichst) realitätsnaher Bewertungsinputs und Bewertungsergebnisse:

- Szenarien
- Dokumentierte Rahmenbedingungen
- Dokumentierte Entscheidungen
- Architekturüberblick
- Pain Points und sensitive Punkte
- Dokumentierte Kompromisse, Risiken und Nicht-Risiken

6.3. Referenzen

Keine. Schulungsanbieter sind für die Auswahl und Beschreibung von Beispielen verantwortlich.

Referenzen

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

B

- [Bachmann+2000] Bachmann, F., L. Bass, et al.: Software Architecture Documentation in Practice. Software Engineering Institute, CMU/SEI-2000-SR-004.
- [Barbacci+2003] Barbacci, M.R., Ellison, R., et al.: Quality Attribute Workshops (QAWs), Third Edition. Software Engineering Institute, CMU/SEI-2003-TR-016.
- [Bass+2003] Bass, L., Clements, P. und Kazman, R. (2003): Software Architecture in Practice. Addison-Wesley, Reading, Mass.
- [Bass+2006] Bass, L., Nord, R., et al.: Risk Themes Discovered Through Architecture Evaluations. Software Engineering Institute, CMU/SEI-2006-TR-012.

C

- [Clements+2002] Clements, P., R. Kazman, M. Klein: Evaluating Software Architectures – Methods and Case Studies.Addison Wesley, 2002.
- [Clements+2003] Clements, P., F. Bachmann, L. Bass, D. Garlan, J. Ivers et al: Documenting Software Architectures – Views and Beyond. Addison Wesley, 2003.

D

- [DeMarco+2006] DeMarco, T.: Controlling Software Projects: Management, Measurement and Estimation. Prentice Hall, 1986.

H

- [Hello2morrow] Hello2Morrow, online: http://www.hello2morrow.com/

K

- [Kazman+2000] Kazman, R., Klein, M., Clements, P.: ATAM: Method for Architecture Evaluation. Software Engineering Institute, CMU/SEI-2000-TR-004.
- [Kazman+2005] Kazman, R., Bass, L.: Categorizing Business Goals for Software Architectures. Software Engineering Institute, CMU/SEI-2005-TR-021.
- [Kruchten 1995] Kruchten, P.: Architectural Blueprints – The 4-1 View Model of Architecture. IEEE Software November 1995; 12(6), p. 42-50.

M

- [Martin 2000] Martin, R.C.: Design Principles and Design Patterns. White-Paper, 2000.

N

- [Nord+2003] Nord, R.L., Barbacci, M.R., et al.: Integrating the Architecture Tradeoff Analysis Method (ATAM) with the Cost Benefit Analysis Method (CBAM). Software Engineering Institute, CMU/SEI-2003-TN-038.

S

- [Sonarqube] Sonarqube, online: https://www.sonarqube.org/
- [Starke 2011] Starke, G. (2011): Effektive Softwarearchitekturen - Ein praktischer Leitfaden. 5. Auflage 2011, Carl Hanser Verlag, München.