

# Nachhaltige Dokumentation von Architekturentscheidungen

# Das Ausgangsproblem

**Stell dir vor,  
du wirst in ein Legacy-  
Projekt mit unbekannter  
Code-Base geworfen.**

# Aber nichts ist dokumentiert!



# Warum ist das ein Problem?

**Warum Entscheidungen getroffen wurden, ist oft nicht mehr nachvollziehbar.**

**Teams verändern sich.**

**Anforderungen verändern sich.**

**Technologien verändern sich.**

# Ohne den relevanten Kontext hat man nur eine Wahl:

A. Die Entscheidung  
hinnehmen.

B. Die Entscheidung  
ablehnen.

# Beide Ansätze haben (große) Nachteile.

Der sinnvollste Weg, diesem Problem entgegenzuwirken, ist Dokumentation.

"Waaas?  
Dokumentation?"

"Wir arbeiten \$AGIL!"

§AGIL zu arbeiten  
bedeutet nicht,

Dokumentation zu  
vernachlässigen.

**§AGIL bedeutet, dass  
Dokumentation Mehrwert  
liefern muss.**

**Je umfangreicher die Dokumentation, umso wahrscheinlicher ist sie nicht mehr aktuell.**

**Nur aktuelle Dokumentation liefert Mehrwert.**

**Je schlanker die Dokumentation, umso  
wahrscheinlicher bleibt sie aktuell.**

**Schlankere Dokumentation kann nachhaltig sein.**

# Nachhaltige Architekturentscheidungen

**Was bedeutet es,  
eine Architektur-  
entscheidung nachhaltig  
zu treffen?**

# **Der Artikel *Sustainable Architectural Design Decisions* [6] definiert 5 Kriterien:**

- 1. Strategisch**
- 2. Messbar**
- 3. Erreichbar**
- 4. Gut eingebettet**
- 5. Zeitlos**

# 1. Strategisch

**Die strategischen Konsequenzen (also die  
Langzeitfolgen) der Entscheidung sollten bekannt  
sein.**

## 2. Messbar

**Die Ergebnisse der Entscheidung sollten messbar sein, idealerweise mittels objektiver Kriterien.**

### 3. Erreichbar

**Die Entscheidung sollte umsetzbar sein.**

Am besten weder over- noch under-engineered. 😊

## 4. Gut eingebettet

**Die Entscheidung sollte auf Fachwissen basieren.**

**Sie sollte in den Kontext der Firma, des Projekts und des Teams passen.**

## 5. Zeitlos

**Die Entscheidung sollte auf Annahmen basieren, die mit der Zeit nicht an Wert verlieren.**

**"When in doubt, choose boring technology."**

**Mit diesen Kriterien gewappnet, bleibt noch eine  
Frage:**

**Wie erreichen wir, nachhaltig zu dokumentieren?**

# Architecture Decision Records (ADRs)

**ADRs sind kurze Textdateien,**

**die eine einzelne Architekturntscheidung  
dokumentieren.**

# Sie sollten drei Dinge umfassen:

Kontext

Beschreibung

Konsequenzen

# Kontext

- Technische,
- betriebliche,
- soziale, oder
- politische

**Rahmenbedingungen mit direktem Einfluss auf die Architekturentscheidung.**

# Beschreibung

**Eine *knappe* Beschreiung der Entscheidung, mit  
Outline der notwendigen Schritte.**

# Konsequenzen

**Die erwarteten Konsequenzen, sobald die Entscheidung umgesetzt wurde.**

Idealerweise inklusive eines geeigneten Gegenmittels.

# ADR Templates

# Nygard (aka "das OG Template")

- 1. Titel**
- 2. Kontext**
- 3. Entscheidung**
- 4. Status**
- 5. Konsequenzen**

# Y-Statements

- 1. Kontext**
- 2. "angesichts"**
- 3. "haben wir entschieden"**
- 4. "und abgelehnt"**
- 5. "um zu erreichen"**
- 6. "akzeptierend, dass"**

# Y-Statements (Beispiel)

Im Kontext eines Web-Shop-Services, im Angesicht notwendiger Datenkonsistenz, haben wir entschieden, dass wir das Database-Session-State-Pattern und abgelehnt dass wir das Client-Session-State-Pattern nutzen wollen, um Datenkonsistenz zu erreichen, akzeptierend, dass eine Session-Datenbank implementiert werden muss.

Tooling

**ADRs brauchen nicht notwendigerweise Tooling,  
es kann einem die Arbeit damit aber vereinfachen.**

adr-tools<sup>1</sup>

**"A command-line tool for working with a log of  
Architecture Decision Records (ADRs)."**

Basiert auf dem *Nygard*-Template.

<sup>1</sup> <https://github.com/npryce/adr-tools>

# **Erstelle ein ADR-Verzeichnis im Projekt-Root:**

```
$ adr init doc/architecture/decisions
```

# Lege einen neuen ADR an:

```
$ adr new "Implement in PHP as part of the monolith"
```

**Auch wenn eine getroffene Entscheidung ersetzt wird, sollte sie aus historischen Gesichtspunkten trotzdem behalten werden.**

```
$ adr new -s "$ID" "Implement as Go micro service"
```

# Eine "Definition of Done"

**Jetzt wissen wir, wie eine Architekturentscheidung  
nachhaltig getroffen und nachhaltig dokumentiert  
werden kann.**

**Aber woran machen wir fest, dass wir fertig sind,  
unsere Entscheidung zu dokumentieren?**

# **Wir brauchen eine "Definition of Done"!**

**Basierend auf [1] mache ich folgenden Vorschlag<sup>2</sup>:**

<sup>2</sup> Formuliert als Fragen, um Diskussion zu enablen. 😎

# 1. Sind wir sicher, dass die Architektur funktioniert?

**2. Haben wir zwischen  
mindestens 2 Ansätzen  
abgewogen?**

**3. Haben wir in  
ausreichend großer  
Runde diskutiert und eine  
gemeinsame Sicht auf die  
Architektur?**

# 4. Haben wir den Ausgang der Entscheidung dokumentiert?

# 5. Wissen wir, wie wir die Entscheidung umsetzen, bewerten und widerrufen<sup>3</sup>?

<sup>3</sup> Falls notwendig.

# Zusammenfassung

**ADRs enablen uns,  
aktiv an einem  
nachhaltigen  
Entscheidungsfindungs-  
prozess teilzunehmen.**

# Sie sollten ...

- 1. nah beim Objekt der Entscheidung liegen.<sup>4</sup>**
- 2. den Kontext und die Konsequenzen erfassen.**
- 3. die Dokumentation schlank und relevant halten.**

<sup>4</sup> Zum Beispiel direkt im GitHub-Repository im Fall von Source Code.

# Weiteres Lesematerial



- [1] A DoD for Architectural Decision Making
- [2] adr-tools
- [4] Architectural Decisions – The Making Of
- [5] Documenting Architecture Decisions
- [6] Sustainable Architectural Design Decisions
- [7] Love Unrequited
- [8] Y-Statements