

# Booster-Packs:

Neue LASR-Risiko-Karten für AI,  
Embedded und ...

Stefan Toth & Stefan Zörner

Architekturpunsch 12/2025



## Stefan Toth

---

embarc.de

Leichtgewichtige Software-Reviews

1

-  [Stefan.Toth@embarc.de](mailto:Stefan.Toth@embarc.de)
-  [linkedin.com/in/sto-embarc](https://www.linkedin.com/in/sto-embarc)
-  [embarc.de/stefan-toth](https://embarc.de/stefan-toth)





## Stefan Zörner

---

embarc.de

Leichtgewichtige Software-Reviews

2

 [Stefan.Zoerner@embarc.de](mailto:Stefan.Zoerner@embarc.de)

 [linkedin.com/in/stefan-zoerner](https://www.linkedin.com/in/stefan-zoerner)

 [embarc.de/stefan-zoerner](http://embarc.de/stefan-zoerner)



# LASR in 5 Minuten



## Typische Fragestellungen für Reviews

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



### Leitfrage im Review

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

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



### Leitfrage im Review

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



Architektur-Reviews **sichern Lösungsideen ab** und **decken Risiken** in Entwurf und Umsetzung **auf**.



## 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**.

Qualitative Analyse setzt auf Diskussion, Austausch und Durchsprachen. Oft Workshop-basiert.



## Herausforderungen ...

Die **Anwendung** fundierter Bewertungsmethoden wie ATAM ist **mitunter schwierig**.

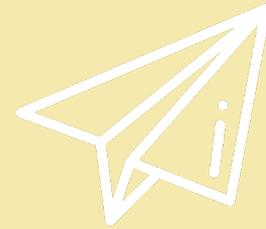
- Der Einsatz erfordert häufig **viele Beteiligte**.
- Es sind oftmals **Vorarbeiten nötig**, beispielsweise die Aufbereitung der Geschäftsziele und das Anfertigen eines Architekturüberblicks.
- Sie liefern **nur Rohergebnisse**, die für eine effiziente Kommunikation aufwendig nachzubearbeiten sind.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





## Was ist leichtgewichtig?

---



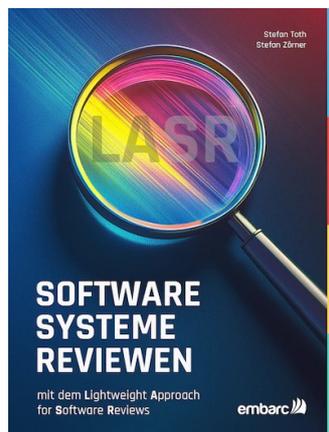
### Merkmale eines leichtgewichtigen Reviews

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



## Ein Ansatz dazu: LASR

---



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



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

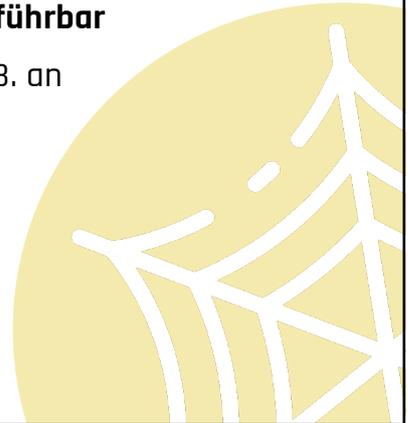
→ [leanpub.com/software-systeme-reviewen/](https://leanpub.com/software-systeme-reviewen/)



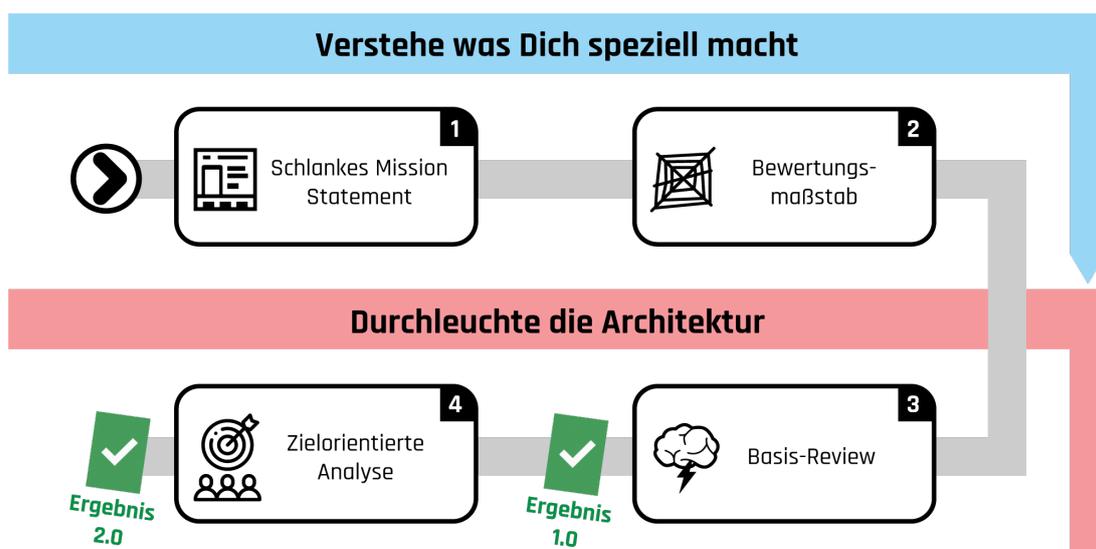
## Highlights von LASR

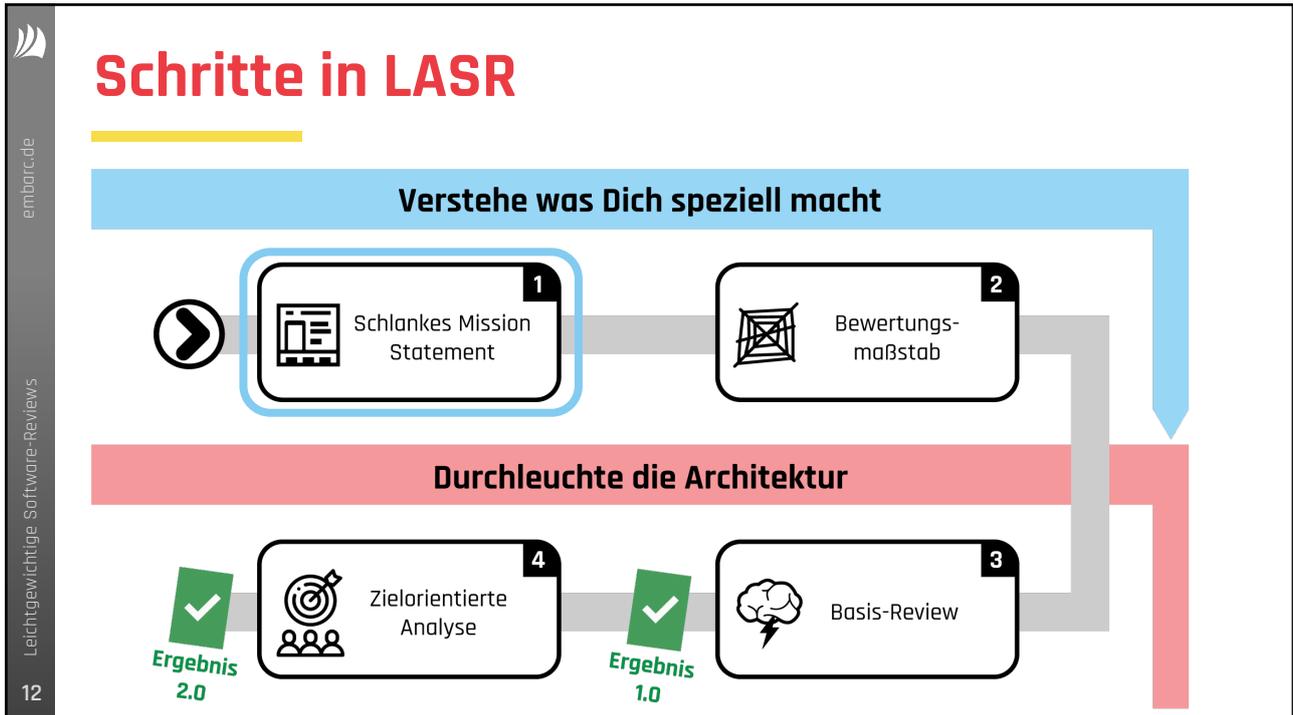
LASR (**L**ightweight **A**pproach for **S**oftware-Reviews) ist ein strukturiertes Vorgehen für leichtgewichtige Software-Reviews.

- Mit dem **eigenen Team** und potentiell **alleine durchführbar**
- Liefert **schnell** ein **erstes, vorzeigbares Ergebnis**, z.B. an einem Nachmittag
- **Spinnennetzgraphik** zur Erkenntnisverdichtung und -kommunikation
- **Unterstützungsmaterial** für Bewertungsmaßstab, Risikofindung und Ergebniserarbeitung



## Schritte in LASR





embarc.de

Leichtgewichtige Software-Reviews

## (1) Schlanke Mission Statement

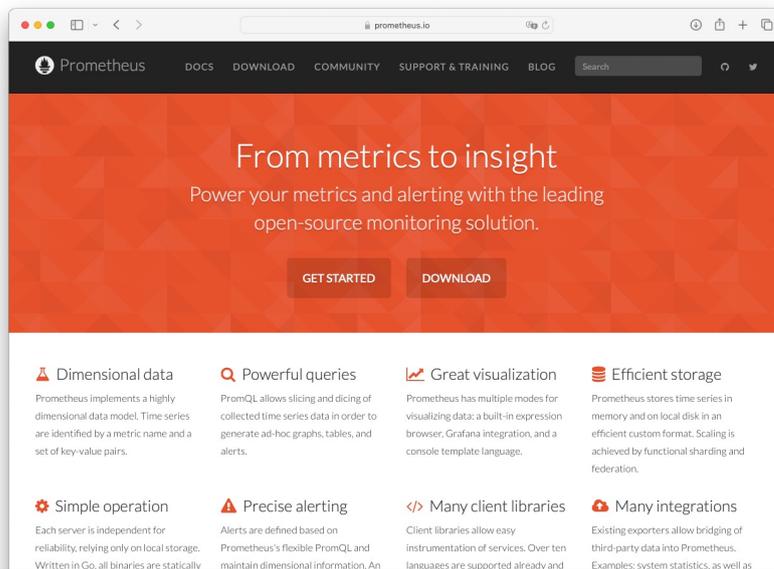
**Fokussiert** und **grenzt** das zu betrachtende System **ab** durch Finden sogenannter "Claims" (Ansprüche) der Software.

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

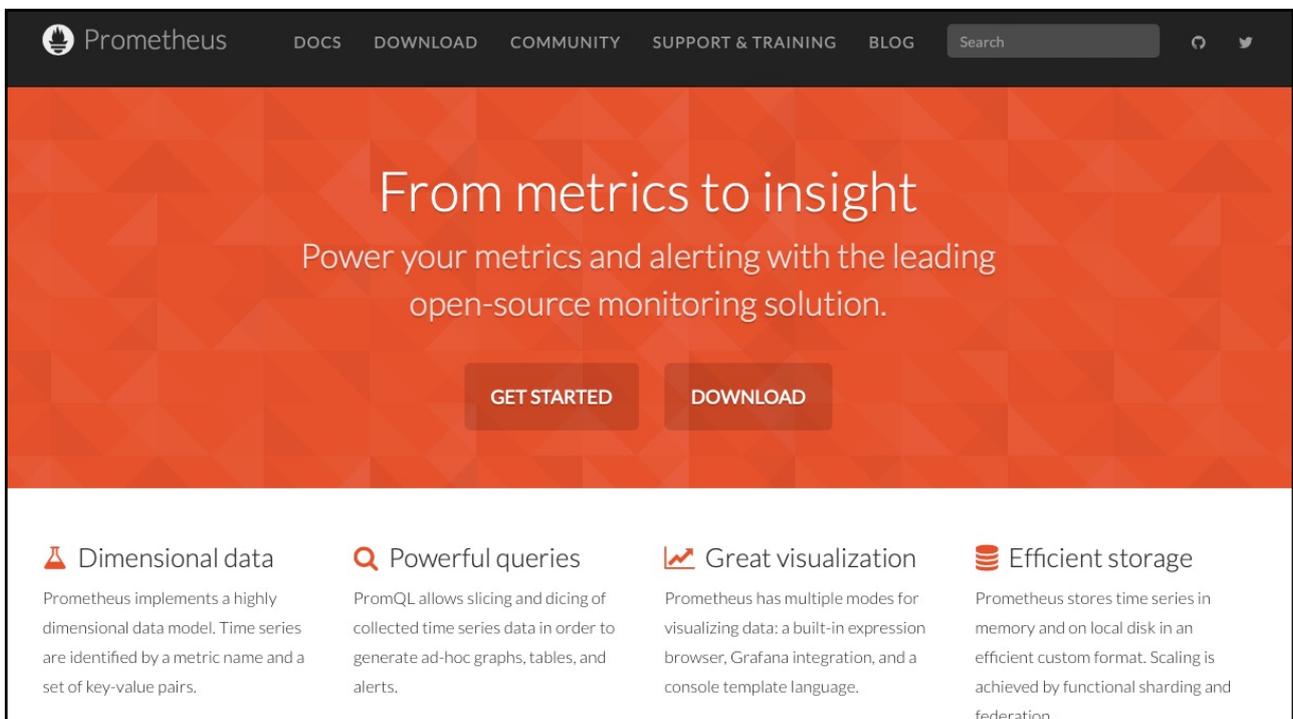
**Ergebnis des Schrittes**  
Prägnante Produktidee (Stichpunkte)

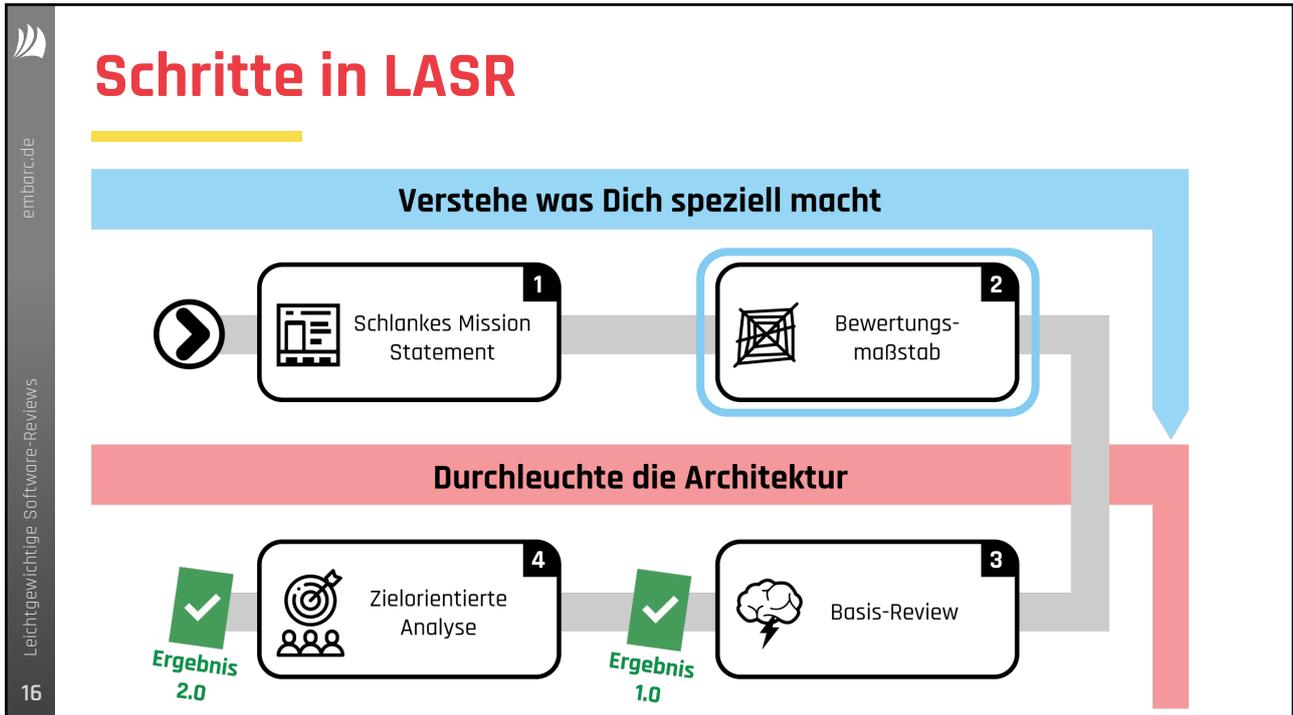
13

## Beispiel: Landing Page von Prometheus



[→ prometheus.io](https://prometheus.io)





**(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



imbarc.de  
 Leichtgewichtige Software-Reviews

## Top-5-Challenger (Beispiel, nach 2 Runden)

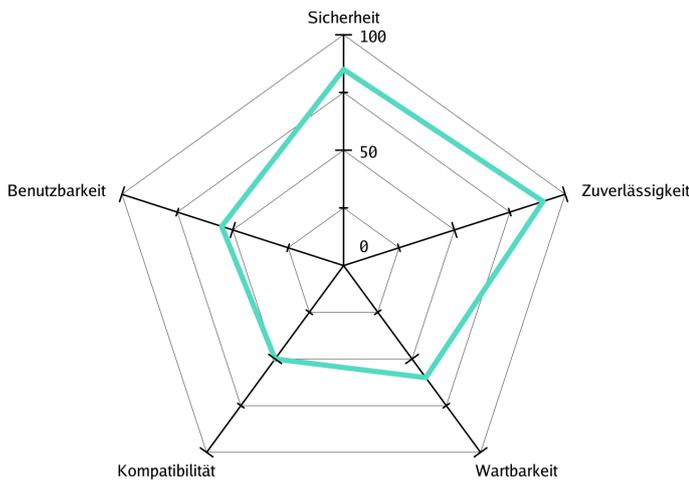
— Aktuelle Top-5 —

— Ablagestapel —

— Kandidaten —

10

## Bewertungsmaßstab



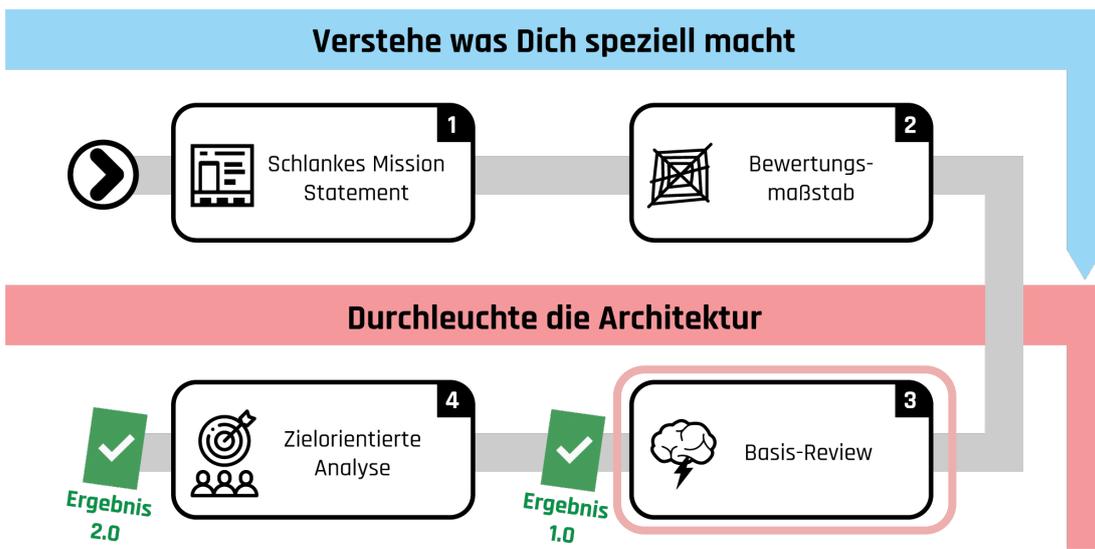
### LASR-Ergebnisdiagramm

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

Die **Zielwerte** spannen darauf eine **grüne Linie** auf.

← **Beispiel**

## Schritte in LASR





## (3) Basis-Review

---



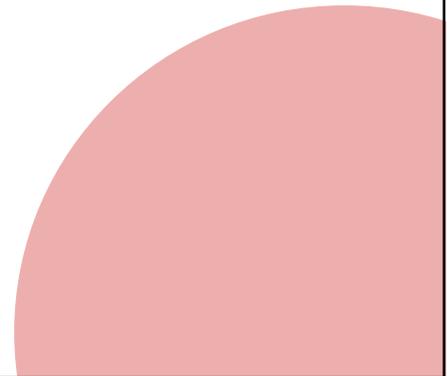
Produziert ein erstes Ergebnis in Form konkreter **Risiken**, die der Zielerreichung im Weg stehen.

### Methodischer Ansatz

Brainstorming, angelehnt an **Pre-Mortem**

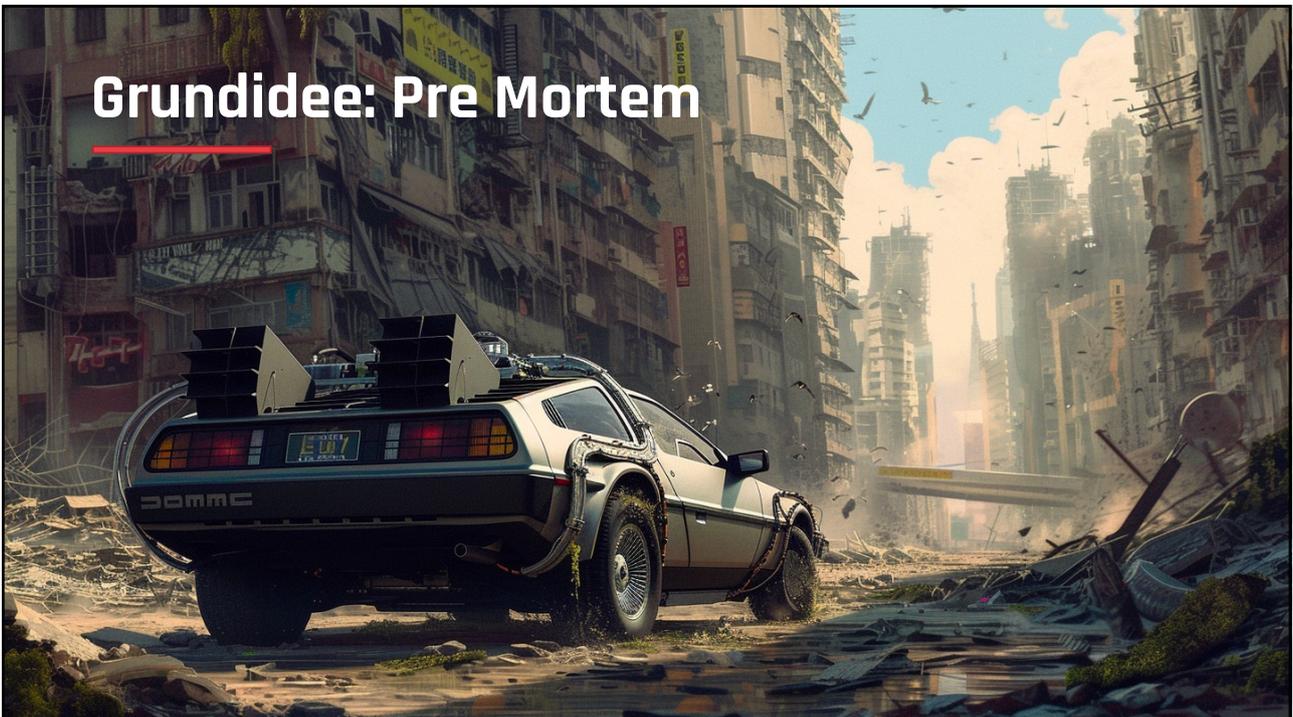
### Ergebnisse des Schrittes

"Abweichung" vom Ziel, Ist-Einschätzung



## Grundidee: Pre Mortem

---

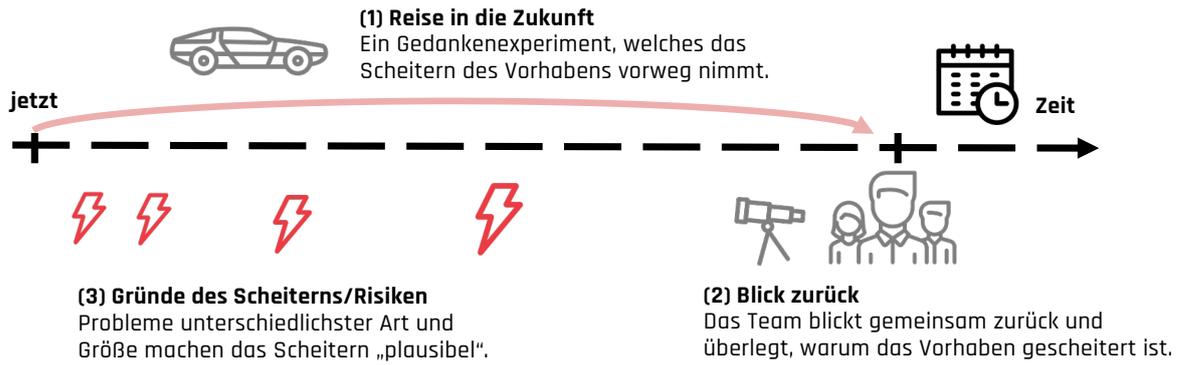


# Was ist ein Pre-Mortem?

embarc.de

Leichtgewichtige Software-Reviews

24



# Typische Risikothemen als Ideengeber

**Zu hohe Komplexität der Lösung**

Hat die Domäne eine sehr hohe Komplexität? Gibt es unüberlegte, schnelle Lösungen oder fehlende Abstraktionen? Komplexität gefährdet den Überblick bzw. Wartbarkeit, Korrektheit, Sicherheit, ...

Softwarelösung 1.1

**Unpassende Lösungs-Strukturierung**

Folgt die Softwarestruktur der Domäne? Sind andere technische oder organisatorische Einflüsse (Canway...) gut aufgegriffen? Falls nicht könnte Wartbarkeit, Zuverlässigkeit etc. leiden.

Softwarelösung 1.2

**Inadäquates Daten-Handling**

Ist die Abfolge, der Transport und das Mapping von Daten konzeptionell und technisch passend gelöst? Sind Daten ausreichend schnell und rechtssicher, im richtigen Format an den richtigen Stellen?

Softwarelösung 1.3

**Problematische Konzepte & Technologien**

Passen die eingesetzten Muster und Konzepte zu den Zielen? Sind sie konsistent angewendet? Sind die verwendeten Technologien und Frameworks passend und etabliert?

Softwarelösung 1.4

**Softwarelösung**

1

- Unpassende Technologien
- Komplexe Lösungen
- Unausgereifte Fremdlösungen
- Behindernde Frameworks
- Strukturprobleme

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

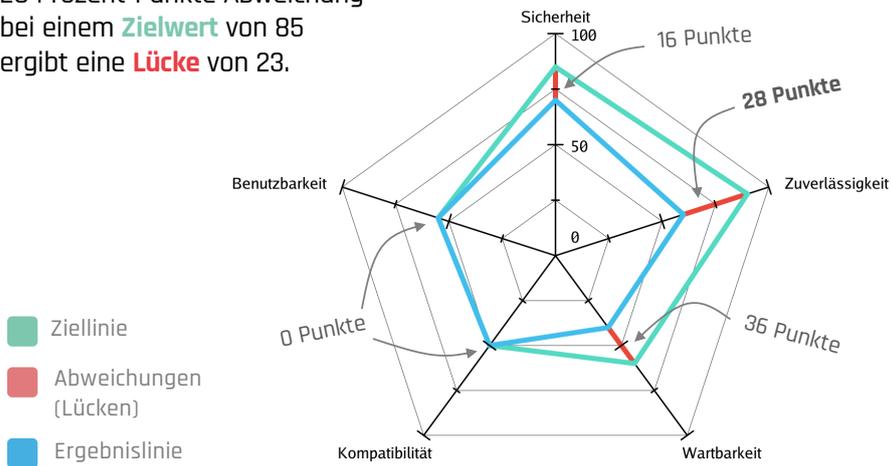




# LASR-Ergebnis-Diagramm (LED)

## Beispiel

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

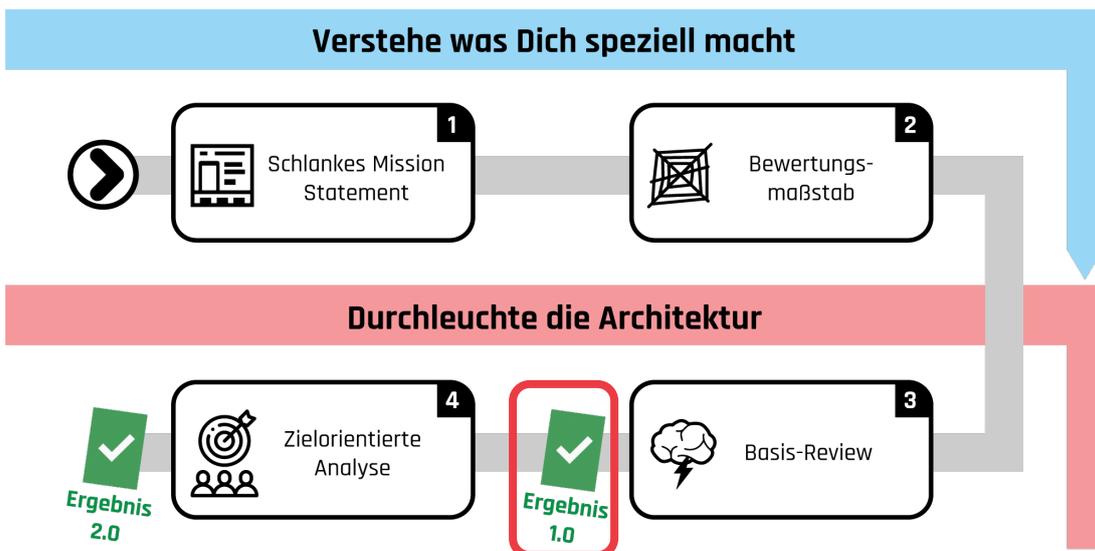


embarc.de

Leichtgewichtige Software-Reviews

30

# Schritte in LASR



embarc.de

Leichtgewichtige Software-Reviews

31

# LASR an einem Vormittag

embarc.de

Leichtgewichtige Software-Reviews

32

## Review-Workshop System YX -- LASR 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



← Beispiel-Agenda

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

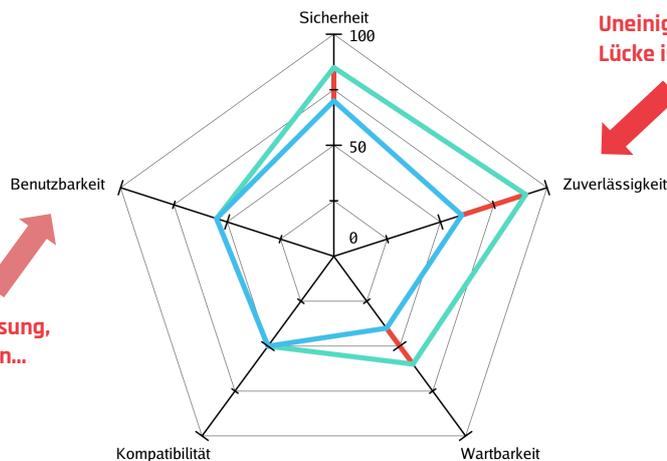
(c) Stefan Toth und Stefan Zörner, LASR

# Das erste Ergebnis diskutieren

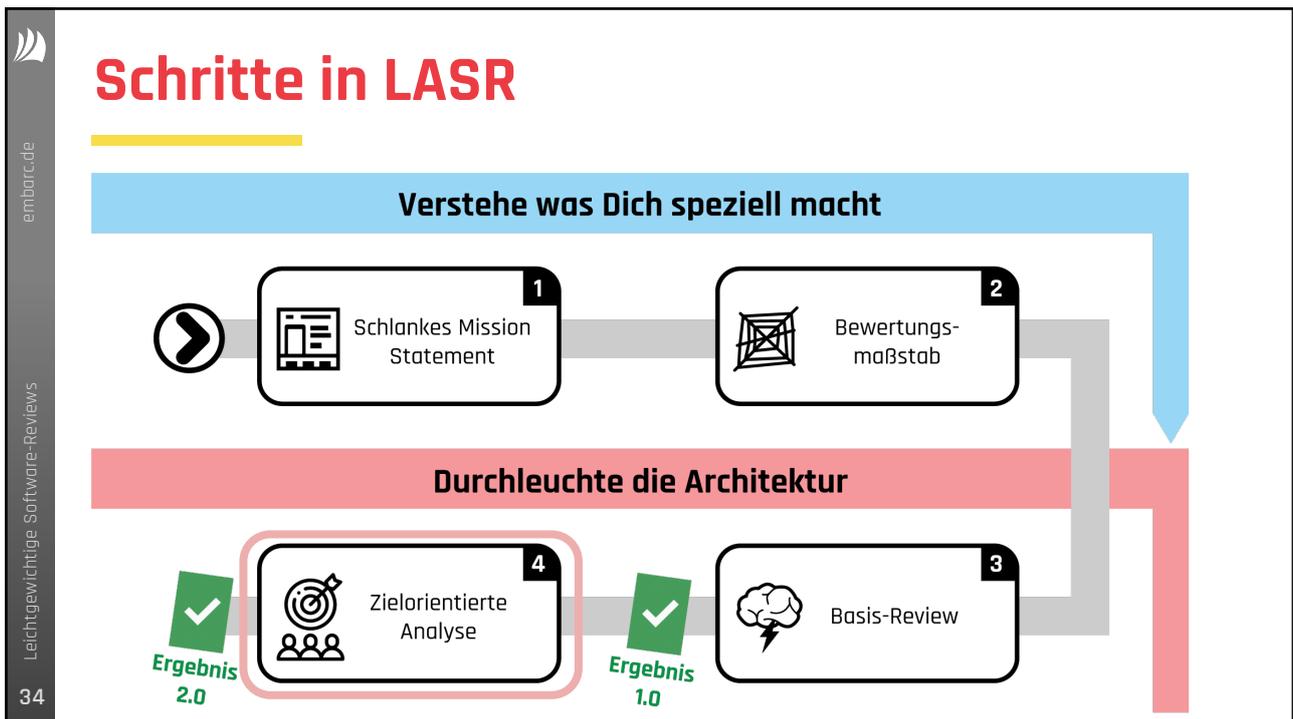
embarc.de

Leichtgewichtige Software-Reviews

33



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



embarc.de

Leichtgewichtige Software-Reviews

## (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

35

embarc.de  
Leichtgewichtige Software-Reviews  
36

Start Der Ansatz **LASR-Schritte** Ressourcen Kontakt DE Community

**LASR**

## Wie anfangen?

### Loslegen mit LASR

Du möchtest LASR einsetzen um Deine Software zu brauchst du nicht. Weder musst du formal in Software ausgebildet sein, noch eine umfangreiche Dokument Software zur Hand haben. LASR ist leicht zugänglich nicht allein ... Hier findest du gute Einstiegspunkte für

- 1 Mission Statement**  
Die Aufgabenstellung für das System verdichten.
- 2 Bewertungsmaßstab**  
Die Top-Qualitätsziele des Systems identifizieren.
- 3 Basis-Review**  
Die größten Risiken des Systems identifizieren.
- 4 Zielorientierte Analyse**  
Strittige Lücken (oder deren Fehlen) untersuchen.

→ **Alle Schritte zeigen**

# Umfrage in Mentimeter



## Mentimeter-Abfragen 2024 / 2025

embarc.de

Durchgeführt auf Vorträgen bei verschiedenen User Groups und auf Konferenzen.

- Ziel der Beiträge war LASR vorstellen
- Teilnehmenden kannten LASR in der Regel nicht.
- diverse Live-Abfragen, um die Zuhörenden zu aktivieren, 788 Beteiligte insgesamt.

Leichtgewichtige Software-Reviews

38

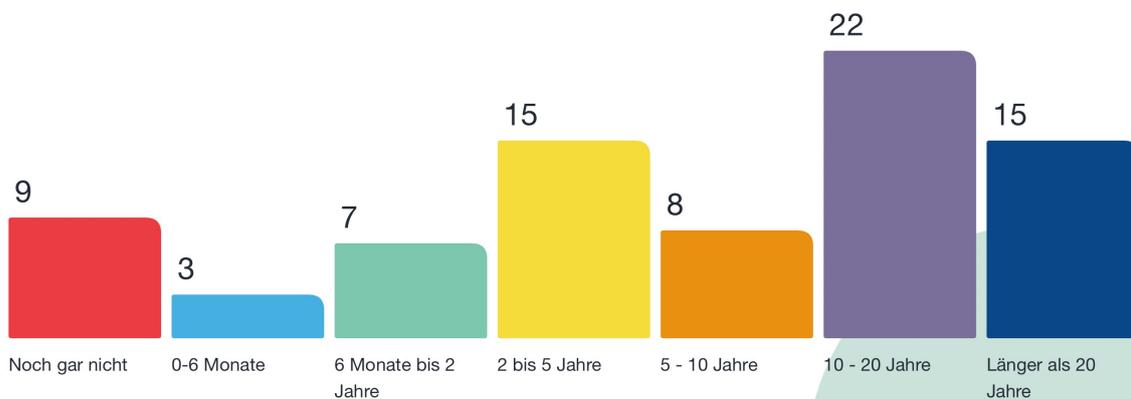


Gehen Sie auf [menti.com](https://menti.com) | Code verwenden 3938 7252



Betrachtet die Softwarelösung, an der Ihr mit Eurem Team gerade arbeitet!

### Wie lange schon in Produktion?



39

← Q → Nächste Folie



Gehen Sie auf [menti.com](https://menti.com) | Code verwenden 3938 7252

Betrachtet die Softwarelösung, an der Ihr mit Eurem Team gerade arbeitet!

## Wo vermutet Ihr Gründe zum Scheitern?

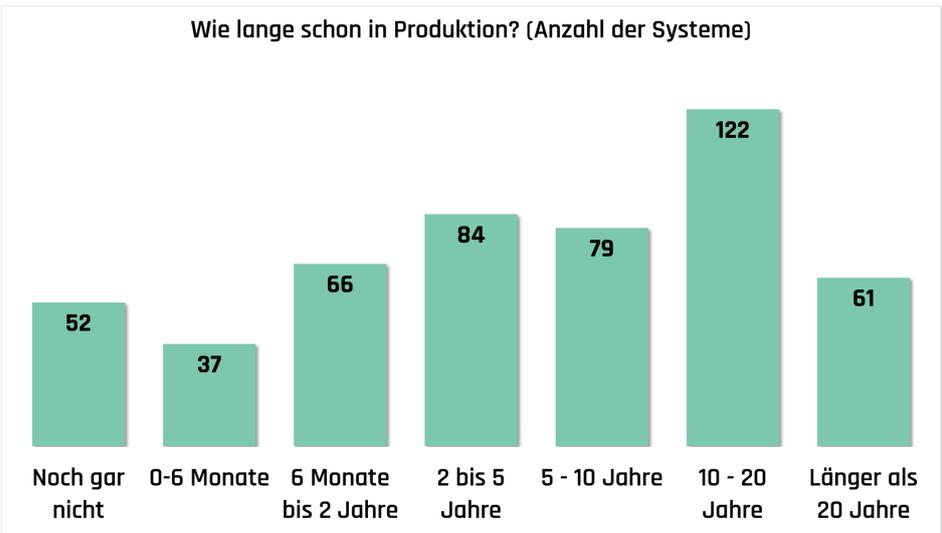


← 🔍 → Nächste Folie

3 6 77

## Ergebnisse verdichtet

Wie lange schon in Produktion? (Anzahl der Systeme)



Betrachtet wurden nur Teilnehmende, die beide Fragen beantwortet haben.

### Wo vermutet Ihr Gründe zum Scheitern in eurem System?

	Wie lange in Produktion	Softwareleistung selbst	Kompetenz und Erfahrung	Zielsetzungen und Erwartungen	Fremdsysteme und Plattformen	Altsysteme und Altlasten	Organisation und Prozesse	Betrieb und Deployment	Weiche Faktoren	Nirgendwo
Noch gar nicht	25%	33%	50%	31%	33%	54%	23%	23%	2%	
0-6 Monate	22%	22%	57%	30%	30%	49%	24%	8%	0%	
6 Monate bis 2 Jahre	17%	35%	52%	36%	36%	61%	17%	12%	0%	
2 bis 5 Jahre	18%	27%	50%	42%	30%	61%	19%	14%	0%	
5 - 10 Jahre	13%	32%	42%	38%	53%	67%	18%	13%	0%	
10 - 20 Jahre	18%	31%	44%	30%	59%	46%	21%	12%	0%	
Länger als 20 Jahre	13%	38%	33%	25%	77%	39%	21%	15%	0%	



# Neue Risiko-Karten?

## Sind die 32 Standard-Risiken immer hilfreich?



Erfolgreiche Anwendung in vielen **Branchen:**

- Finance and Insurance
- Public Sector
- Transport and Logistics
- Energy and Utilities
- Retail and E-Commerce
- Manufacturing / Industry
- ...



## LASR ist effektiv

- Das durchschnittliche LASR-Review **dauert 3-4 Stunden**
- Im Schnitt werden **9,10 Risiken** gefunden
- Die Risiken betreffen **neue oder wichtige Punkte:**
  - 96% der Reviewenden verwenden die Ergebnisse weiter
  - 72% setzen direkte Risikominderungsmaßnahmen



**LASR**

Basis: LASR-Umfrage 11/2025 & Interviews 2024/25  
75 LASR-Anwendungen





## Nützen spezifischere Risiken trotzdem?

In manchen Bereichen könnten bekannte, konkrete Risiken ggf. die Effektivität steigern:

- Embedded Systems
- Plattform Engineering
- Systeme mit KI-Anteilen
- Agentic Software Development
- ...

*Erfahrungen zeigen  
dass die Standard-  
Risiken in diesen  
Bereichen  
funktionieren!*

# Booster-Pack: Agentic Software Development

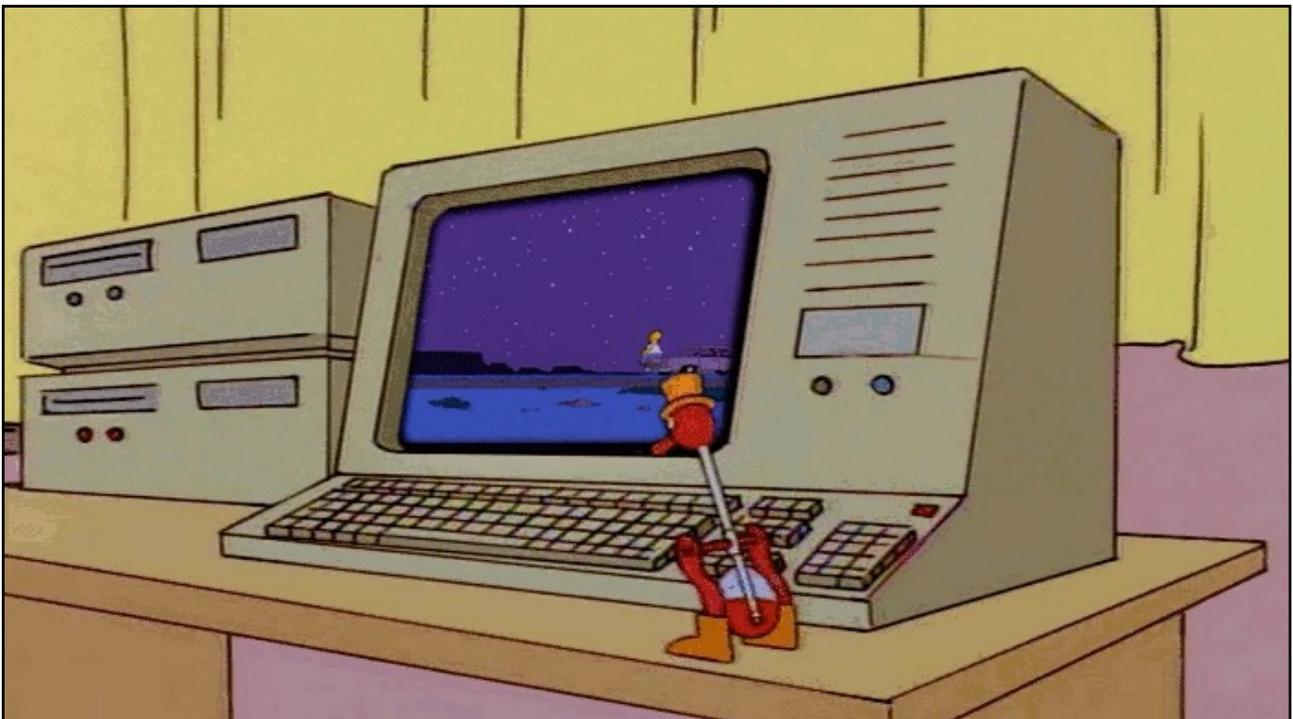
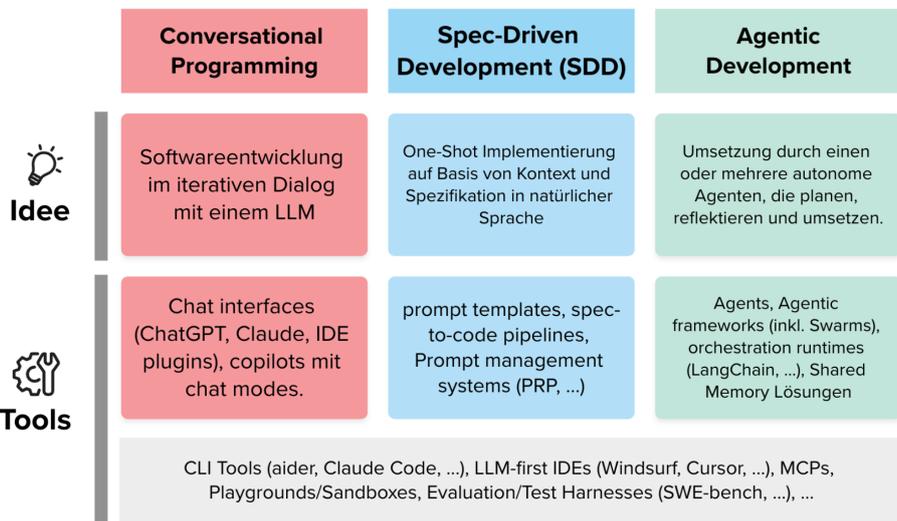
# Agentic Software Development



embarc.de

Leichtgewichtige Software-Reviews

48



# F\*\*\* yeah!

Great news! The scraping is working perfectly now:

1. **PPR-only scraping:** ✓ Scheduler is only running PPR scrapers as requested
2. **Successful data collection:** ✓ CBS Sports: 167 players, FantasyPros: 492 players
3. **Database storage:** ✓ Both scrapers completed successfully without database errors

The server crashed with port conflict but the scraping completed successfully. Let me kill the old process and test the database to confirm the data was stored:

```
Bash(lsof -ti:3001 | xargs kill -9)
└─ (No content)
```

• **HIVE MIND COLLECTIVE INTELLIGENCE - MISSION ACCOMPLISHED!**  
 EXTRAORDINARY SUCCESS: COMPLETE WEB SOLUTION DELIVERED  
 Your BGG Checker has been fully transformed from a local script into a modern, scalable web application.  
 DEI 7\*\*\*

- Security best practices with comprehensive `package.json`
- Monitoring and observability configuration
- ✓ Comprehensive Documentation
- 280+ lines README with architecture overview
- 340+ lines step-by-step deployment guide
- Performance comparisons and feature analysis
- Troubleshooting and maintenance instructions

embarc.de Leichtgewichtige Software-Reviews 50



There's a new kind of coding I call "vibe coding", where you fully give in to the vibes, embrace exponentials, and forget that the code even exists.

# Vibe-Coding

[vaɪb kɔʊ.dɪŋ]

In seiner explorativsten Form vertraut der Nutzer vollständig darauf, dass der KI-Output wie beabsichtigt funktioniert. Wie **Karpathy** es formulierte, ist das so, als würde man „vergessen, dass der Code überhaupt existiert“.

embarc.de Leichtgewichtige Software-Reviews 51



embark.de

Leichtgewichtige Software-Reviews

53

**N's Blog**

### Karpathy's 'Vibe Coding' Movement Considered Harmful

250,000+ views 550+ votes

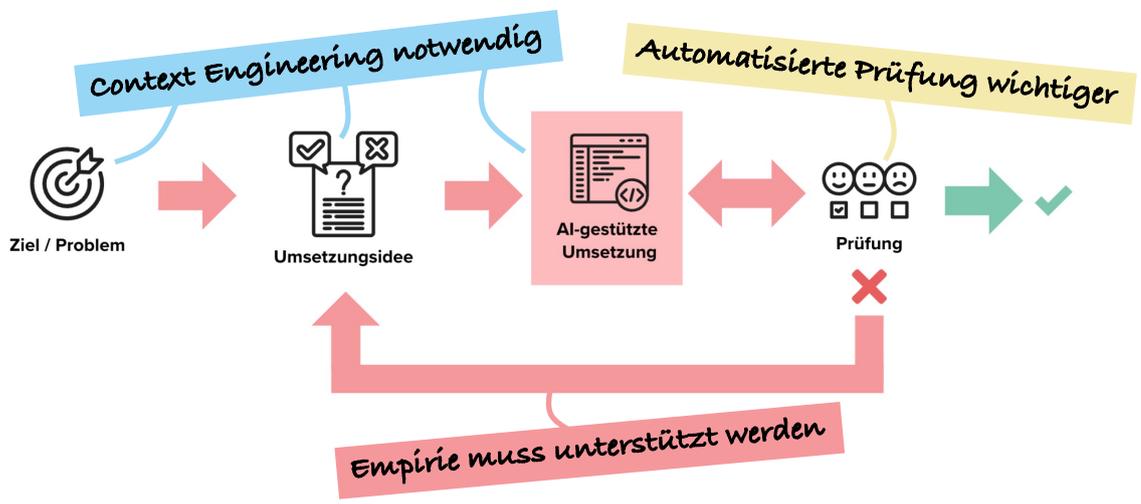
By Namanyay Goel  
Mar 27 2025 [Subscribe](#)

Last Tuesday at 1 AM, I was debugging a critical production issue in my AI dev tool. As I dug through layers of functions, I suddenly realized — unlike the new generation of developers, I was *grateful* that I could actually understand my codebase. That's when I started thinking more about Karpathy's recent statements on vibe coding.

For those who missed it, Andrej Karpathy recently shared his thoughts on what he calls "vibe coding" — essentially surrendering code comprehension to AI tools and hoping for the best. His exact words? *"I 'Accept All' always, I don't read the diffs anymore."*

I have learnt a lot from Karpathy and use AI tools daily, but there's a world of difference between augmenting your capabilities and completely surrendering your understanding

# Generative KI in der Entwicklung



embarc.de  
Leichtgewichtige Software-Reviews  
54

# Passende Standardrisiken

embarc.de  
Leichtgewichtige Software-Reviews  
55



## Booster-Pack: Agentic Development

### Risikobereiche:

- LLM-Verwendung
- Gen-AI Autonomie
- Ziele und Kontext
- Tests und Guardrails
- Agentic Development Know-how
- Weiche Faktoren und Prozesse
- Entwickelte Softwarelösung



## 1 - LLM-Verwendung

- Enge Kopplung an LLM
- Fehlender Determinismus
- Urheber- und Lizenzschwierigkeiten
- Grobgranularer Aufgabenschnitt ▶



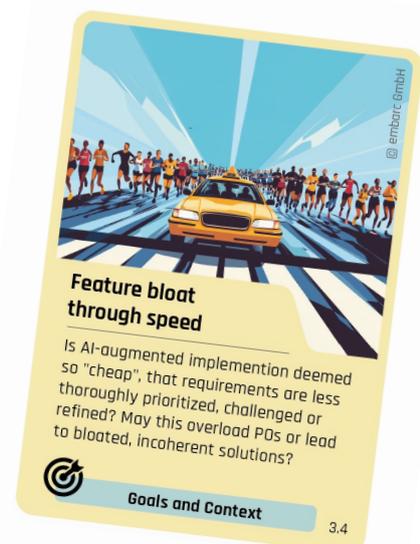
## 2 - Gen-AI Autonomie

- Frei agierende "Agent Swarms" ▶
- Wildwuchs von Abhängigkeiten
- Freizügig vergebene Berechtigungen
- Aufgeblähte Dokumentation



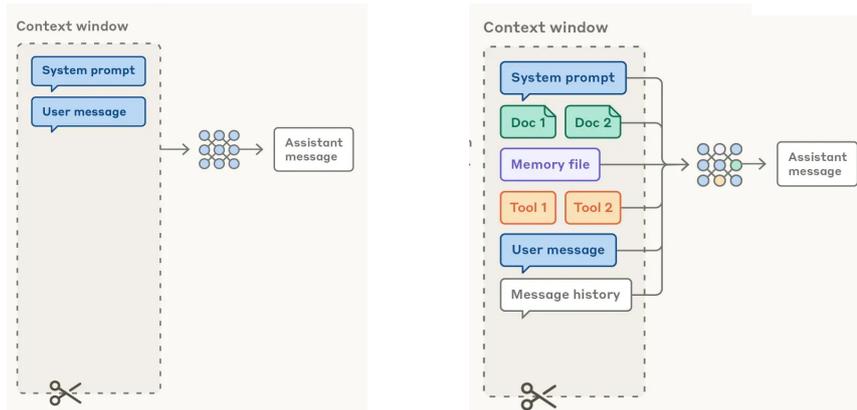
## 3 - Ziele und Kontext

- Implizite Ziele und Randbedingungen
- Überladener LLM-Kontext
- Tool-getriebenes Design
- Feature-Overkill ▶



# 3 - Ziele und Kontext

**Context engineering** is the discipline of building dynamic systems that supply an LLM with everything it needs to accomplish a task.



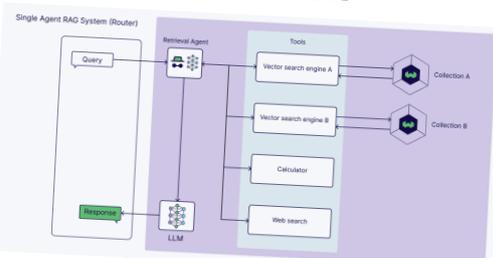
embarc.de  
Leichtgewichtige Software-Reviews  
60

# Viele Wege...

## Prompt-Template

Project: XYZ  
 Runtime: Java 21, Spring Boot 3.3  
 DB: Postgres 15 via Jooq  
 Forbidden: direct JDBC, reflection, thread pools (use virtual threads), mutable singletons  
 Approved libs: ...  
 API surface: REST only; no GraphQL

## Principles-RAG



## System Prompt

**Security & Privacy**  
 - Handle data as {CLASSIFICATION}. Never log secrets. Use {ORG\_SECRET\_MANAGER} for credentials.  
 - Only call approved endpoints on the allowlist: {SERVICES\_ALLOWLIST}. Deny any others.

**Compliance & License**  
 - Only introduce dependencies on the approved list: {PACKAGE\_ALLOWLIST}.  
 - If a needed package is not approved, propose an internal alternative or a zero-dependency approach.

**Architecture & Patterns**  
 - Default to the reference architecture: {REF\_ARCH\_URL\_OR\_DOC\_ID}.  
 - Use {LANG} + {FRAMEWORK} conventions and {STYLE\_GUIDE}. No global singletons. Prefer DI.

**Coding Standards**  
 - Enforce {LINTER/FORMATTER}. 100% typed public APIs. Unit tests for new code; snapshot tests only with justification.

**Observability**  
 - Instrument with {TRACING\_LIB} + {METRICS\_LIB}. Emit {REQUIRED\_FIELDS} per span/log.

**Output Contract**  
 - Always return: [Plan], [Code], [Why it complies], [Risks], [Tests], [Ops notes].  
 - If any rule conflicts, STOP and ask for an exception request using the "Exception Template".

embarc.de  
Leichtgewichtige Software-Reviews  
61



# Product Requirement Prompts (PRPs)

- Die Macht des ersten prompts...

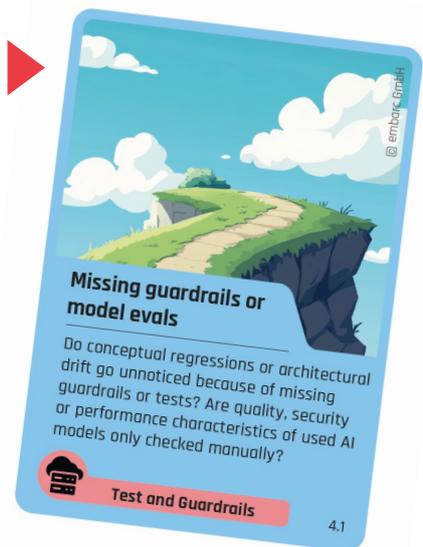
```

context-engineering-intro/
├── .claude/
│   ├── commands/
│   │   ├── generate-prp.md # Generates comprehensive PRPs
│   │   └── execute-prp.md # Executes PRPs to implement features
│   └── settings.local.json # Claude Code permissions
├── PRPs/
│   ├── templates/
│   │   ├── prp_base.md # Base template for PRPs
│   │   └── EXAMPLE_multi_agent_prp.md # Example of a complete PRP
│   └── examples/
│       ├── CLAUDE.md # Your code examples (critical!)
│       ├── GLOBAL.md # Global rules for AI assistant
│       ├── INITIAL.md # Template for feature requests
│       ├── INITIAL_EXAMPLE.md # Example feature request
│       └── README.md # This file
    
```



# 4 - Tests und Guardrails

- Fehlende Guardrails oder Modell-Evals
- Ineffektive Teststrategie
- Unzureichendes Sandboxing
- Fehlender Umgang mit Bias





## Tests schreiben lassen... Nicht immer gut...

embarc.de

```
def test_evidence_collection(self):
    """Test collecting evidence for validation decisions."""
    validation = {
        "stressor_id": "S1",
        "checked": True,
```

After thoroughly analyzing the Python test files in /tests/unit, I must deliver a harsh verdict: These tests are completely fake and do not test any actual implementation.

The Fundamental Problem

Language Mismatch:

- Implementation: TypeScript/Node.js (.ts files in /src)
- Tests: Python (.py files in /tests/unit)

This is absurd. You cannot test TypeScript code with Python unit tests.

```
{
  "type": "config",
  "location": "config/database.yml",
  "snippet": "replicas: [db-replica-1, db-replica-2]"
}
]
}

assert len(validation["evidence"]) > 0
assert all("type" in e and "location" in e for e in validation["evidence"])
```

Leichtgewichtige Software-Reviews

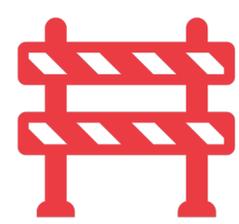
64



## Guardrails

embarc.de

- **Preventative guardrails** are proactive guardrails that specify the outer bounds of what developers can do.
- **Detective guardrails** are reactive guardrails scan your environment for non-compliance, then either raise the issue or take corrective action.



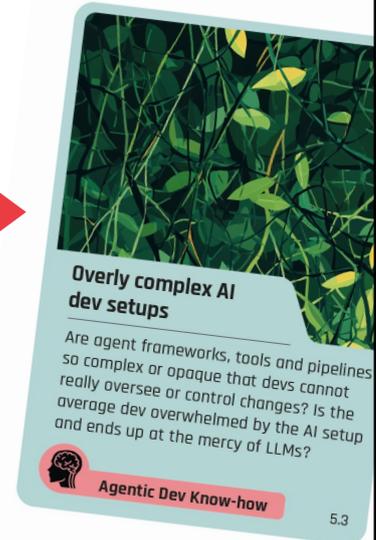
Leichtgewichtige Software-Reviews

65



## 5 - Agentic Development Know-how

- Fehlendes oder isoliertes AI-Wissen
- Freizügiger Umgang mit Innovation
- Übermäßig komplexes Entwicklungs-Setup
- Prompt-Jailbreaks und Datenlecks



## 6 - Weiche Faktoren und Prozesse

- Weniger Kommunikation im Team
- Fehlende Standards für den KI-Einsatz
- Übermäßige Abhängigkeit von LLMs
- Fehlende Verantwortlichkeiten / Governance



## 7 - Entwickelte Softwarelösung

- Aufgeblähter Code / komplexe Lösungen
- Unzureichende strukturelle Isolation
- Design- und Konzeptdrift
- Oberflächliche oder veraltete Lösungen



## Aufgeblähter Code / komplexe Lösungen

```

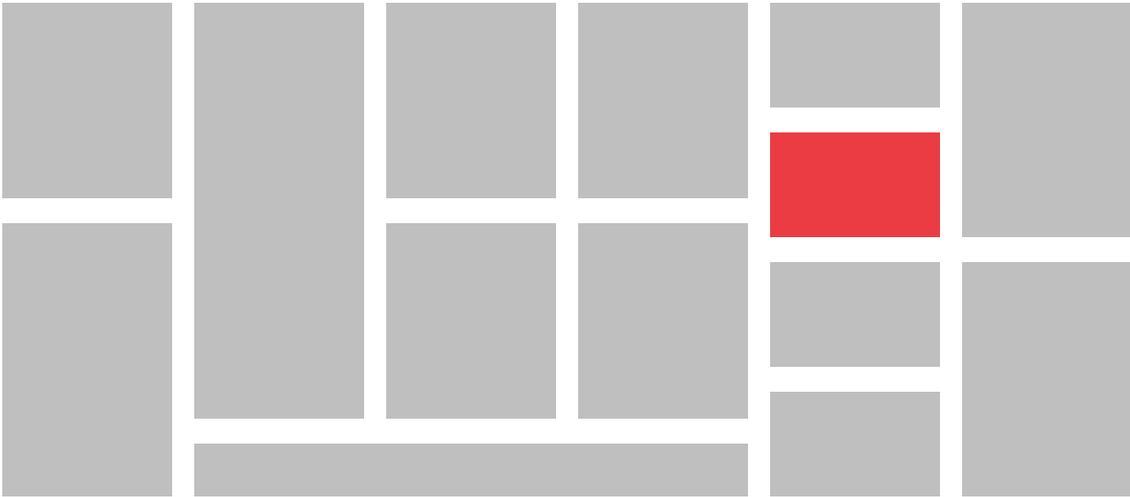
4316 rating: match.bgg_game?.rating || null,
4317 weight: match.bgg_game?.weight || null,
4318 best_players: match.bgg_game?.best_players || null,
4319 two_player_not_recommended_pct: match.bgg_game?.two_player_not_recommended_pct || null,
4320 },
4321 wienextra_game: {
4322   title: match.catalogGame.title,
4323   url: match.catalogGame.url
4324 },
4325 });
4326
4327 // Store enhanced data in KV with longer TTL (1 hour)
4328 await kv.put(`progress:wishlist:${jobId}:enhanced`, {
4329   step: 10,
4330   total_steps: 10,
4331   message: `Wishlist refresh completed with BGG ${jobId}`,
4332   completed: true,
4333   matches_found: enhancedMatchResults.length,
4334   matches: enhancedMatchResults,
4335   enhanced: true, // Flag to indicate this has been enhanced
4336   timestamp: new Date().toISOString()
4337 }, { expirationTtl: 3600 }); // 1 hour TTL
4338
4339 console.log(`🚀 Background BGG detail fetcher for job ${jobId} completed`);
4340
4341 } catch (error) {
4342   console.error(`❌ Error storing enhanced match data for job ${jobId}:`, error);
4343 }
4344 }
    
```

```

1140 });
1141
1142 // =====
1143 // Worker Export
1144 // =====
1145
1146 export default {
1147   /**
1148    * HTTP Request Handler
1149    */
1150   fetch: app.fetch,
1151
1152   /**
1153    * Scheduled Handler - Runs daily cron jobs
1154    */
1155   async scheduled(_event: ScheduledEvent, env: Bindings, _ctx: ExecutionContext) {
1156     console.log('🕒 Running scheduled daily sync...');
1157     try {
1158       // Note: Without queues, the cron job would need to be handled differently
1159       // For now, just log that it ran. You can add direct processing here if needed.
1160       console.log('✅ Scheduled task triggered');
1161     } catch (error) {
1162       console.error('❌ Scheduled task failed:', error);
1163     }
1164   }
1165 };
    
```



# Unzureichende strukturelle Isolation



# Weitere Informationen



## Risiko-Kartenset bestellen

embarc.de

Leichtgewichtige Software-Reviews

72



→ <https://www.embarc.de/lasr-kartenset-bestellen/>

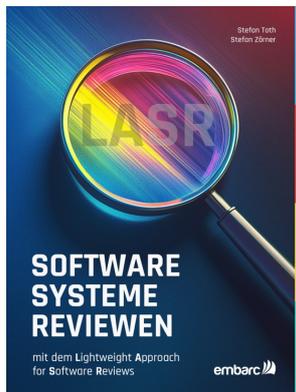


## Rabatt-Code für den Punsch

embarc.de

Leichtgewichtige Software-Reviews

73



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

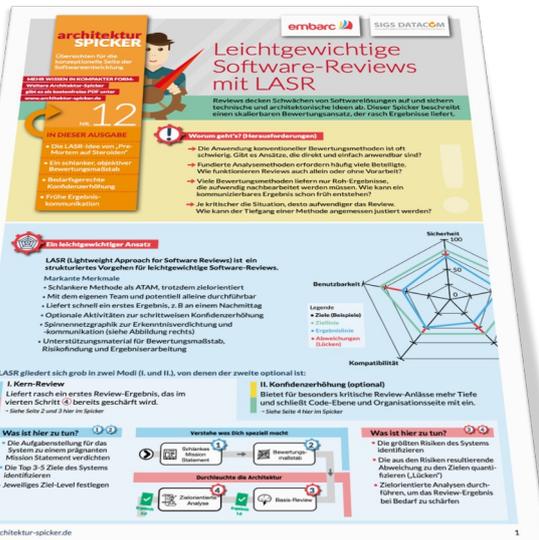
Für alle Teilnehmenden des  
Architekturpunsch 2025,  
reduziert den Preis um 45%

**Coupon: PUNSCH2025**



→ [leanpub.com/software-systeme-reviewen/c/PUNSCH2025](https://leanpub.com/software-systeme-reviewen/c/PUNSCH2025)

# Architektur-Spicker zum Thema ...



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

**Architektur-Spicker #12**  
Leichtgewichtige Software  
Reviews mit LASR

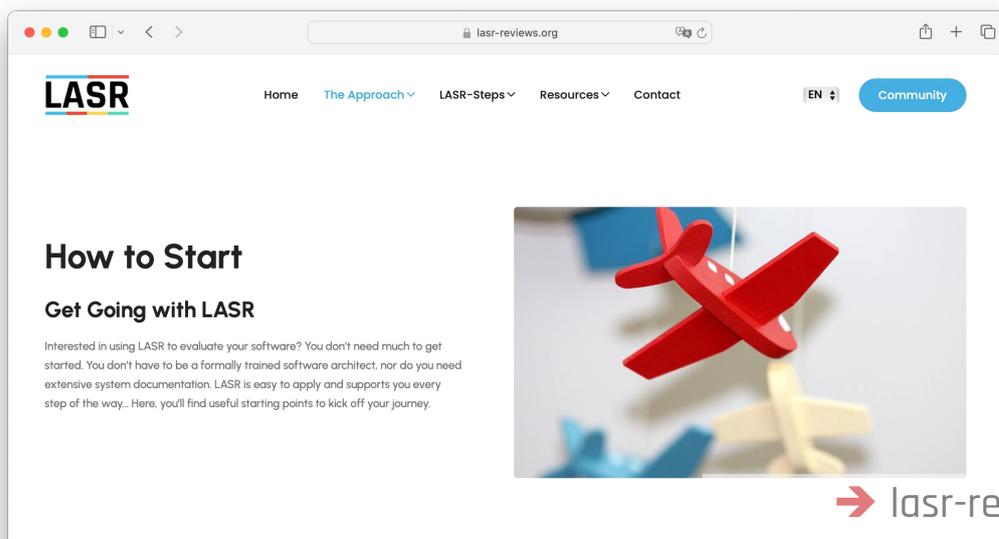


PDF, 4 Seiten  
Kostenloser Download.

→ [embarc.de/spicker](http://embarc.de/spicker)



# „Offizielle“ LASR-Webseite (EN + DE)



→ [lasr-reviews.org](http://lasr-reviews.org)



## Feedback & Fragen?

Wir freuen uns auf Fragen,  
Diskussionen, Ausprobieren!

