Stefan Zörner

LASR – Die Software-architektur eures Systems in Eigenregie bewerten

Session auf der W-JAX
Konferenz für Java, Architektur- und Software-Innovation
München, 05. November 2025

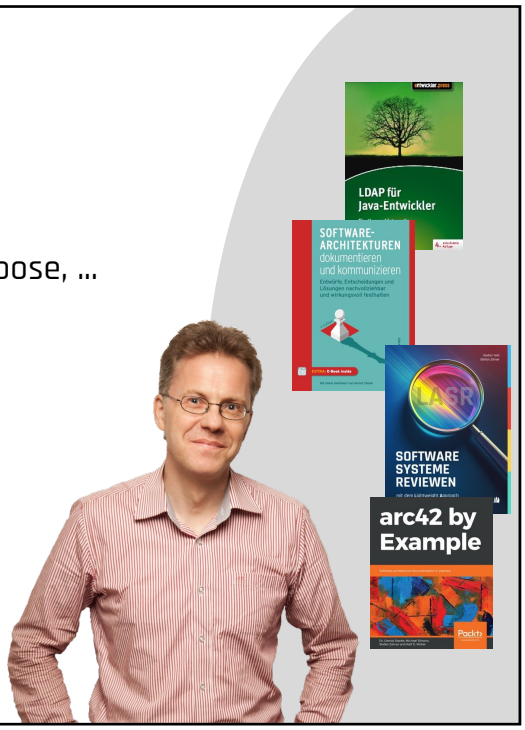
0

Stefan Zörner

- Softwarearchitekt bei embarc in Hamburg
- Vorher Bayer AG, Mummert + Partner, IBM, oose, …

Schwerpunkte

- Methodische Softwarearchitektur (Entwurf, Bewertung, Dokumentation)
- Architektur-Reviews



1

Abstract

Die Softwarearchitektur eures Systems in Eigenregie bewerten.

Mit Architekturbewertungen ist es möglich, Schwächen und Potenziale von Softwarelösungen herauszuarbeiten, Entscheidungen abzusichern und Verbesserungsmaßnahmen zu bewerten. Klassische Analyseansätze aus diesem Umfeld wie ATAM sind fundiert, kommen aber gerade in beweglichen Softwarevorhaben etwas schwergewichtig, mitunter fast zeremoniell daher.

In dieser Session lernt ihr daher mit LASR (Lightweight Approach for Software Reviews) eine leichtgewichtige Herangehensweise kennen. Ihr könnt diese mit eurem Team unmittelbar anwenden, euer Softwaresystem beleuchten und zügig zu ersten Erkenntnissen kommen. Wir greifen auf die Essenzen etablierter Bewertungsmethoden zurück und erarbeiten uns einen roten Faden durch ein Review, inkl. möglicher Vertiefungspunkte für eine höhere Konfidenz im Bewertungsergebnis.

2

00.

Agenda

Die Softwarearchitektur eures Systems in Eigenregie bewerten

01. **Warum leichtgewichtig bewerten?**

02. **Verstehe, was Dich speziell macht**

03. **Durchleuchte die Architektur**

04. **Erhöhe die Konfidenz (bei Bedarf)**

05. **Weitere Informationen**

embarc

3

01.

Warum leichtgewichtig bewerten?

4

Was ist Softwarearchitektur?

Softwarearchitektur :=

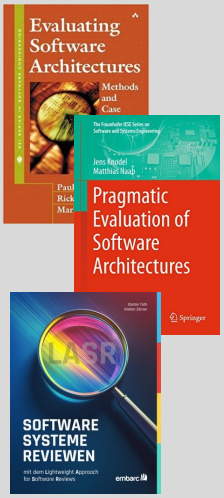
Σ wichtige Entscheidungen

wichtig =
- fundamental (betrifft viele)
- im weiteren Verlauf nur schwer zu ändern
- entscheidend für den Erfolg Eures Softwaresystems

5

Was ist ein Review?

Wörtlich:
Review == etwas **(über)prüfen**, besprechen, ...



Gegenstand („etwas“) in unserem Fall:
Die Architektur(-entscheidungen) eines Softwaresystems

Typischer Begriff in der Fachwelt / -literatur auch:
Architekturbewertung (englisch „Evaluation“)

Kann durch Außenstehende erfolgen – muss aber nicht.

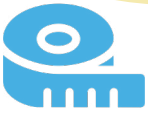
6

Kernelemente eines Reviews

Für eine Bewertung oder ein Review braucht es mindestens

Einen Anlass
Warum bewerten wir?

Einen Gegenstand
Was bewerten wir?



Einen Maßstab
Wonach (wo gegen) bewerten wir?

7

Kernelemente eines Reviews

Für eine Bewertung oder ein Review braucht es mindestens

Einen Anlass
Warum bewerten wir?

Einen Gegenstand
Was bewerten wir?

Einen Maßstab
Wonach (wo gegen) bewerten wir?

8

Typische Anlässe (... findet Ihr Euch wieder?)

Anlass 1 („Neuanfang“)

Eine **Neuentwicklung** steht an und **erste Lösungsansätze** stehen im Raum.

Leitfrage:
Seid ihr und euer Team auf dem **richtigen Weg**?

9

Typische Anlässe (... findet Ihr Euch wieder?)

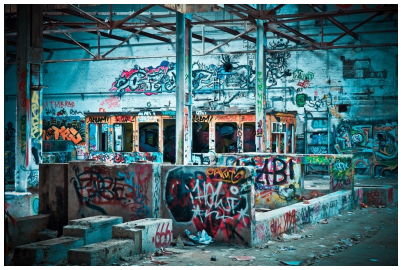
Anlass 2 („Wünsche“)

Unterschiedliche Stakeholder verfolgen **widersprüchliche Ziele** mit eurer Software.

Leitfrage:
Wie **konkretisiert** und **priorisiert** ihr deren Wünsche?

10

Typische Anlässe (... findet Ihr Euch wieder?)



Anlass 3 („Unruhe“)

Das **Management** hat das **Vertrauen** in eure Lösung verloren.

Leitfrage:
Wie gewinnt ihr es zurück und strahlt **Sicherheit** aus?

11

Typische Anlässe (… findet Ihr Euch wieder?)

Anlass 4 („Legacy“)

Größere **Umbaumaßnahmen** in eurer Software stehen an.

Leitfrage:
Wie wählt ihr passende Lösungsansätze **nachvollziehbar** aus?

12

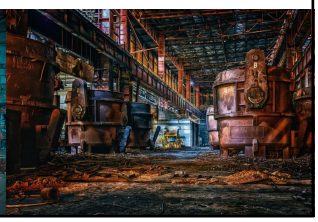
Das Leistungsversprechen von Reviews

In all diesen Situationen unterstützen Review-Ansätze und helfen euch und eurem Team die Leitfragen zu beantworten.

Konkret: Software-Reviews …

- … **decken** Kompromisse und **Risiken** von Softwarelösungen **auf**.
- … **sichern** technische und architektonische **Ideen ab**.



13

Grundsätzliche Ansätze

Quantitative Analyse

Setzt auf Messungen
und Metriken.
In der Regel **Tool-basiert**.

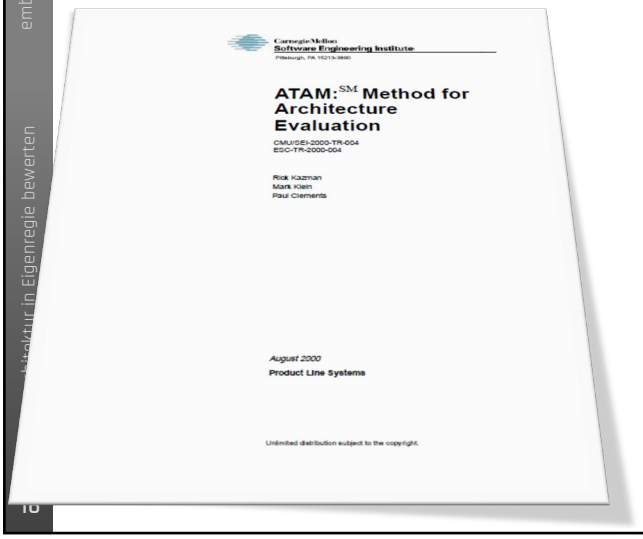
Qualitative Analyse

Setzt auf Diskussion,
Austausch und Durchsprachen.
Oft **Workshop-basiert**.

Bewertungsmethoden (Auswahl)

SAAM Software Architecture Analysis Method	**ARID** Architecture Review for Intermediate Designs
ATAM Architecture Tradeoff Analysis Method	**DCAR** Decision Centric Architecture Review
CBAM Cost-Benefit Analysis Method	**DASE** Decision and Scenario based architecture evaluation
TARA Tiny Architecture Review Approach	**Pre-Mortem** Risk-Brainstorming and Mitigation
PBAR Pattern Based Architecture Review	**LASR** Lightweight Approach for Software Reviews

ATAM



Architecture Tradeoff Analysis Method

- **Akademischer Ursprung**: Carnegie Mellon University, 2000
- Bekannteste Methode zur Bewertung von Softwarearchitektur
- **Qualitativer** Ansatz, Szenarien- und **Workshop**-basiert
- Früh anwendbar, geht von den **Zielen** aus

16

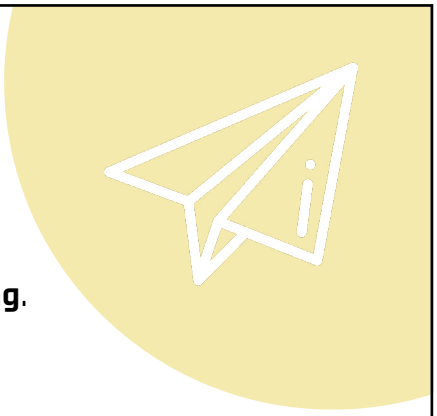
Herausforderungen …

Die Anwendung fundierter Bewertungsmethoden ist mitunter schwierig.

- Der Einsatz erfordert häufig **viele Beteiligte**.
- Es sind oftmals **Vorarbeiten nötig**, beispielsweise die Aufbereitung der Geschäftsziele und das Anfertigen eines Architekturüberblicks.
- Sie liefern **nur Rohergebnisse**, die für eine effiziente Kommunikation aufwendig nachzubearbeiten sind.
- Durchführung und Moderation verlangen einiges ab – die Methoden **unterstützen** dabei **wenig**.

17

Was ist leichtgewichtig?



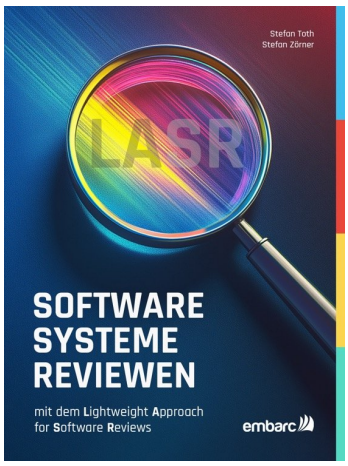
Merkmale eines leichtgewichtigen Reviews

- Die **Anzahl** der Beteiligten / **Stakeholder** ist **gering**.
- **Aufwand** und Dauer sind **überschaubar**.
- **Ergebnisse** liegen vergleichsweise **schnell** vor.

Konkret z.B.

- Entwicklungsteam führt Review **allein** und **ohne Vorbereitung** durch.
- Bereits nach einem halben Tag liegt ein **kommunizierbares Ergebnis** vor.

Erfahrungswissen



Software-Systeme reviewen
mit dem Lightweight Approach
for Software Reviews - LASR

Leanpub

Autoren: Stefan Toth, Stefan Zörner
Verlag: Leanpub, September 2023
Sprache: Deutsch, EPUB, PDF

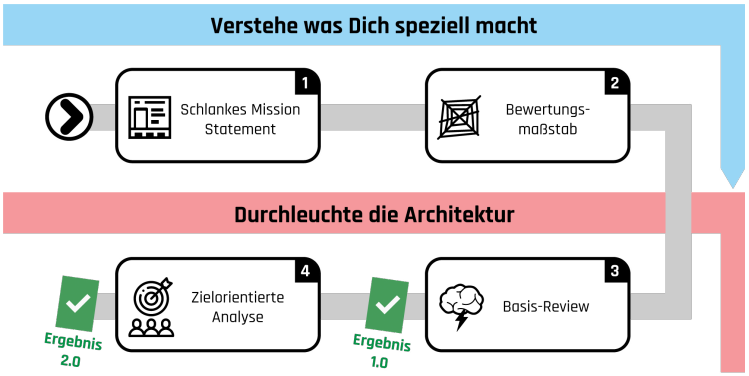
➔ leanpub.com/software-systeme-reviewen/

Ein schlanker Ansatz

LASR (Lightweight Approach for Software-Reviews) ist ein strukturiertes Vorgehen für leichtgewichtige Software-Reviews.

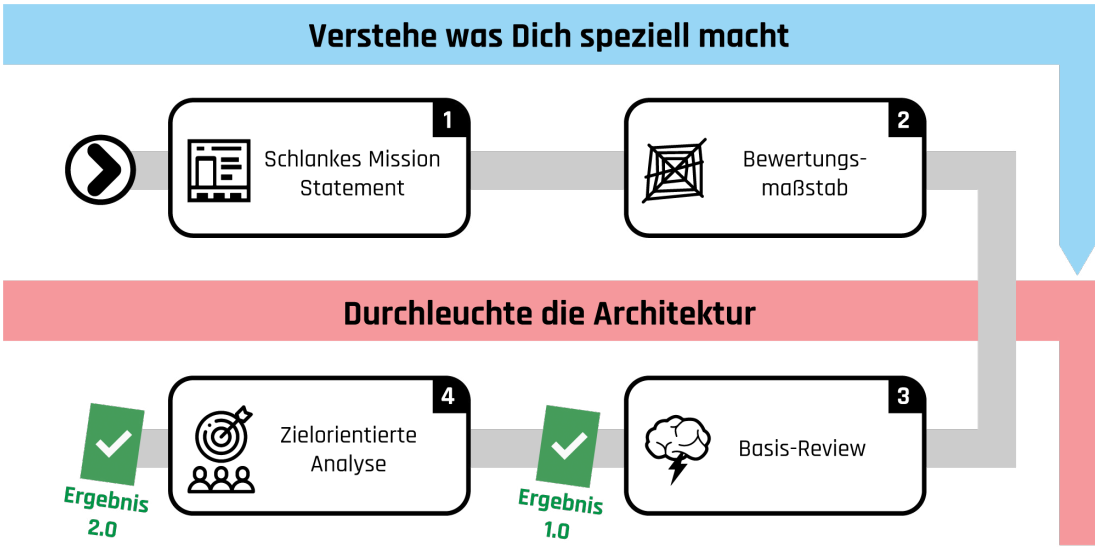
Ablauf LASR-Review

- 2 Tätigkeiten
- 4 Schritte
- Ein frühes erstes Ergebnis



20

Tätigkeiten und Schritte im LASR-Review



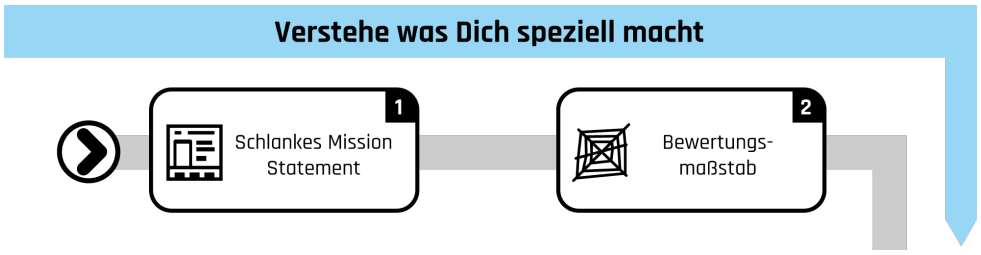
21

02.

Verstehe, was Dich speziell macht

22

Verstehe, was Dich speziell macht



Was ist zu tun?

- Die Aufgabenstellung für das System zu einem prägnanten Mission Statement verdichten
- Die Top 3-5 Ziele des Systems identifizieren
- Jeweiliges Ziel-Level festlegen

23

Kernelemente eines Reviews

Für eine Bewertung oder ein Review braucht es mindestens

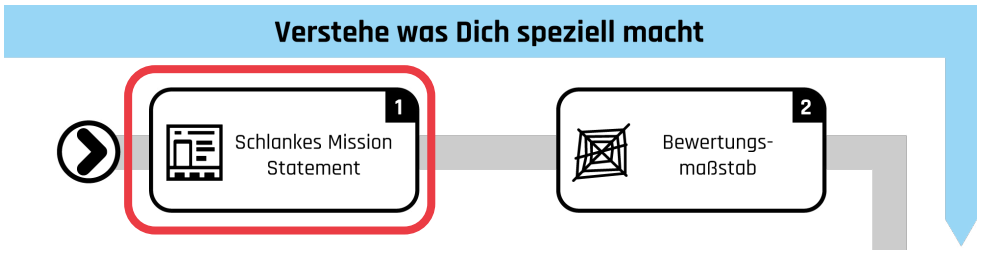
Einen Anlass
Warum bewerten wir?

Einen Gegenstand
Was bewerten wir?

Einen Maßstab
Wonach (wo gegen) bewerten wir?

24

Verstehe, was Dich speziell macht



Was ist zu tun?

- **Die Aufgabenstellung für das System zu einem prägnanten Mission Statement verdichten**
- Die Top 3-5 Ziele des Systems identifizieren
- Jeweiliges Ziel-Level festlegen

25

(1) Schlankes Mission Statement

Fokussiert und **grenzt** das zu betrachtende System **ab** durch Finden sogenannter "Claims" (Ansprüche) der Software.

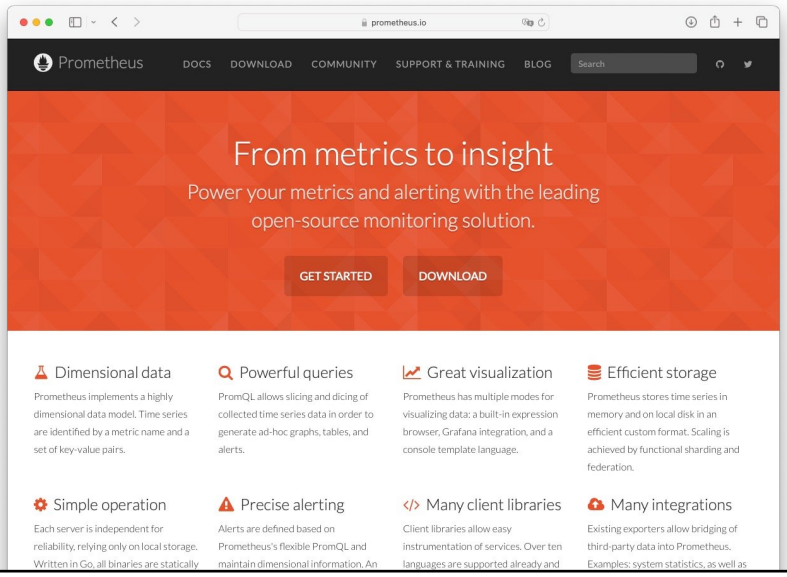
Methodischer Ansatz

Sammeln der Claim(s) mit **Landing Page-Metapher**

Ergebnis des Schrittes

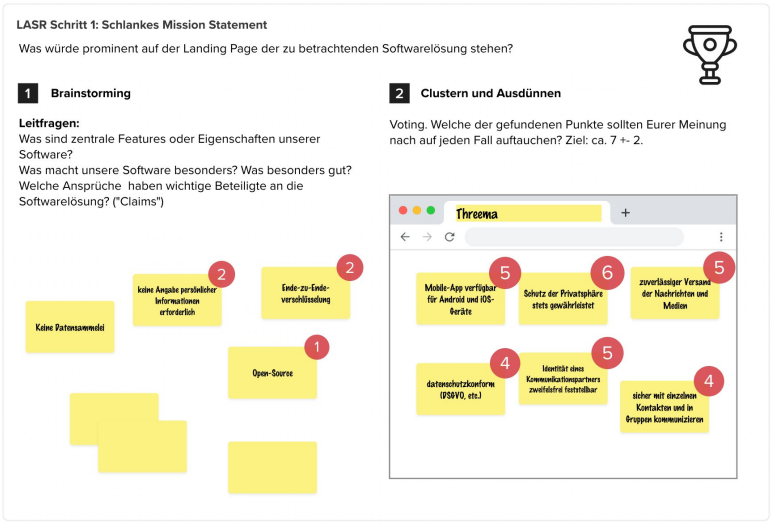
Prägnante Produktidee (Stichpunkte)

Beispiel: Landing Page von Prometheus



→ prometheus.io

Erarbeiten im Workshop

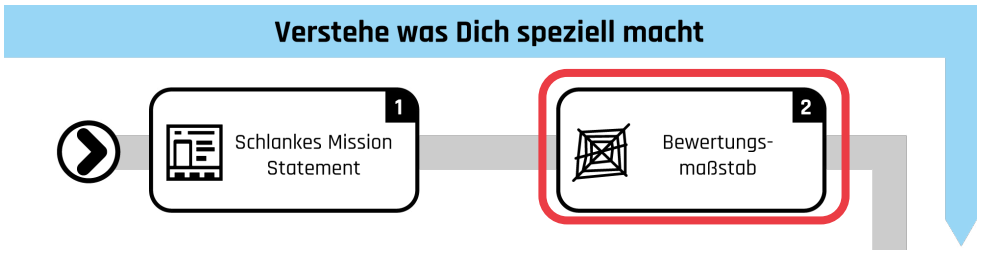


Beispiel im digitalen Whiteboard (Mural)

Quelle: S. Toth, S. Zörner: „Software-Systeme reviewen", Leanpub 2023

28

Verstehe, was Dich speziell macht

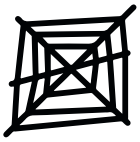


Was ist zu tun?

- Die Aufgabenstellung für das System zu einem prägnanten Mission Statement verdichten
- **Die Top 3-5 Ziele des Systems identifizieren**
- Jeweiliges Ziel-Level festlegen

29

(2) Bewertungsmaßstab



Identifiziert und priorisiert grob die entscheidenden **Qualitätsmerkmale**.

Methodischer Ansatz

Top-5-Challenger zur Zielauswahl

Ergebnisse des Schrittes

3-5 Qualitätsziele inklusive grober Ziel-Einschätzung

30

Qualitätsmerkmale (Begriffe nach ISO 25010:2023)

Funktionale Eignung
(Functional Suitability)

Sind die berechneten Ergebnisse genau genug / exakt, ist die Funktionalität angemessen? ...

Effizienz
(Performance Efficiency)

Antwortet die Software schnell, hat sie einen hohen Durchsatz, einen geringen Ressourcenverbrauch? ...

Kompatibilität
Compatibility)

Ist die Software konform zu Standards, arbeitet sie gut mit anderen zusammen? ...

Benutzbarkeit
(Interaction Capability)

Ist die Software intuitiv zu bedienen, wiedererkennbar, leicht zu erlernen, attraktiv? ...

Zuverlässigkeit
(Reliability)

Ist das System verfügbar, tolerant gegenüber Fehlern, nach Abstürzen schnell wieder hergestellt? ...

Sicherheit
(Security)

Ist das System sicher vor Angriffen? Sind Daten und Funktion vor unberechtigtem Zugriff geschützt? ...

Wartbarkeit
(Maintainability)

Ist die Software leicht zu ändern, erweitern, testen, verstehen? Lassen sich Teile wiederverwenden? ...

Übertragbarkeit
(Flexibility)

Ist die Software leicht auf andere Situationen oder Zielumgebungen (z.B. anderes OS) übertragbar? ...

Betriebssicherheit
(Safety)

Sind Personen, Tiere, Sachen und Umwelt vor Schäden durch die Software geschützt? ...

31

Kernelemente eines Reviews

Für eine Bewertung oder ein Review braucht es mindestens

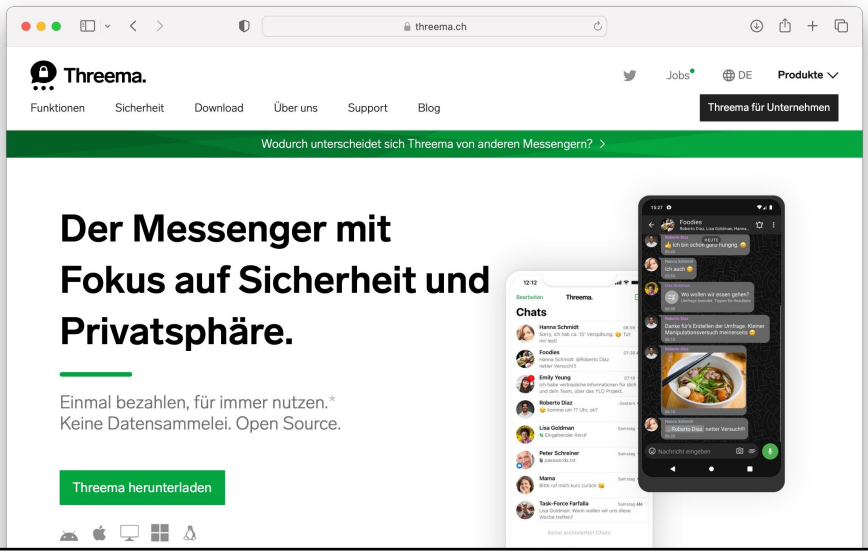
Einen Anlass
Warum bewerten wir?

Einen Gegenstand
Was bewerten wir?

Einen Maßstab
Wonach (wo gegen) bewerten wir?

32

Weiteres Beispiel für eine Startseite



33

Die Qualitätsziele von Threema

Sicherheit	Kommunikations- und Nutzerdaten sind geschützt.
Benutzbarkeit	Einfach zu verwenden.
Zuverlässigkeit	Zuverlässig und effizient im Betrieb.
Kompatibilität	Interoperabel über alle Plattformen hinweg.
Wartbarkeit	Leicht erweiterbar und anpassbar.

Quelle: „Architektur-Porträt: Threema“, Stefan Zörner 2023

34

35

Top-5-Challenger (Unterstützungsmaterial)

Material:

- 14 Karten mit Qualitätsmerkmalen

Vorbereitung:

- Mischt die Karten in einem verdeckten Stapel.
- Deckt 5 Karten zufällig auf und legt sie nebeneinander. (das ist Eure Start Top-5)
- Legt die übrigen 9 in einer 3 x 3 Matrix für alle gut sichtbar offen aus.

Top-5-Challenger, Vorbereitung (Beispiel)

Start Top-5

Kandidaten

Ablauf einer Runde

Einen „Challenger“ nominieren

- Team-Mitglied schlägt Ziel vor, das in Top-5 fehlt.
- Die Gruppe diskutiert darüber.
- Der Challenger verdrängt eine Karte oder landet selbst auf dem Ablagestapel.

38

Nach 2 Runden, Beispiel (Beispiel)

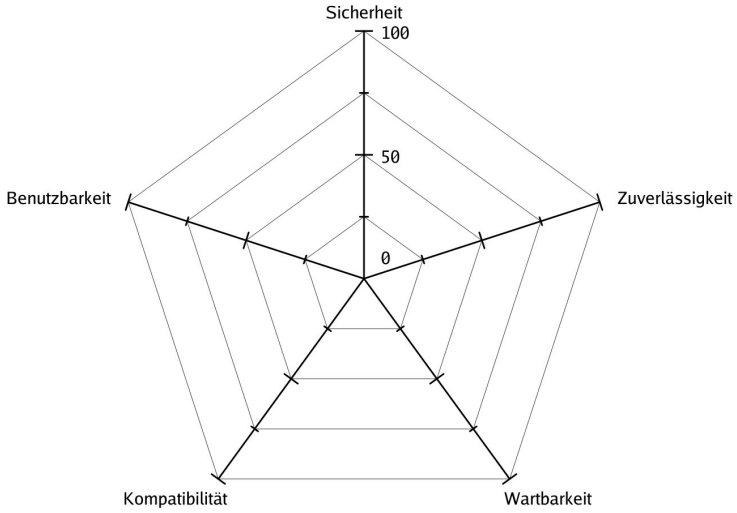
Aktuelle Top-5

Ablagestapel

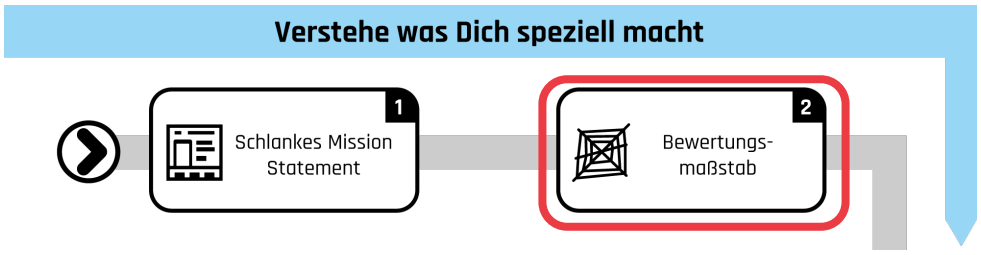
Kandidaten

39

Ergebnis nach diesem Schritt (Beispiel)



Verstehe, was Dich speziell macht

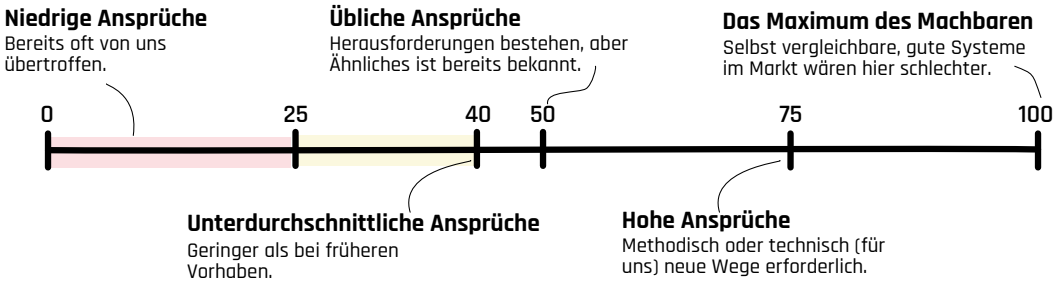


Was ist zu tun?

- Die Aufgabenstellung für das System zu einem prägnanten Mission Statement verdichten
- Die Top 3-5 Ziele des Systems identifizieren
- **Jeweiliges Ziel-Level festlegen**

Zielwerte bestimmen

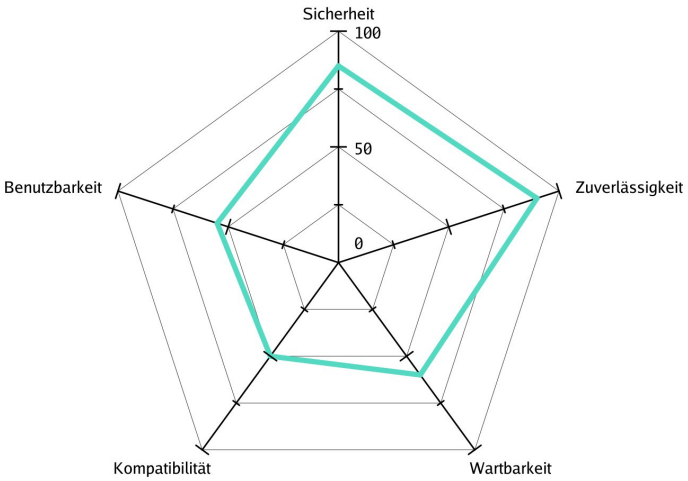
Der Zielwert zwischen 0 und 100 beschreibt jeweils, wie hoch die Erwartungen in dem Bereich sind. Im Vergleich zu anderen Zielen der Top-3-5 und anderen Vorhaben aus dem Kontext des Unternehmens / der Organisation.



Hinweis: Es handelt sich hier um eine Einschätzung, nichts Messbares.

42

Bewertungsmaßstab einzeichnen



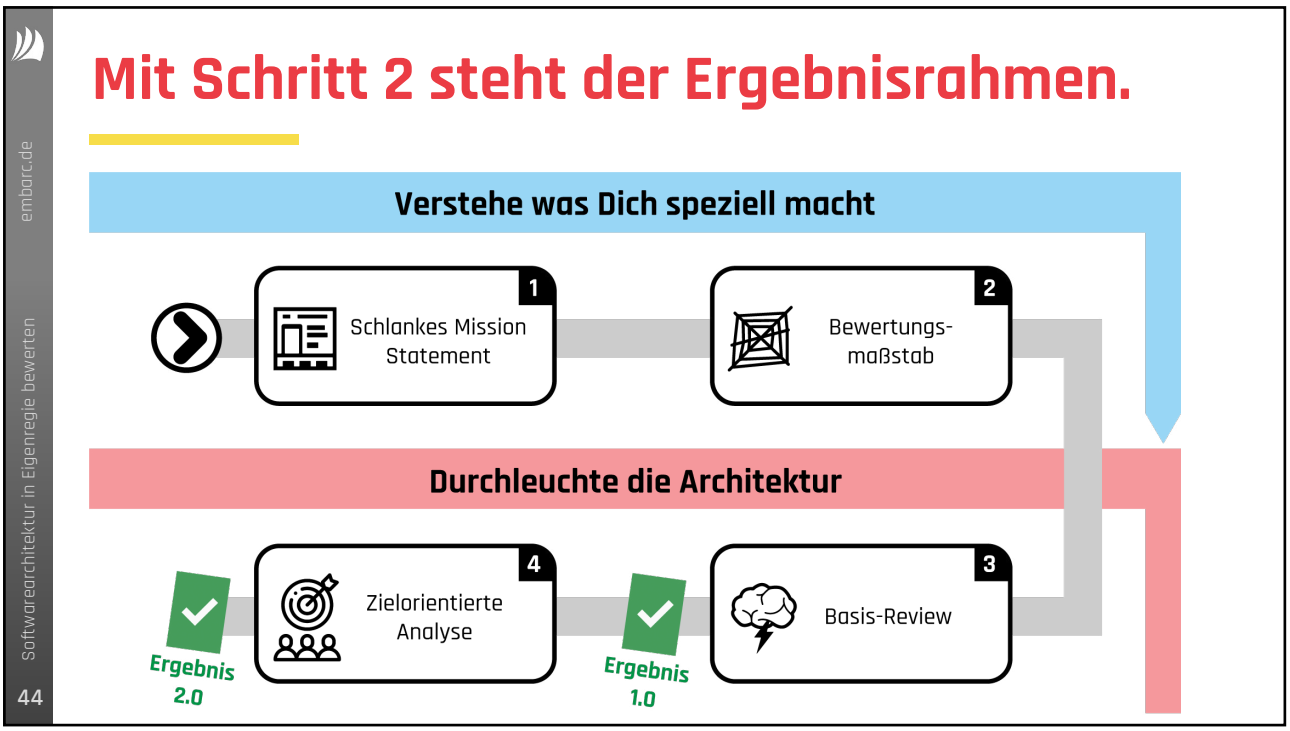
LASR-Ergebnisdiagramm

Die Top-3-5 Qualitätsziele bilden die Achsen des Diagramms.

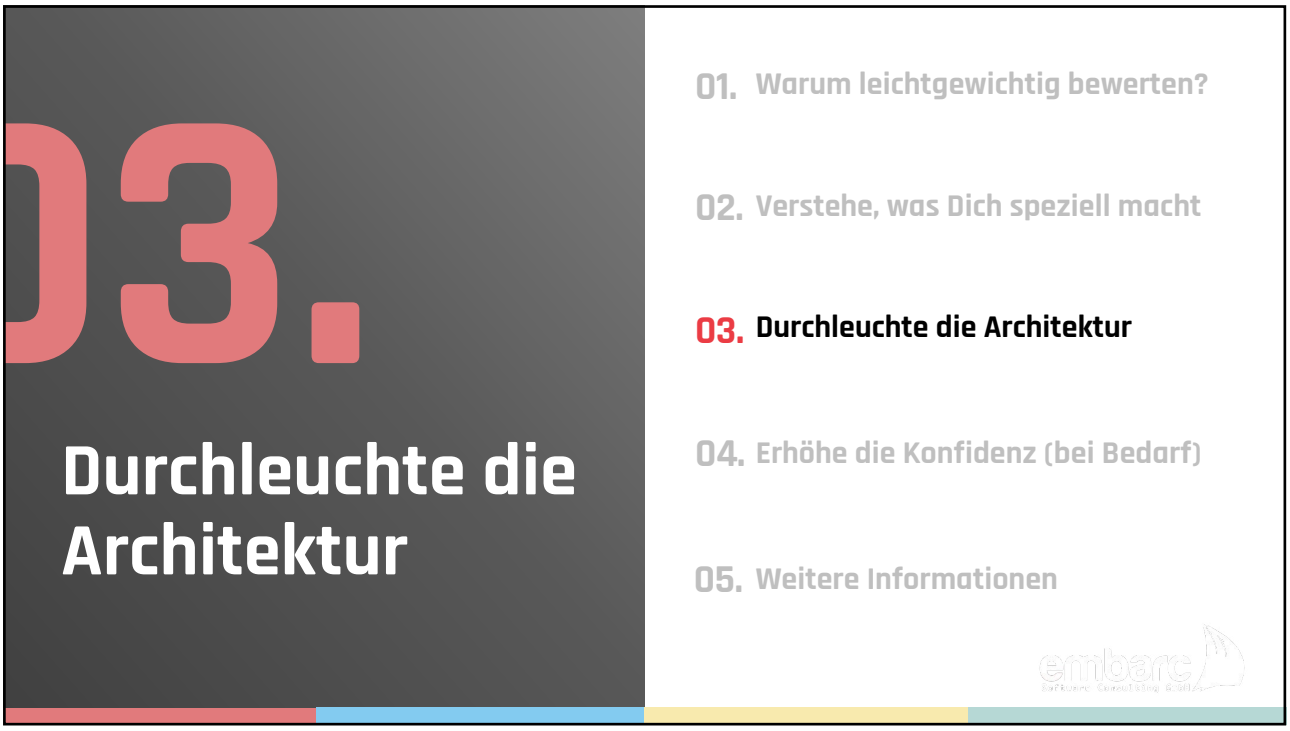
Die Zielwerte spannen darauf eine grüne Linie auf.

Beispiel

43

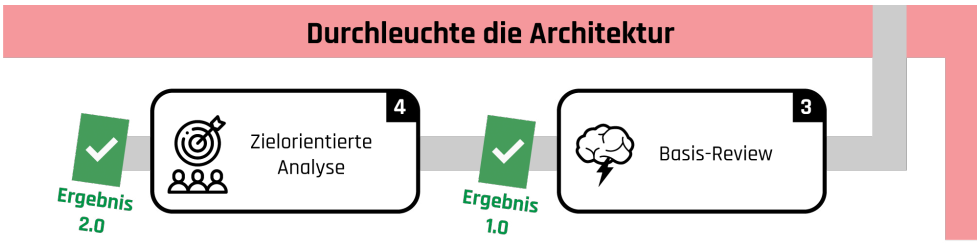

embarc.de
Softwarearchitektur in Eigenregie bewerten
44
Mit Schritt 2 steht der Ergebnisrahmen.
Verstehe was Dich speziell macht
1
Schlankes Mission Statement
2
Bewertungs-maßstab
Durchleuchte die Architektur
4
Zielorientierte Analyse
3
Basis-Review
Ergebnis 2.0
Ergebnis 1.0

44


03.
Durchleuchte die Architektur
01. Warum leichtgewichtig bewerten?
02. Verstehe, was Dich speziell macht
03. Durchleuchte die Architektur
04. Erhöhe die Konfidenz (bei Bedarf)
05. Weitere Informationen

45

Durchleuchte die Architektur



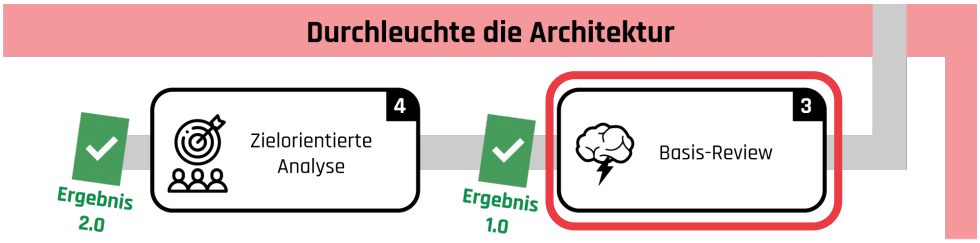
Was ist zu tun?

- Die größten Risiken des Systems identifizieren
- Die aus den Risiken resultierende Abweichung zu den Zielen quantifizieren („Lücken“)
- Zielorientierte Analysen durchführen, um das Review-Ergebnis bei Bedarf zu schärfen

46

embarc.de
Softwarearchitektur in Eigenregie bewerten

Durchleuchte die Architektur



Was ist zu tun?

- **Die größten Risiken des Systems identifizieren**
- Die aus den Risiken resultierende Abweichung zu den Zielen quantifizieren („Lücken“)
- Zielorientierte Analysen durchführen, um das Review-Ergebnis bei Bedarf zu schärfen

47

(3) Basis-Review

Produziert ein erstes Ergebnis in Form konkreter **Risiken**, die der Zielerreichung im Weg stehen.

Methodischer Ansatz

Brainstorming, angelehnt an **Pre-Mortem**

Ergebnisse des Schrittes

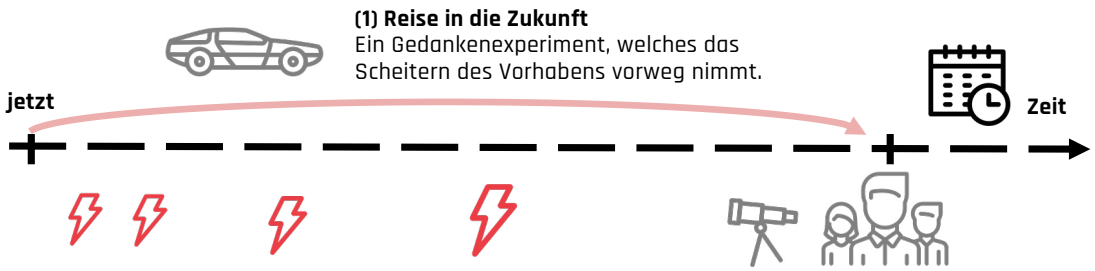
"Abweichung" vom Ziel, Ist-Einschätzung

48

Grundidee: Pre Mortem

49

Was ist ein Pre-Mortem?



(1) Reise in die Zukunft
Ein Gedankenexperiment, welches das Scheitern des Vorhabens vorweg nimmt.

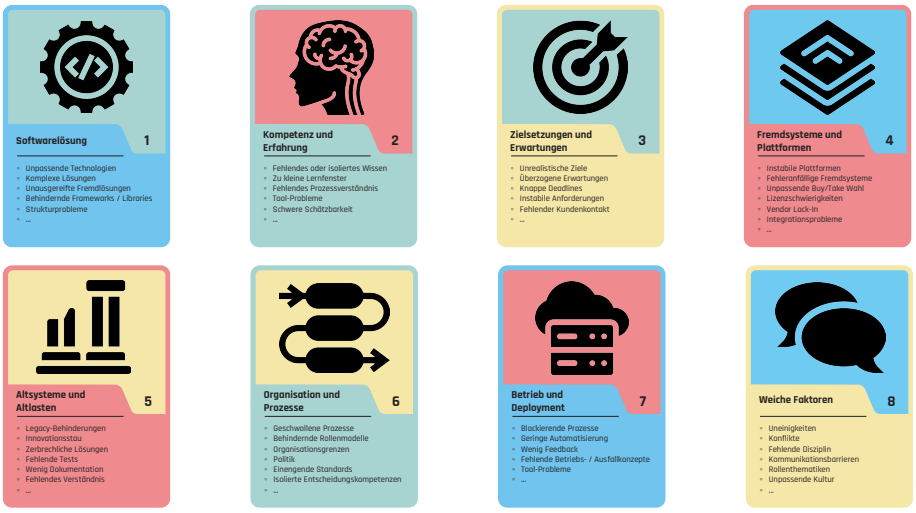
(2) Blick zurück
Das Team blickt gemeinsam zurück und überlegt, warum das Vorhaben gescheitert ist.

(3) Gründe des Scheiterns/Risiken
Probleme unterschiedlichster Art und Größe machen das Scheitern „plausibel".

50

51

Die 8 Risiko-Kategorien von LASR



52

Typische Risikothemen als Ideengeber

Brainstorming-Unterstützung: Das LASR-Kartenset hält insgesamt 32 konkrete Risikokarten als Ideengeber parat, 4 in jeder der 8 Kategorien.

53

Disclaimer

- Die folgenden **Beispiel-Risiken** sind **realistisch** für Dinge, die Team-Mitglieder im Rahmen eines Risiko-Workshops im Pre-Mortem aufwerfen.
- Bezogen auf unser Beispiel **Threema** hier sind sie **rein fiktiv**. Sie dienen der Illustration.

54

Beispielkarte

55

Hypothetisches Risiko (1)

Aufhebung des Gruppen-Limits

„Wir haben die Begrenzung in Gruppen-Größen aufgehoben. Unsere User sind begeistert von den neuen Möglichkeiten. Später bricht der Chat-Server im Threema-Backend unter der massiven Last und der Menge zu speichernden Nachrichten zusammen.“

56

Hypothetisches Risiko (2)

Das Electron-Framework fällt weg

„Wir waren gezwungen das Electron-Framework in unserer Desktop-App durch eine Alternative / etwas Selbstgebautes zu ersetzen. Das zieht sich. Zwischenzeitlich haben wir den Download des Desktop-Clients von der Seite genommen, weil er auf aktuellen OS nicht mehr funktioniert. “

57

Hypothetisches Risiko (3)

Private Kommunikation scannen

„Die europäische Gesetzgebung hat Messenger-Anbieter verpflichtet, sämtliche private Kommunikation und Dateien zu durchleuchten. Threema muss daraufhin die Lösung für private Nutzer komplett neu denken. “

58

59

Risiken einordnen

?

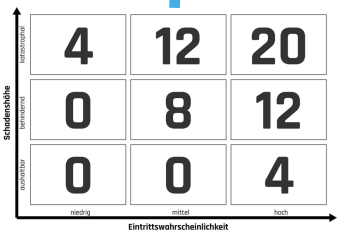
Schadenshöhe	niedrig	mittel	hoch
katastrophal	4	12	20
behindernd	0	8	12
aushaltbar	0	0	4

Eintrittswahrscheinlichkeit

60

Den betroffenen Zielen zuordnen

zoom in



Risikobereich mittel/hoch

mittlere Wahrscheinlichkeit dass das Risiko zum Problem wird

hoher Schaden für zumindest ein Qualitätsziel (katastrophal)

Anordnung im Spielbereich:

Qualitätsziel 1: Risikopunktezahl: (Wert pro Risiko) **12**

Qualitätsziel 2: Risikopunktezahl: (Wert pro Risiko) **12**

Qualitätsziel 3: Risikopunktezahl: (Wert pro Risiko) **12**

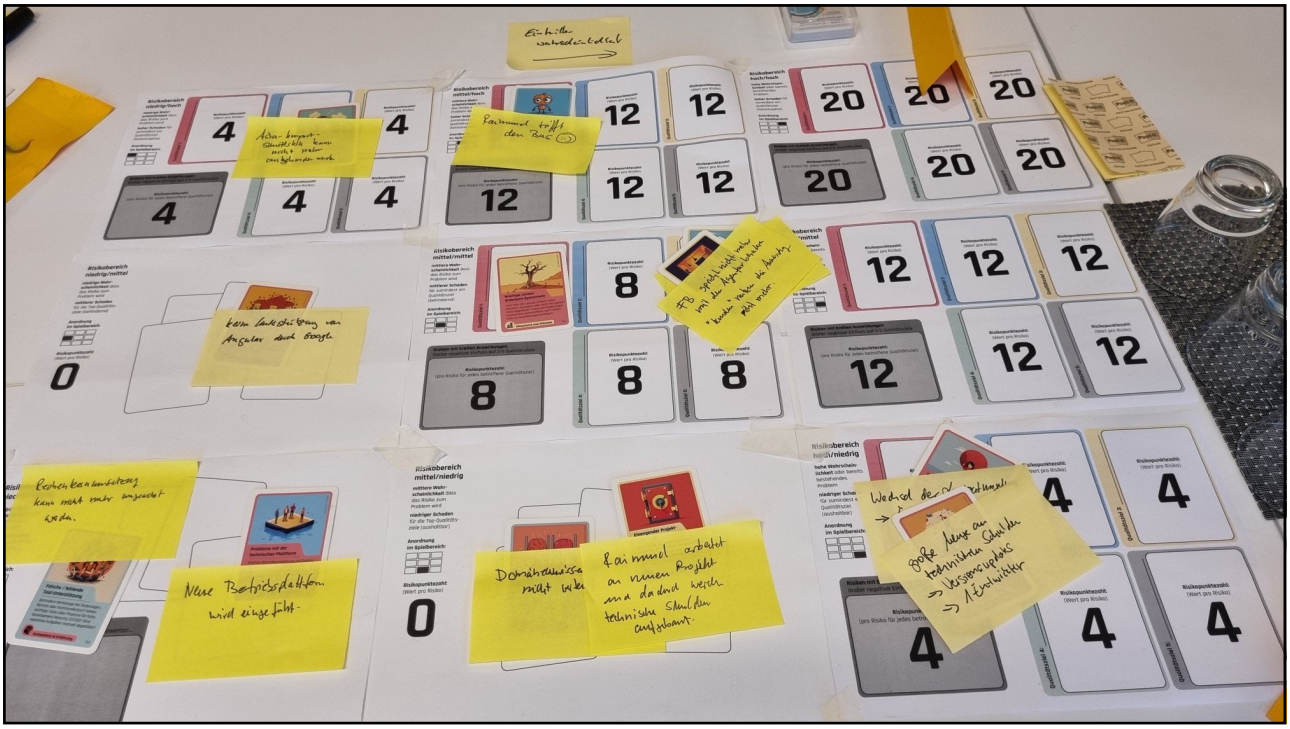
Risiken mit breiten Auswirkungen
Großer negativer Einfluss auf 2-5 Qualitätsziele

Risikopunktezahl:
(pro Risiko für jedes betroffene Qualitätsziel)
12

Qualitätsziel 4: Risikopunktezahl: (Wert pro Risiko) **12**

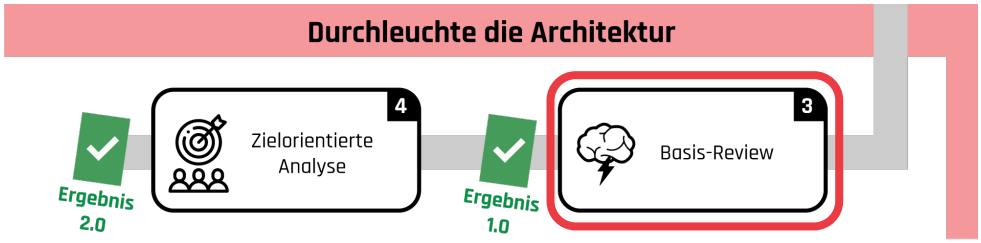
Qualitätsziel 5: Risikopunktezahl: (Wert pro Risiko) **12**

61



62

Durchleuchte die Architektur



Was ist zu tun?

- Die größten Risiken des Systems identifizieren
- **Die aus den Risiken resultierende Abweichung zu den Zielen quantifizieren („Lücken“)**
- Zielorientierte Analysen durchführen, um das Review-Ergebnis bei Bedarf zu schärfen

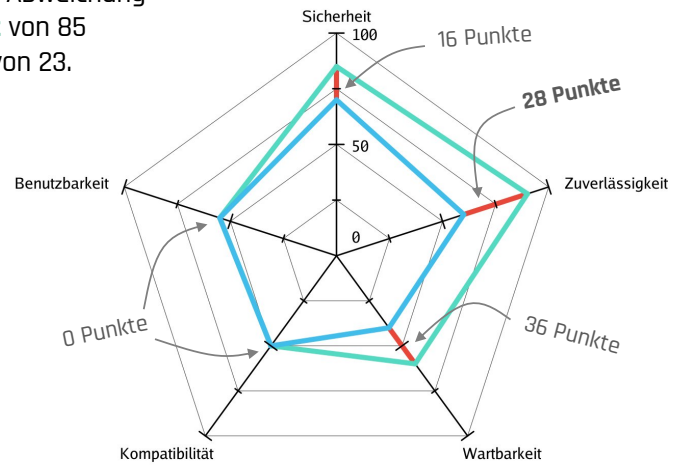
63

Von Risiken zu „Lücken“

28% Abweichung
vom **Ziellevel**
auf der **Qualitätszielachse**

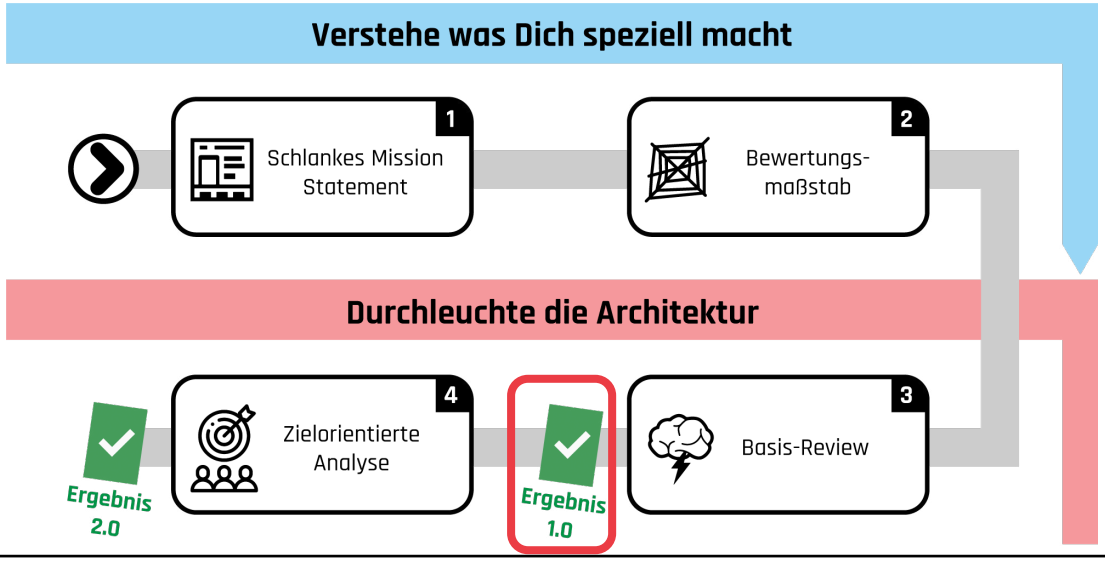
Abweichungen einzeichnen

Beispiel
28 Prozent-Punkte Abweichung
bei einem **Zielwert** von 85
ergibt eine **Lücke** von 23.



- Ziellinie
- Abweichungen (Lücken)
- Ergebnislinie

Basis-Review, 1. Ergebnis nach Schritt 3



66

LASR an einem Vormittag

Review-Workshop System YX -- Agenda

Die Zeiten dienen Euch zur groben Orientierung.
Wir machen aber pünktlich Schluss.

09:00 - 10:00	10:15 - 11:30	11:45 - 12:15
Begrüßung, Einstieg LASR — ca. 10 Minuten	3 Basis-Review ("Pre-Mortem") — ca. 60 Minuten	3 Ergebnisse diskutieren — ca. 10 Minuten
1 Abgrenzung, Mission Statement ("Claims") — ca. 15-20 Minuten	3 Zielerreichung betimmen — ca. 15 Minuten	3 Ziel-orientierte Analyse planen, TODOs — ca. 10-20 Minuten
2 Bewertungsmaßstab festlegen — ca. 20-30 Minuten	kurze Pause	Ende des Workshops
kurze Pause (+ Basis-Review vorbereiten)		

(c) embarc GmbH

Beispiel-Agenda

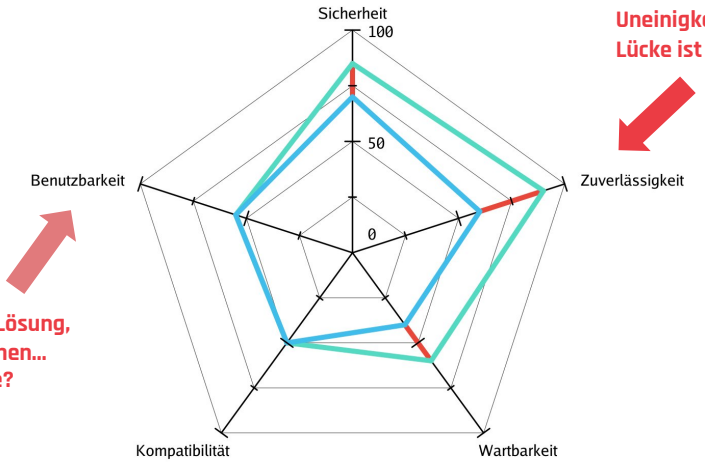
Quelle: S. Toth, S. Zörner: „Leichtgewichtige Software-Reviews mit LASR ", Informatik Aktuell 2024

67

04. Erhöhe die Konfidenz (bei Bedarf)

68

Fokus: Zielorientierte Analyse (Beispiele)

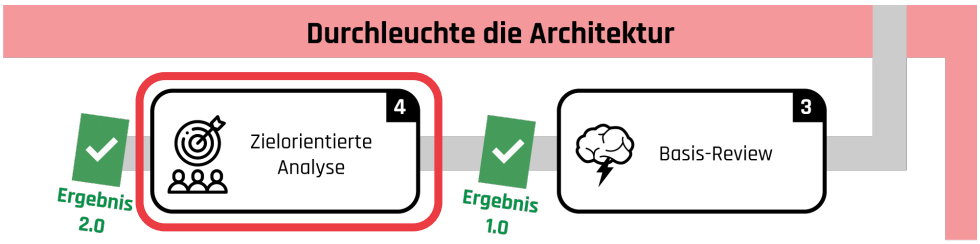


Uneinigkeit!
Lücke ist strittig

Keine Stärken in der Lösung,
Kein explizites Vorgehen...
Trotzdem keine Lücke?

69

Durchleuchte die Architektur



Was ist zu tun?

- Die größten Risiken des Systems identifizieren
- Die aus den Risiken resultierende Abweichung zu den Zielen quantifizieren („Lücken“)
- **Zielorientierte Analysen durchführen, um das Review-Ergebnis bei Bedarf zu schärfen**

70

(4) Zielorientierte Analyse

Untersucht **gezielt** Stärken und Schwächen im System und **schärft** so **das Ergebnis**.

Methodischer Ansatz

Qualitative Durchsprache "strittiger" Ziel-Achsen

Ergebnisse des Schrittes

verbesserte Ist-Einschätzung, Qualitätsaussagen

71

embarc

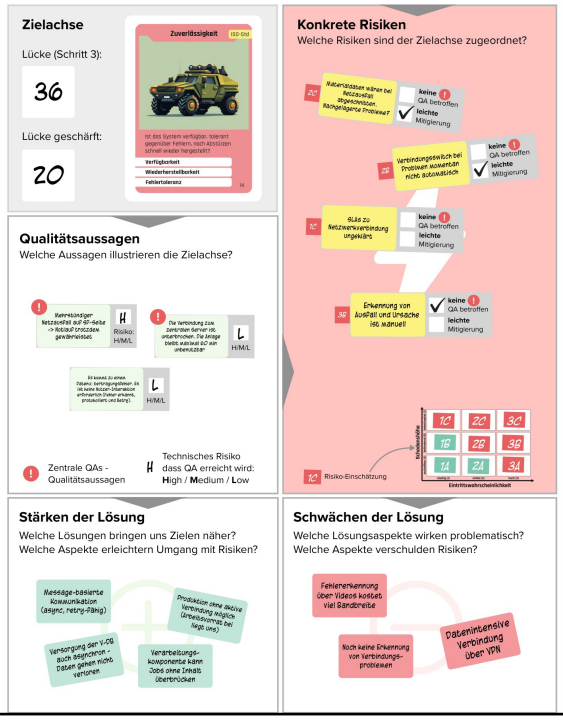
embarc.de

Softwarearchitektur in Eigenregie bewerten

Template

Vorlage, um die Ergebnisse der Analyse zu sammeln, zu verdichten und TODOs abzuleiten.

Beispiel



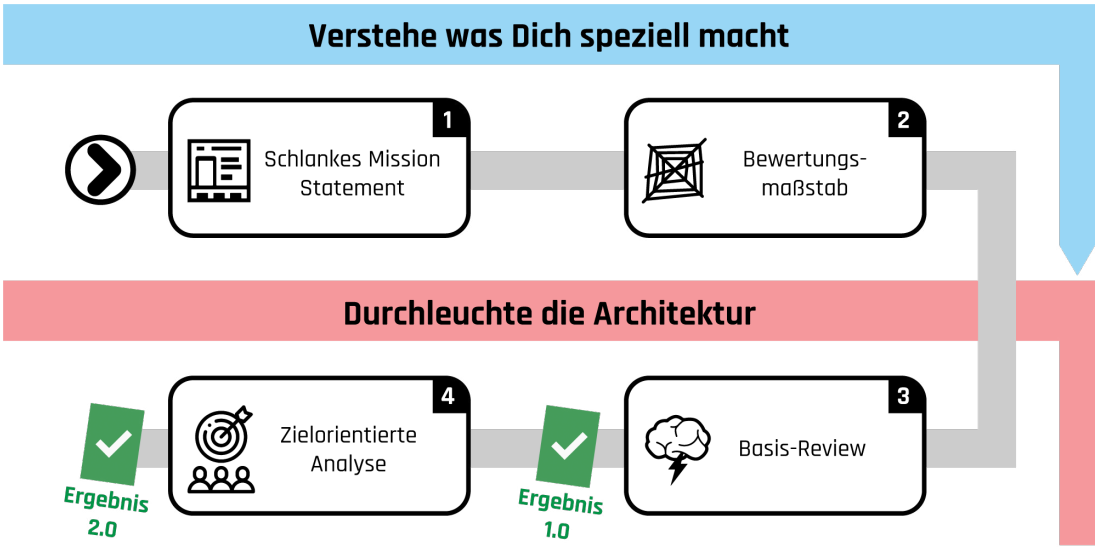
72

embarc

embarc.de

Softwarearchitektur in Eigenregie bewerten

Tätigkeiten und Schritte im Kern-Review



73

Typische Unsicherheiten im Anschluss

Hängt da noch mehr dran?
Gefundene **Risiken** sind in ihrer Natur oder Auswirkung **zu wenig verstanden**, um direkt hilfreiche Aktivitäten daraus abzuleiten.

In der Theorie OK, aber was ist mit …?
Rahmenbedingungen (z.B. Standards) oder die eigene **Organisation** wurde als Risikofaktor **zu wenig betrachtet**.

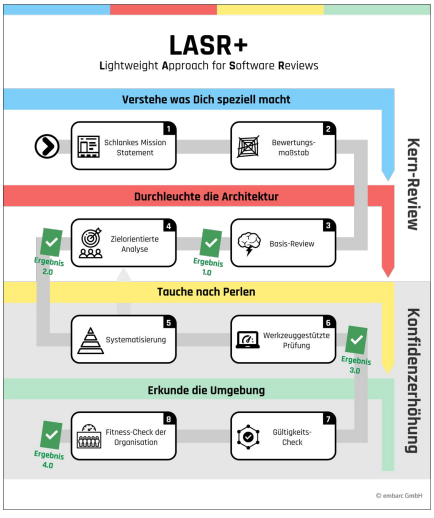
Wo ist der Code?
Die **Architekturebene** wird als **zu abstrakt** für die tatsächlichen Probleme des Vorhabens wahrgenommen.

Wo fangen wir an?
Die **Priorisierung** der identifizierten Probleme ist **schwierig**, ggf. fehlt es bei kleinteiligen Ergebnissen an Überblick oder Einordnung.

Ist nicht auch XY wichtig?
Der **Fokus** auf max. 5 Qualitätsziele wirkt **problematisch.**

74

Ausblick: Nach dem LASR-Review …



LASR bietet einen erweiterten, optionalen Modus (LASR+) zur Konfidenzerhöhung.

I. LASR-Review

Liefert rasch ein erstes Review-Ergebnis, das im vierten Schritt bereits geschärft wird.

II. LASR+ (optional)

Bietet bei Unsicherheiten mit dem Ergebnis mehr Tiefe und schließt Code-Ebene und Organisationsseite mit ein.

75

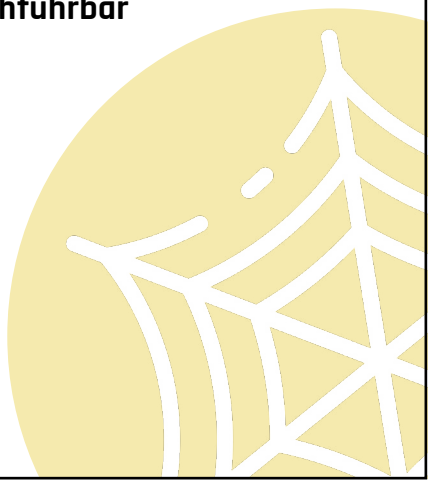
05.

Weitere Informationen

76

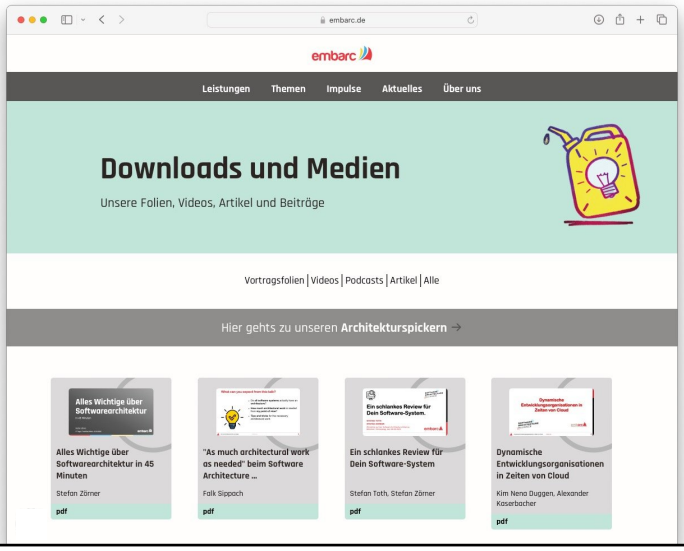
Markante Merkmale von LASR

- Schlankere Methode als ATAM, trotzdem **zielorientiert**
- Mit dem **eigenen Team** und potentiell **alleine durchführbar**
- Liefert **schnell** ein **erstes Ergebnis**, z.B. an einem Nachmittag
- **Spinnennetzgraphik** zur Erkenntnisverdichtung und -kommunikation
- optionale Aktivitäten zur schrittweisen **Konfidenzerhöhung bei Bedarf**
- **Unterstützungsmaterial** für Bewertungs-maßstab, Risikofindung und Ergebniserarbeitung



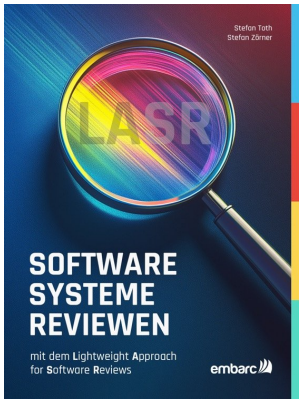
77

Folien als PDF zum Download



➔ embarc.de/download/

Buchtipp zum Thema



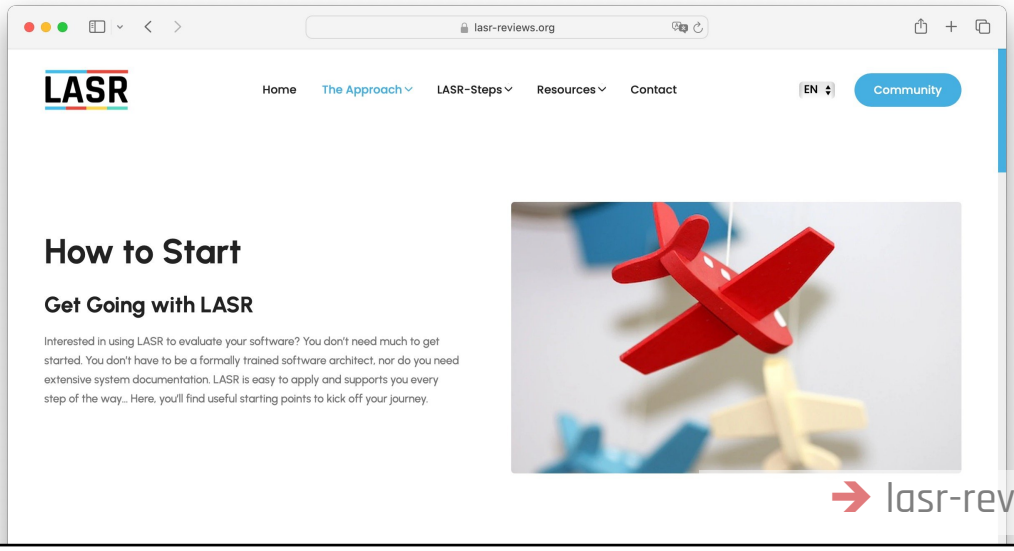
Software-Systeme reviewen
mit dem Lightweight Approach
for Software Reviews - LASR

Leanpub

Autoren: Stefan Toth, Stefan Zörner
Verlag: Leanpub, September 2023
Sprache: Deutsch, EPUB, PDF

➔ leanpub.com/software-systeme-reviewen/

„Offizielle“ LASR-Webseite (EN + DE)



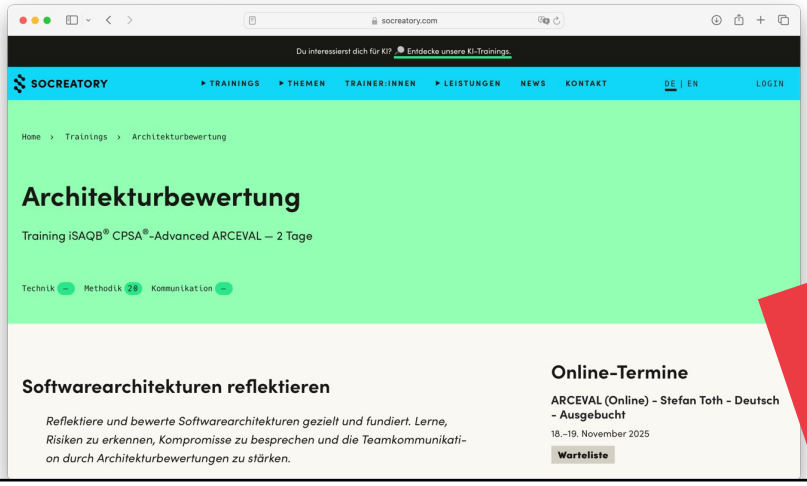
➔ lasr-reviews.org

80

Seminar: Architekturbewertung

➔ socreatory.com/de/trainings/arceval

SOCREATORY

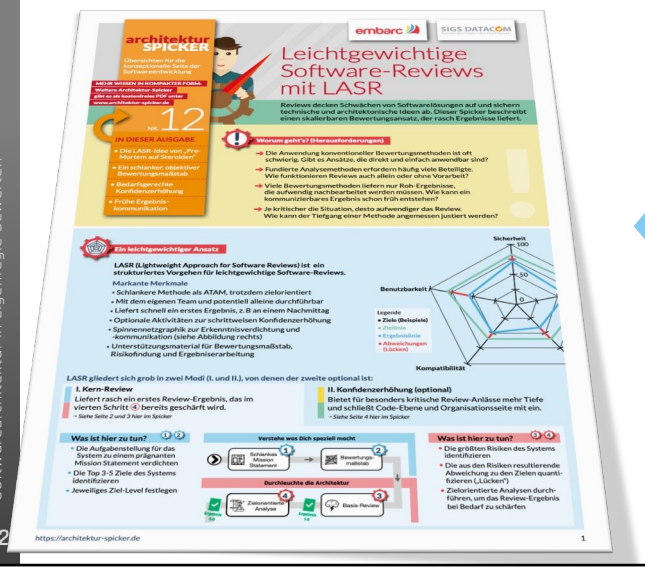


iSAQB® ACCREDITED ARCEVAL Training

Nächster Termin in München (Therme Erding) **09.-10.12.2025** mit Stefan Toth

81

Architektur-Spicker zum Thema …



„Mit unseren Architektur-Spickern beleuchten wir die konzeptionelle Seite der Softwareentwicklung."

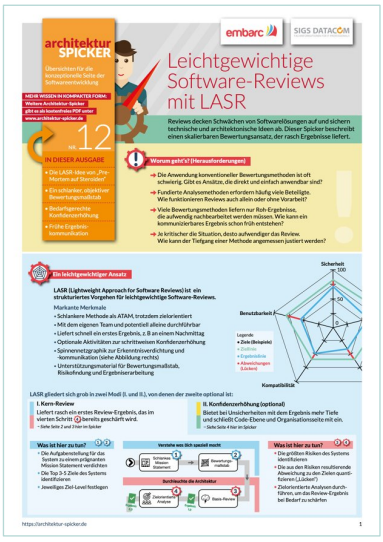
Architektur-Spicker #12
Leichtgewichtige Software Reviews mit LASR

PDF, 4 Seiten
Kostenloser Download.

→ architektur-spicker.de

82

Zum Mitnehmen



Den LASR-Spicker könnt ihr euch direkt vorne bei mir abholen (solange der Vorrat reicht).

83


Architektur-
Punsch
Unser Line-Up
embarc.de/architektur-punsch-2025/
Kim Duggen
Oliver Zeigermann
Arnold Franke
Sven Woltmann
Felix Kammerlander
Falk Sippach
Alex Thurow
Stefan Zörner
Stefan Toth
Cosima Laube
mehr Infos...
Workshops, neue Perspektiven & Plaudern in lockerer
Punsch-Atmosphäre am 08. Dezember 2025!

84


embarc.de
Softwarearchitektur in Eigenregie bewerten
85
Vielen Dank.
Ich freue mich auf Eure Fragen!
Stefan.Zoerner@embarc.de
linkedin.com/in/stefan-zoerner
@StefanZoerner@mastodon.social
embarc

85